WEBWALKER: BENCHMARKING LLMS IN WEB TRAVERSAL

Jialong Wu[♠♣*], **Wenbiao Yin**[♣], **Yong Jiang**[♣†], **Zhenglin Wang**[♠], **Zekun Xi**[♣], **Runnan Fang**[♣], **Linhai Zhang**[◇], **Yulan He**[◇], **Deyu Zhou**[♠†], **Pengjun Xie**[♣], **Fei Huang**[♣]
[♠]Southeast University [◇]King's College London [♣]Alibaba Group
jialongwu@{alibaba-inc.com, seu.edu.cn}

ABSTRACT

Retrieval-augmented generation (RAG) demonstrates remarkable performance across tasks in open-domain question-answering. However, traditional search engines may retrieve shallow content, limiting the ability of LLMs to handle complex, multi-layered information. To address it, we introduce **WebWalkerQA**, a benchmark designed to assess the ability of LLMs to perform web traversal. It evaluates the capacity of LLMs to traverse a website's subpages to extract high-quality data systematically. We propose **WebWalker**, which is a multi-agent framework that mimics human-like web navigation through an explore-critic paradigm. Extensive experimental results show that WebWalkerQA is challenging and demonstrates the effectiveness of RAG combined with WebWalker, through the horizontal and vertical integration in real-world scenarios.

1 INTRODUCTION

Large Language Models (LLMs) have demonstrated impressive capabilities across a wide range of natural language processing tasks (Ouyang et al., 2022; OpenAI, 2022a). While their knowledge base remains static post-training, integrating external search engines via retrieval-augmented generation (RAG) allows LLMs to retrieve up-to-date information from the web, enhancing their utility in dynamic, knowledge-intensive scenarios (Lewis et al., 2020). However, traditional online search engines, *e.g.*, Google or Bing, perform horizontal searches of queries and may not effectively trace the deeper content embedded within websites.

Interacting with the web pages and digging through them can effectively address this issue. Previous works related to web pages focus on addressing action-based requests, such as Mind2Web (Deng et al., 2023) and WebArena (Zhou et al., 2024a); these HTML-based instruction-action benchmarks face challenges such as excessively noisy information and overly long inputs, which can significantly hinder performance due to limitations in long-context understanding. Additionally, they fail to capture the complexities of real-world scenarios where relevant information is buried deep within web pages and requires multiple layers of interaction.

To fill this gap, a new task **Web Traversal** is proposed, given an initial website corresponding to a query, systematically traverses web pages to uncover information. We propose **WebWalkerQA**, designed specifically to evaluate LLMs on their ability to handle queries embedded in complex, multi-step web interactions on a given root website. WebWalkerQA focuses on text-based reasoning abilities, using a Question-Answer format to evaluate traversal and problem-solving capabilities in web scenarios. We constrain actions to "*click*" to evaluate the agent's navigation and information-seeking capabilities. This paradigm is more targeted and aligns better with practical applications. WebWalkerQA reflects real-world challenges, emphasizing the ***depth*** of the source information across education, conference, organization, and game domains, where official sources are published and paths to information are more structured with clickable buttons and reasoning logic. Several types, including `multi-source` and `single-source` QAs, are developed to evaluate the ability of LLMs to mimic different human web-navigation paradigms.

[*] Work done during internship at Tongyi Lab, Alibaba Group.
[†] Corresponding Author.

Additionally, we introduce a strong baseline **WebWalker**, a multi-agent framework designed to emulate human-like web navigation through vertical exploration. The framework consists of an explorer agent and a critic agent. Given the need for reasoning capabilities to navigate and interact with web pages effectively, the explorer agent is built upon the ReAct framework (Yao et al., 2023), leveraging a thought-action-observation paradigm, while the critic agent is responsible for maintaining memory and generating responses based on the exploration conducted by the explorer agent.

We evaluate the performance of the WebWalker, built on various mainstream LLMs, including both closed-source and open-sourced, using WebWalkerQA as the benchmark. However, even with the most powerful LLMs as the backbone, its performance on WebWalkerQA remains suboptimal, thereby validating the challenge posed by WebWalkerQA.

We then conduct further experiments to validate the integration with the RAG for information-seeking QA tasks. Our findings are as follows: ***(i)*** Web navigation still requires efforts in tasks that demand planning and reasoning; ***(ii)*** By combining RAG with the WebWalker, this horizontal and vertical coordination proves effective; ***(iii)*** Vertical exploration of pages offers a promising direction for scaling inference time in RAG systems.

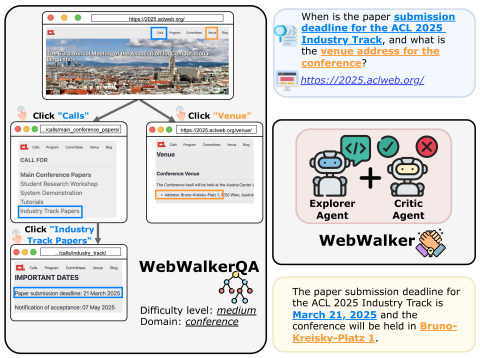


Figure 1: A `multi-source` QA[1] example from WebWalkerQA that requires traversing web pages to gather information for answering the given question.

The contributions of our work are as follows:

- We construct a challenging benchmark, **WebWalkerQA**, which is composed of 680 queries from four real-world scenarios across over 1373 webpages.
- To tackle the challenge of web-navigation tasks requiring long context, we propose **WebWalker**, which utilizes a multi-agent framework for effective memory management.
- Extensive experiments show that the WebWalkerQA is challenging, and for information-seeking tasks, vertical exploration within the page proves to be beneficial.

2 Related Work

2.1 Web-Oriented Benchmark

Before the era of LLMs, several web-oriented benchmarks had already been proposed (Liu et al., 2018; Xu et al., 2021; Humphreys et al., 2022; Yao et al., 2022; Mialon et al., 2024; Xu et al., 2024). LLMs are capable of interacting with complex environments, like the open web in HTML or DOM format (Tan et al., 2024), leading to the development of an increasing number of benchmarks aimed at evaluating the interaction capabilities of LLMs with web content. The widely used benchmark today, Mind2Web (Deng et al., 2023), is a dataset designed for evaluating web agents that follow instructions to complete complex tasks, typically through multiple-choice questions. Subsequent works have extended the interaction to the vision domain, incorporating information from screenshots (Zheng et al., 2024a;b; He et al., 2024a; Koh et al., 2024a; Cheng et al., 2024). The web-oriented benchmark is becoming progressively more human-like, vision-centric, and increasingly broad, complex, and realistic (Liu et al., 2024; Hong et al., 2024; Kim et al., 2024; Zhang et al., 2024c). The most closely to ours are the MMInA (Zhang et al., 2024c) and AssistantBench (Yoran et al., 2024), both of which focus on time-consuming tasks that require navigation across multiple pages. In our work, WebWalkerQA takes the form of QA pairs. Unlike all previous works, we construct both single-source and multi-source queries from the width perspectives of the website, aiming to simulate two types of page exploration patterns typically exhibited by humans. The comparison between WebWalkerQA and other benchmarks is shown in Table 2.

[1] In our paper, `multi-source` refers to the requirement of information from multi distinct web pages.

	Language	Format	Depth	Width	Hop	# Pages
Mind2Web (Deng et al., 2023)	En	Multi-choice	✗	✗	✗	100
WebArena (Zhou et al., 2024a)	En	Action	✗	✗	✗	6
AssistantBench (Yoran et al., 2024)	En	QA	✗	✓	✓	525
MMInA (Zhang et al., 2024c)	En	Action	✗	✓	✓	100
GAIA (Mialon et al., 2024)	En	QA	✗	✓	✓	-
WebWalkerQA	En&Zh	QA	✓	✓	✓	1373

Table 1: Comparison between WebWalkerQA and other benchmarks. **Depth** refers to the extent of exploration required on a given website. **Width** denotes whether answering a query necessitates multiple sources. **Hop** indicates whether multiple steps are required to complete the task. **#Pages** refers to the number of webpages involved.

2.2 Agents on Web-Navigation

Based on web-oriented benchmarks, numerous web agents have been proposed (Nakano et al., 2021; Liu et al., 2023; Zhou et al., 2023; Lai et al., 2024; Zhou et al., 2024b). Web agents primarily follow two lines of development: one leverages a small language model trained specifically to filter actions or identify relevant HTML elements (Zheng et al., 2024a; Deng et al., 2024; Furuta et al., 2024). The other line focuses on prompting LLMs (Reddy et al., 2024; Song et al., 2024; Koh et al., 2024b), where different agentic modules are used to guide the model in accomplishing complex web navigation tasks more effectively. In addition, with the rise of visual web-oriented benchmarks, many agents now use screenshots as sensory input (He et al., 2024b; Abuelsaad et al., 2024; Iong et al., 2024). Unlike previous works, WebWalker specializes in information-seeking by reasoning over HTML button data. It emulates human-like page interactions with web pages to access reliable, authoritative information utilizing a multi-agent framework.

3 WebWalkerQA

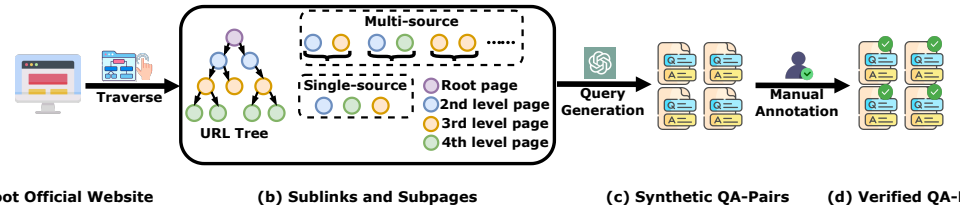


Figure 2: Data Generation Pipeline for **WebWalkerQA**. We first collect root official websites across conference, organization, education, and game domains. Then we mimic human behavior by systematically clicking and collecting subpages accessible through sublinks on the root page. Using predefined rules, we leverage GPT4o to generate synthetic QA-pairs based on the gathered information, followed by manual verification to ensure accuracy and relevance.

We present WebWalkerQA in this section, starting with an overview of the data collection process to ensure quality (§3.1), followed by a discussion of WebWalkerQA's statistics (§3.2). Finally, we introduce the new task, Web Traversal, and describe the evaluation metrics for WebWalkerQA(§3.3).

3.1 Data Collection

To make the annotation process cost-efficient and accurate, we employ a two-stage funnel annotation strategy, combining LLM-based and human annotation. In the first stage, GPT-4o (OpenAI, 2022b), performs initial annotations, followed by a second stage, where crowd-sourced human annotators conduct quality control and filtering to refine the final results. The overall data collection pipeline is illustrated in Figure 2.

LLM-based Annotation The collection pipeline is outlined as follows:

- **Step1**: Traverse official websites recursively, collecting information on accessible sub-links and their respective pages.
- **Step2**: Construct queries based on the provided page information and specified role, such as focusing on the solo page or considering both pages simultaneously.
- **Step3**: Verify and filter for legitimate queries that deviate from natural, human-like phrasing, retaining only QA pairs with short answers containing entities.

The additional details, including step-specific prompts and case examples, are provided in Appendix E. As illustrated in Figure 2 (b), our dataset construction includes both `multi-source` and `single-source` types, corresponding to two types of human information-seeking behaviours within web pages. The `single-source` type simulates a user deeply exploring a single piece of information hidden within web pages, while the `multi-source` type simulates `multi-source` scenarios where users rely on multiple pages to solve a query. Notably, the `multi-source` QA tasks can not be easily exploited by search engine shortcuts (Mavi et al., 2024).

Human Annotation After the synthetic queries are generated by LLM, human annotators can rewrite and calibrate the questions and answers to ensure the QA pairs are correct and consistent.

3.2 Data Statistics

Through such data construction method with LLM and human participation, we obtain 680 question-answer pairs for WebWalkerQA. The annotated case is shown in Figure 8. We will provide comprehensive statistics on WebWalkerQA, categorized by type, domain, and language.

Type WebWalkerQA contains two types of data: `multi-source` and `single-source` QAs. `Single-source` QAs are labeled as $\texttt{single-source}_i$, where $\texttt{i} \in [2, 4]$, denoting the depth of the corresponding subpage. Similarly, `Multi-source` QAs are labeled as $\texttt{multi-source}_i$, where $\texttt{i} \in [2, 8]$, representing the sum of the depths of the two associated subpages[2]. In other words, answering this query requires reading both pages simultaneously.

Difficulty Level We categorize the questions into three difficulty levels: easy, medium, and hard, based on the value of i. Specifically, $\texttt{single-source}_2$, $\texttt{single-source}_3$, and $\texttt{single-source}_4$ correspond to the easy, medium, and hard levels, respectively. Similarly, for multi-source questions, $\texttt{multi-source}_{2-4}$, $\texttt{multi-source}_{4-6}$, and $\texttt{multi-source}_{6-8}$ correspond to the easy, medium, and hard levels, respectively. The data statistics for the different data types are presented in Table 2.

`Single-source` QAs			`Multi-source` QAs		
Easy	Medium	Hard	Easy	Medium	Hard
80	140	120	80	140	120

Table 2: Dataset statistics on data difficulty level.

Domain WebWalkerQA encompasses four real-world domains: conference, organization, education, and game. These domains are selected because they provide authoritative information relevant to their respective fields, and their pages contain rich clickable content, offering substantial depth for exploration.

Language WebWalkerQA is a bilingual dataset that includes both Chinese and English[3], reflecting the most widely used and universal languages in real-world web environments.

The statistics of WebWalkerQA on domain and language are illustrated in Figure 3. The

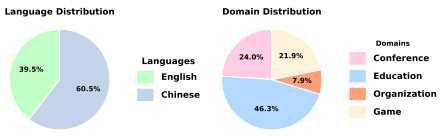


Figure 3: The language and domain distribution.

[2]Taking $\texttt{multi-source}_6$ as an example, it may refer to a query constructed from two 3rd level pages or from one page at the 2nd level and another at the 4th level.

[3]Classification based on the language of the root webpages.

proportions of the conference, organization, education, and game domains are 24.0%, 7.9%, 46.3%, and 24.0%, respectively. In terms of language distribution, Chinese and English account for 60.5%, 39.5%, respectively. WebWalkerQA features a diverse distribution of languages and domains to ensure a comprehensive evaluation.

3.3 Web Traversal Task and Evaluation

Formally, given an initial website URL U_{root} and a query $\mathcal{Q}$, which needs to be answered by exploring the website. The goal of this task is to gather enough information through page traversal to ultimately answer the query $\mathcal{Q}$. The task is to navigate the website to find the corresponding information.

WebWalkerQA can be evaluated from both *performance* and *efficiency* perspectives. using question-answering *accuracy* (*acc.*) as the performance metric and the *action count* (*A.C.*) of successful agentic executions answering correctly as the efficiency metric. Due to the varying lengths of generated text, it is challenging to perform exact match evaluation, even though we have controlled for short answers. We use GPT-4 as the evaluator, which determines the *correctness* of responses by comparing the predicted answer with the ground truth using CoT prompting strategy (Wei et al., 2022)[4].

4 WebWalker

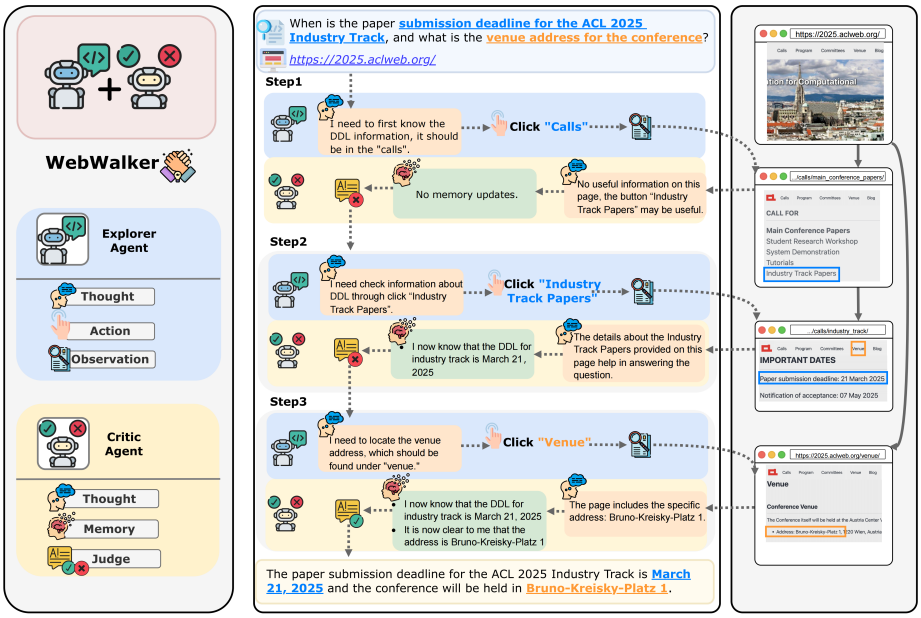


Figure 4: The overall framework of **WebWalker**.

We introduce **WebWalker**, a multi-agent framework designed to interact with web environments to answer queries. The WebWalker framework consists of two agents: an explorer agent and a critic agent. As illustrated in Figure 4, the explorer agent traverses the web pages in Thought-Action-Observation ($\mathcal{T}, \mathcal{A}, \mathcal{O}$) paradigms. The critic agent updates the memory until sufficient information is accumulated to effectively address the query. The details regarding prompts for both agents are presented in Appendix E.3.

4.1 Think then Explore

The explorer agent explores the subpages by interacting with HTML buttons on the page. At time step t, the explorer agent receives an observation $\mathcal{O}_t$ from the web environment and takes an action

[4] https://api.python.langchain.com/en/latest/langchain/evaluation.html, Details of the prompt for the evaluator are provided in Appendix F

$\mathcal{A}_t$, following the policy $\pi(\mathcal{A}_t|\mathcal{H}_t)$. The observation $\mathcal{O}_t = (p_t, l_t)$ consists of the information from the current page p_t and a set of clickable sublinks $l_t = \{button_i\}_{i=1}^{K}$, where each $button_i$ describes HTML button information for one of the K sublinks and have an associated URL. The action $\mathcal{A}_t$ involves selecting a URL of a subpage to explore and does **not** encompass answering the question. Specifically, we utilize the web page's markdown content along with clickable HTML buttons (and corresponding URL) extracted using Beautiful Soup as the observation for the current page. The context $\mathcal{H}_t = (\mathcal{T}_1, \mathcal{A}_1, \mathcal{O}_1, \cdots, \mathcal{O}_{t-1}, \mathcal{T}_t, \mathcal{A}_t, \mathcal{O}_t)$ represents the sequence of past observations and actions leading up to the current step t. The context will be updated, and this exploration process will continue until the critic agent determines to answer the query or the maximum number of steps is reached.

4.2 Think then Critique

Due to the policy $\pi(\mathcal{A}_t|\mathcal{H}_t)$ being implicit and the potentially large size of $\mathcal{H}_t$, motivated by pair programming (Williams et al., 2000; Noori & Kazemifard, 2015), we incorporate a critic agent into the WebWalker framework to address these challenges. The critic agent operates after each execution of the explorer agent. Its input consists of the query and the explorer's current observation. The critic initializes a memory to incrementally accumulate relevant information. Formally, at each step, t, following the execution of the explorer agent, the critic agent takes the query $\mathcal{Q}$ and the explorer's current observation and action $(\mathcal{O}_t, \mathcal{A}_t)$ as input. It then updates the memory $\mathcal{M}$, evaluates whether the gathered information is sufficiently complete to answer the query, and provides an answer once the required information is deemed sufficient.

5 Experiment

5.1 Experimental Setting

Baselines We choose widely recognized state-of-the-art agent frameworks, ReAct and Reflexion, as our baselines. **ReAct** (Yao et al., 2023) is a general paradigm that combines reasoning and acting with LLMs by multiple thought-action-observation steps. **Reflexion** (Shinn et al., 2024) is a single-agent framework designed to reinforce language agents through feedback.

Backbones To thoroughly assess the web traversal capabilities of existing LLM-based agents, we select models with a context window of at least 128K to accommodate the extensive length of page information. Given the inherent complexity of the task, we opt for models with at least 7B parameters. We validate a total number of nine models, including both closed-sourced and open-sourced ones: ***Closed-sourced LLMs*** GPT-4o[5] (OpenAI, 2022b); Qwen-Plus[6] (Team, 2024); ***Open-sourced LLMs*** Qwen2.5 series models (Yang et al., 2024) specifically, Qwen2.5-{7,14,32,72}B-Instruct.[7]

Implementation Details Considering the context limitation of models, our proposed WebWalker, along with two baselines, all operate in a zero-shot setting. We limit the number of actions K for the explorer agent to 15, meaning that the explorer agent can explore at most 15 steps. More implementation details are presented in Appendix B.

5.2 Main Results

The main results across six LLMs are presented in Table 3. The closed-source models outperform the open-source models in both performance and efficiency. For open-source models, performance and efficiency improves as the model size increases. Our proposed WebWalker framework outperforms Reflexion, which in turn outperforms React. We only counted the action count (*A.C.*) from correct executions, and as the model size increases, the *A.C.* grows, indicating that larger LLMs have enhanced long-range information-seeking ability. Even the best-performing WebWalker using GPT-4o as its backbone does not surpass 40%, highlighting the challenge posed by WebWalkerQA. It can

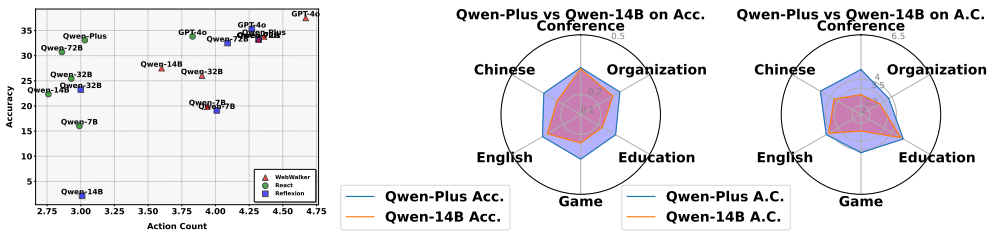
[5] https://platform.openai.com/docs/models#gpt-4o

[6] https://www.alibabacloud.com/help/en/model-studio/

[7] The LLaMA series models (Dubey et al., 2024) demonstrate limited ability to handle react-format instructions in our preliminary experiments.

Backbones	Method	Single-source QA						Multi-source QA						Overall	
		Easy		Medium		Hard		Easy		Medium		Hard			
		acc.	*A.C.*	*acc.*	*A.C.*	*acc.*	*A.C.*	*acc.*	*A.C.*	*acc.*	*A.C.*	*acc.*	*A.C.*	*acc.*	*A.C.*
Closed-Sourced LLMs															
GPT-4o	ReAct	53.75	2.53	45.00	3.34	30.00	5.61	32.50	2.34	31.43	3.97	15.00	6.77	33.82	3.83
	Reflexion	56.25	2.91	51.43	3.88	30.83	5.75	35.00	3.67	27.14	4.13	16.67	7.05	35.29	4.27
	WebWalker	55.00	2.97	50.00	3.43	30.00	6.02	47.50	4.00	34.29	3.85	15.83	6.57	**37.50**	4.67
Qwen-Plus	ReAct	48.75	1.67	48.57	2.69	28.33	4.00	35.00	2.60	27.86	3.11	14.17	6.55	33.08	3.03
	Reflexion	53.75	3.66	40.00	3.79	24.17	5.88	47.50	3.28	30.00	4.07	15.00	7.11	33.23	4.32
	WebWalker	55.00	3.72	47.14	3.19	30.00	6.13	35.00	3.89	27.14	4.39	15.00	7.38	**33.82**	4.36
Open-Sourced LLMs															
Qwen-2.5 -7B	ReAct	37.50	3.36	18.5 7	4.88	9.17	5.45	17.50	3.42	11.43	3.62	5.83	4.57	16.02	2.99
	Reflexion	37.50	4.03	25.00	3.48	11.67	4.57	30.00	2.66	15.71	5.45	4.17	7.8	19.11	4.07
	WebWalker	41.25	3.39	24.71	3.86	12.50	5.93	18.75	3.00	20.71	3.34	5.83	7.28	**19.85**	3.94
Qwen-2.5 -14B	ReAct	36.25	1.86	32.14	2.75	15.00	3.61	27.50	2.31	22.86	3.00	5.00	5.00	22.35	2.76
	Reflexion	46.25	2.21	34.29	2.83	15.00	4.44	36.25	2.51	22.86	3.34	5.83	5.42	25.14	3.01
	WebWalker	41.25	2.42	41.43	3.24	23.33	4.42	30.00	3.95	22.86	3.56	10.00	6.16	**27.50**	3.60
Qwen-2.5 -32B	ReAct	47.50	2.21	35.71	3.20	16.67	3.55	36.25	2.68	18.57	3.00	8.33	3.70	25.44	2.93
	Reflexion	42.50	2.52	32.86	2.65	16.67	3.90	31.25	2.84	23.57	3.12	5.83	5.00	23.26	3.00
	WebWalker	41.25	2.69	34.29	4.14	22.50	5.14	27.50	3.13	25.00	3.51	10.00	6.08	**26.02**	3.90
Qwen-2.5 -72B	ReAct	47.50	1.68	38.57	2.79	20.00	4.04	45.00	2.25	32.14	3.13	10.00	5.41	30.73	2.86
	Reflexion	57.50	3.04	44.29	3.88	28.33	5.82	36.25	3.62	25.00	3.60	12.50	6.26	32.50	4.09
	WebWalker	58.75	2.70	48.57	3.07	25.83	5.77	35.00	3.57	29.29	4.87	15.00	7.38	**33.26**	4.32

Table 3: Main results of three methods across closed-sourced and open-sourced LLMs as the backbone. *Acc.* and *A.C.* refer to accuracy and action count, respectively.



(a) (b)

Figure 5: (a) ▲ represents WebWalker using various models as backbones, ■ represents Reflextion with different backbone models, and ● denotes ReAct employing various backbone models. (b) Performance across domains and languages of WebWalker building upon Qwen-14B and Qwen-Plus.

be observed that as the depth increases or the number of sources required increases, the difficulty of acquiring the information needed to resolve the query becomes greater, resulting in a decline in accuracy performance. The performance distribution of accuracy and action count for different methods across various models is shown in Figure 5a. The further towards the top-right corner, the more effective and prolonged the web traversal becomes. We observe that increasing the model size or introducing reflection on the process of each action can address certain problems requiring multi-step solutions, thereby enabling long-distance task-solving capabilities in web traversal tasks.

5.3 Results across Domains and Languages

WebWalkerQA is a bilingual dataset encompassing both Chinese and English and spans multiple domains, including games, conferences, education, and organizations. The performance across different domains and languages is shown in Figure 5b. In the domain of **conference**, the framework demonstrates relatively superior performance, likely due to the more explicit and directive nature of the button information, which facilitates more straightforward inferences. The framework performs similarly in both Chinese and English, as the models we employed are both pre-trained and supervised-fine-tuned in a bilingual setting.

5.4 Error Assessment

Systems	Single-source QA			Multi-source QA			Overall
	Easy	Medium	Hard	Easy	Medium	Hard	
Close Book (No Retrieval)							
Gemini-1.5-Pro	12.50	7.86	8.33	11.25	6.43	5.00	8.08
o1-preview	16.25	10.00	9.17	7.50	10.71	6.67	9.85
Commerical Systems							
Doubao	45.00	15.00	18.33	13.75	8.57	10.00	16.76
Gemini-Search	40.00	32.14	29.17	30.00	23.57	17.50	27.94
ERNIE-4.0-8K	52.50	30.00	28.33	21.25	18.57	30.00	28.97
Kimi	77.50	41.43	40.83	26.25	26.43	22.50	37.35
Tongyi	41.25	45.00	41.67	40.00	41.43	34.17	40.73
Open-Sourced Systems							
Naive RAG	37.50	25.71	24.17	20.00	14.29	12.50	20.73
MindSearch	15.00	11.43	10.83	8.75	12.14	10.00	11.32
Avg.	37.50	24.29	23.42	19.86	18.02	16.48	-

Table 4: Accuracy results on Commercial and Open-sourced Searched-enhanced RAG systems.

For incorrect execution, errors can also be categorized into three types: refusal to answer or locating wrongly, reasoning error, and exceeding the maximum number of steps K. The prediction distribution is shown in Figure 6. The model with a relatively small number of parameters using the ReAct framework lacks the capacity to explore the depth of information, making judgments within just a few iterations of taking action, regardless of whether relevant information has been found. It tends to "*give up*" and exhibits characteristics of *impatience*. Introducing memory to manage the long context, along with an increase in model parameters, provides evidence that this phenomenon stems from the interference of long contexts having noisy information and the inherent capabilities of the model itself, consistent with the analysis drawn in §5.2. Some errors are categorized as reasoning errors, where the golden page has been found in the visited pages but is still incorrectly marked. This underscores the challenge of reasoning on page information in certain cases.[8]

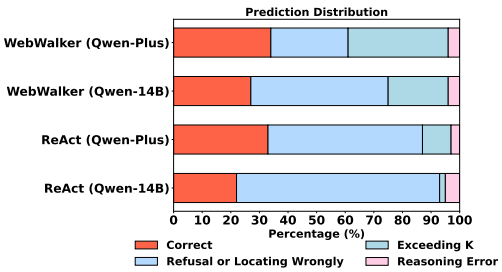


Figure 6: Predication distribution of WebWalker and React method building on Qwen-14B and Qwen-Plus.

6 Discussion

6.1 RAG Performance on WebWalkerQA

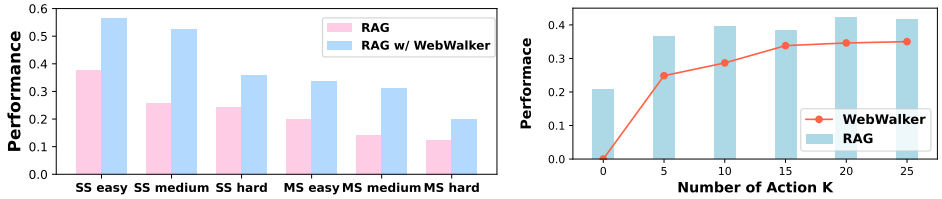
We evaluate the performance of RAG systems in tackling WebWalkerQA's challenges, specifically, whether they can retrieve deep information, presented in Table 4.

We first evaluate the performance under ***Close Book*** settings using the state-of-the-art model OpenAI o1 (OpenAI, 2024) and Gemini-1.5-Pro without retrieval. We then access the performance of several commercial and open-sourced RAG systems[9]. Without performing the search, even the strongest models exhibit very poor performance. WebWalkerQA is built on official websites with dynamically updated information, while pre-trained models rely on static knowledge limited by a cutoff date and lack dynamic updates[10]. Both commercial and open-sourced RAG systems exhibit relatively poor performance on WebWalkerQA, with the best result coming from Tongyi, which only reaches 40%. Commercial RAG systems are typically modular, consisting of various components such as rewrite, router, reranker, and others. Some systems, like ERNIE, may have stronger search capabilities for

[8]The corresponding case is presented in Appendix G.1.

[9]The commercial RAG systems are accessed through business-oriented API. The details of RAG systems are provided in Appendix C.

[10]The case study is shown in Appendix G.2.



(a) (b)

Figure 7: (a) Performance under standard RAG and RAG combined with WebWalker configurations. **SS** and **MS** denote `single-source` and `multi-source` QAs. (b) Overall performance on WebWalker and RAG combined WebWalker at varying values of K, using Qwen-Plus as backbones.

Chinese, resulting in higher values. For open-sourced RAG systems, `Multi-source` queries have lower accuracy than `Single-source` queries, which validates the challenge posed by WebWalkerQA, as search engines are **unable** to retrieve all relevant information in one or several single horizontal search attempts. Furthermore, as the difficulty increases, *e.g.* the depth of information growing deeper, the performance tends to deteriorate. Overall, search engines still face challenges when retrieving content that is buried deeper.

Findings (i): *RAG systems struggle with key challenges that require effective web traversal.*

6.2 WebWalker Combined with RAG System

The standard RAG system can be viewed as a *horizontal search* for relevant documents in response to a query, while WebWalker can be considered as a vertical exploration approach. WebWalker can seamlessly integrate into standard RAG systems to acquire deep information and enhance problem-solving capabilities. We integrate WebWalker building upon Qwen-2.5-Plus into the naive RAG system, and the detailed results are shown in Figure 7a. The core contribution of WebWalker is providing useful information for question answering; specifically, the memory $\mathcal{M}$ of the critic agent is append to the relevant documents to aid in generation. It is observed that, after the integration, performance has improved across all difficulty levels, especially in the `multi-source` category.

Findings (ii): *WebWalker can be a module in agentic RAG system, enabling vertical exploration .*

6.3 Scaling Up on Action Count K

Previous work (Yue et al., 2024) explored the inference scaling laws for the RAG system by examining the impact of increasing retrieved documents. We scale up the amount of $K \in \{5, 10, 15, 20, 25\}$ to study the impact of scaling during the inference phase when tracing source information. Figure 7b shows the results of scaling up, where larger values of K lead to better performance, validating the feasibility of vertical scaling within a certain range.

Findings (iii): *Scaling the process of digging through links could represent a potential direction for vertical exploration in RAG systems.*

7 Conclusion

We introduce WebWalkerQA, a benchmark for evaluating LLMs' web traversal abilities in complex, multi-step information-seeking tasks. We also proposed WebWalker, a multi-agent framework that mimics human-like web navigation, combining exploration and critique. Experiments show that WebWalkerQA effectively challenges RAG systems, and combining RAG with WebWalker improves web navigation performance. Our work highlights the importance of deep, vertical exploration in web-based tasks, paving the way for more scalable and reliable LLM-based information retrieval integrated with RAG.

REFERENCES

Tamer Abuelsaad, Deepak Akkil, Prasenjit Dey, Ashish Jagmohan, Aditya Vempaty, and Ravi Kokku. Agent-e: From autonomous web navigation to foundational design principles in agentic systems. In *NeurIPS 2024 Workshop on Open-World Agents*, 2024.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Jiangning Liu, Wenwei Zhang, Kai Chen, and Feng Zhao. Mindsearch: Mimicking human minds elicits deep ai searcher. *arXiv preprint arXiv:2407.20183*, 2024a.

Zehui Chen, Kuikun Liu, Qiuchen Wang, Wenwei Zhang, Jiangning Liu, Dahua Lin, Kai Chen, and Feng Zhao. Agent-FLAN: Designing data and methods of effective agent tuning for large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 9354–9366, Bangkok, Thailand, August 2024b. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.557. URL https://aclanthology.org/2024.findings-acl.557.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. *arXiv preprint arXiv:2401.10935*, 2024.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=kiYqbO3wqw.

Yang Deng, Xuan Zhang, Wenxuan Zhang, Yifei Yuan, See-Kiong Ng, and Tat-Seng Chua. On the multi-turn instruction following for conversational web agents. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8795–8812, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.477. URL https://aclanthology.org/2024.acl-long.477.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. doi: 10.48550/ARXIV.2407.21783. URL https://doi.org/10.48550/arXiv.2407.21783.

Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=efFmBWioSc.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. WebVoyager: Building an end-to-end web agent with large multimodal models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 6864–6890, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.371. URL https://aclanthology.org/2024.acl-long.371.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Hongming Zhang, Tianqing Fang, Zhenzhong Lan, and Dong Yu. Openwebvoyager: Building multimodal web agents via iterative real-world exploration, feedback and optimization. *arXiv preprint arXiv:2410.19609*, 2024b.

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.

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

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

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. *Advances in Neural Information Processing Systems*, 36, 2024.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. In *ICLR 2024 Workshop on Large Language Model (LLM) Agents*, 2024a.

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

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

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

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. *arXiv preprint arXiv:1802.08802*, 2018.

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

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating LLMs as agents. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=zAdUB0aCTQ.

Vaibhav Mavi, Anubhav Jangra, and Adam Jatowt. Multi-hop question answering, 2024. URL https://arxiv.org/abs/2204.09140.

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*, 2024. URL https://openreview.net/forum?id=fibxvahvs3.

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

Dang Nguyen, Jian Chen, Yu Wang, Gang Wu, Namyong Park, Zhengmian Hu, Hanjia Lyu, Junda Wu, Ryan Aponte, Yu Xia, et al. Gui agents: A survey. *arXiv preprint arXiv:2412.13501*, 2024.

Fariba Noori and Mohammad Kazemifard. Simulation of pair programming using multi-agent and mbti personality model. In *2015 Sixth International Conference of Cognitive Science (ICCS)*, pp. 29–36. IEEE, 2015.

OpenAI. Introducing ChatGPT, 2022a. URL https://openai.com/blog/chatgpt.

OpenAI. Gpt-4 system card, 2022b. URL https://cdn.openai.com/papers/gpt-4-system-card.pdf.

OpenAI. Introducing openai o1, 2024. URL https://openai.com/o1/.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Shuofei Qiao, Ningyu Zhang, Runnan Fang, Yujie Luo, Wangchunshu Zhou, Yuchen Jiang, Chengfei Lv, and Huajun Chen. AutoAct: Automatic agent learning from scratch for QA via self-planning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3003–3021, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.165. URL https://aclanthology.org/2024.acl-long.165.

Revanth Gangi Reddy, Sagnik Mukherjee, Jeonghwan Kim, Zhenhailong Wang, Dilek Hakkani-Tur, and Heng Ji. Infogent: An agent-based framework for web information aggregation. *arXiv preprint arXiv:2410.19054*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Yueqi Song, Frank Xu, Shuyan Zhou, and Graham Neubig. Beyond browsing: Api-based web agents. *arXiv preprint arXiv:2410.16464*, 2024.

Jiejun Tan, Zhicheng Dou, Wen Wang, Mang Wang, Weipeng Chen, and Ji-Rong Wen. Htmlrag: Html is better than plain text for modeling retrieved knowledge in rag systems. *arXiv preprint arXiv:2411.02959*, 2024.

Qwen Team. Qwen2.5: A party of foundation models, September 2024. URL https://qwenlm.github.io/blog/qwen2.5/.

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

Laurie Williams, Robert R Kessler, Ward Cunningham, and Ron Jeffries. Strengthening the case for pair programming. *IEEE software*, 17(4):19–25, 2000.

Kevin Xu, Yeganeh Kordi, Tanay Nayak, Ado Asija, Yizhong Wang, Kate Sanders, Adam Byerly, Jingyu Zhang, Benjamin Van Durme, and Daniel Khashabi. Tur[k]ingbench: A challenge benchmark for web agents, 2024. URL https://arxiv.org/abs/2403.11905.

Nancy Xu, Sam Masling, Michael Du, Giovanni Campagna, Larry Heck, James Landay, and Monica Lam. Grounding open-domain instructions to automate web support tasks. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou (eds.), *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1022–1032, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.80. URL https://aclanthology.org/2021.naacl-main.80.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, and et al. Qwen2 technical report. *CoRR*, abs/2407.10671, 2024. doi: 10.48550/ARXIV.2407.10671. URL https://doi.org/10.48550/arXiv.2407.10671.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. AssistantBench: Can web agents solve realistic and time-consuming tasks? In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 8938–8968, Miami, Florida, USA, November 2024. Association for Computational Linguistics. URL https://aclanthology.org/2024.emnlp-main.505.

Zhenrui Yue, Honglei Zhuang, Aijun Bai, Kai Hui, Rolf Jagerman, Hansi Zeng, Zhen Qin, Dong Wang, Xuanhui Wang, and Michael Bendersky. Inference scaling for long-context retrieval augmented generation. *arXiv preprint arXiv:2410.04343*, 2024.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. AgentTuning: Enabling generalized agent abilities for LLMs. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 3053–3077, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.181. URL https://aclanthology.org/2024.findings-acl.181.

Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Qingwei Lin, Saravan Rajmohan, et al. Large language model-brained gui agents: A survey. *arXiv preprint arXiv:2411.18279*, 2024a.

Wenqi Zhang, Ke Tang, Hai Wu, Mengna Wang, Yongliang Shen, Guiyang Hou, Zeqi Tan, Peng Li, Yueting Zhuang, and Weiming Lu. Agent-pro: Learning to evolve via policy-level reflection and optimization. *arXiv preprint arXiv:2402.17574*, 2024b.

Ziniu Zhang, Shulin Tian, Liangyu Chen, and Ziwei Liu. Mmina: Benchmarking multihop multimodal internet agents, 2024c. URL https://arxiv.org/abs/2404.09992.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v (ision) is a generalist web agent, if grounded. In *Forty-first International Conference on Machine Learning*, 2024a.

Boyuan Zheng, Boyu Gou, Scott Salisbury, Zheng Du, Huan Sun, and Yu Su. WebOlympus: An open platform for web agents on live websites. In Delia Irazu Hernandez Farias, Tom Hope, and Manling Li (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 187–197, Miami, Florida, USA, November 2024b. Association for Computational Linguistics. URL https://aclanthology.org/2024.emnlp-demo.20.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=oKn9c6ytLx.

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

Wangchunshu Zhou, Yixin Ou, Shengwei Ding, Long Li, Jialong Wu, Tiannan Wang, Jiamin Chen, Shuai Wang, Xiaohua Xu, Ningyu Zhang, et al. Symbolic learning enables self-evolving agents. *arXiv preprint arXiv:2406.18532*, 2024b.

Yuqi Zhu, Shuofei Qiao, Yixin Ou, Shumin Deng, Ningyu Zhang, Shiwei Lyu, Yue Shen, Lei Liang, Jinjie Gu, and Huajun Chen. Knowagent: Knowledge-augmented planning for llm-based agents, 2024. URL https://arxiv.org/abs/2403.03101.

A LIMITATIONS AND DISCUSSION

We discuss the following limitations:
Dataset Size: Due to the complexity of queries in the web-agent domain, similar to benchmarks such as AssistantBench (Yoran et al., 2024) (214) and MMIna (Zhang et al., 2024c) (1,050), GAIA (Mialon et al., 2024) (466), our proposed WebWalkerQA currently comprises 680 high-quality QA pairs. Additionally, we possess a collection of approximately 14k silver QA pairs, which, although not yet carefully human-verified, can serve as supplementary **training data** to enhance agent performance, leaving room for further exploration.
Multimodal Environment: In this work, we only utilize HTML-DOM to parse clickable buttons. In fact, visual modalities, such as screenshots, can also assist and provide a more intuitive approach (Nguyen et al., 2024; Zhang et al., 2024a; He et al., 2024b). We leave this for future work.
Agent Tuning: WebWalker is driven by prompting without additional training. We can use agent tuning to help LLMs learn web traversal. This involves fine-tuning models with golden trajectories, enabling them to take effective actions for completing information-seeking tasks (Zeng et al., 2024; Chen et al., 2024b; Zhang et al., 2024b; Qiao et al., 2024; Zhu et al., 2024).
Better Integration with RAG Systems: In §6.2, the root url is provided for the WebWalker to execute. To better integrate with the RAG system, one approach could be to first rewrite the query within the RAG system to refine the search, directing it to the query's official websites likely to contain relevant information. The WebWalker can then be used to extract useful information. Both the knowledge retrieved from the RAG system and the information mined by the WebWalker can be combined as augmented retrieval knowledge for generation, leading to a better result.

WebWalker can function independently as a **web information retrieval assistant** for a given webpage or **seamlessly integrate with RAG systems** to expand their scope. Under the agentic RAG paradigm, the *click* action proves to be highly effective.

B IMPLEMENTATION DETAILS

In this study, we utilize `Qwen-Agent`[11] as the foundational codebase for building and developing the baselines proposed WebWalker. The details of LLM hyperparameters for generation are as follows: $top_p = 0.8$. We sincerely thank the contributors and maintainers of `ai4crawl`[12] for their open-source tool, which helped us get web pages in a Markdown-like format. We will release the code of WebWalker in GitHub.

C DETAILS FOR RAG SYSTEMS

We select five mainstream commercial systems and two open-source systems for evaluation.

C.1 COMMERCIAL SYSTEMS

Doubao[13], ERNIE-4.0-8K[14], Tongyi, Kimi, and Gemini-Search are all accessed through their business-oriented API interfaces to ensure reproducibility. The detailed configuration of each API can be found in our codebase.

C.2 OPEN-SOURCED SYSTEMS

(a) **Mindsearch** (Chen et al., 2024a) is to mimic the human minds in web information seeking and integration, which can be instantiated by a multi-agent framework consisting of a WebPlanner and WebSearcher. (b) **Naive RAG built from scratch** We use Google to query the relevant terms and concatenate the information from the Top-10 returned links with the query to provide instructions for the Qwen-Plus to generate a response.

[11] https://github.com/QwenLM/Qwen-Agent
[12] https://github.com/unclecode/crawl4ai
[13] https://www.volcengine.com/docs/82379/1302004
[14] https://cloud.baidu.com/doc/WENXINWORKSHOP/s/clntwmv7t

Annotated Data Format

````
## JSON Format
The keys in the JSON include:
Question, Answer, Root_Url, and Info. The Info field contains
more detailed information, including Hop, Domain, Language,
Difficulty_Level, Source Website, and Golden_Path.
```
{
    "Question": "When is the paper submission deadline for the ACL 2025 Industry Track, and what is the venue address for the conference?",
    "Answer": "The paper submission deadline for the ACL 2025 Industry Track is March 21, 2025. The conference will be held in Brune-Kreisky-Platz 1.",
    "Root_Url": "https://2025.aclweb.org/",
    "Info":{
        "Hop": "multi-source",
        "Domain": "Conference",
        "Language": "English",
        "Difficulty_Level": "Medium",
        "Source_Website": ["https://2025.aclweb.org/calls/industry_track/","https://2025.aclweb.org/venue/"],
        "Golden_Path": ["root->call>student_research_workshop", "root->venue"]
    }
}
```
````

Figure 8: A JSON-format case in WebWalkerQA.

D Annotated Case

An annotated case is shown in Figure 8. The WebWalkerQA dataset will be available at Hugging-Face Datasets.

E Details on Annotation

E.1 Sources of Root Page

The root page is initially identified through a Google search using keywords such as "*conference official website*" or "*game official website*", followed by manual filtering. For the education domain, we choose the official websites of various university computer science departments, closely reflecting real-world scenarios. The distribution of the domain is shown in Figure 3.

E.2 Details on Prompts for Annotation

The prompts for GPT-4o-based initial annotation are presented below.

Prompts for Multi-source Data Annotation

Question Generate

You are a professional web content analyst. Based on the provided material, construct a query statement:
Sublink 1 URL; Sublink 1 INFO
Sublink 2 URL; Sublink 2 INFO
...
Sublink n URL; Sublink n INFO
Requirements:
1. **Core Goal of the Query**: Create a multi-step standalone query where the user needs to integrate information from at least two sublinks to find the final answer. The answer should be a single, clear, concise, and precise entity.
2. **Relevance of Sublinks**: The selected sublinks must have an intrinsic connection, and the answer should be derived by combining information from these two sublinks.
3. **Logical and Complex**: The constructed query should be as complex and specific as possible, challenging, and can leverage time, sequence, or commonly mentioned topics to construct a naturally

coherent reasoning process. Avoid questions about browsing history, browsing paths, etc., which have no practical value.
4. **Accuracy of the Answer**: Ensure the answer is accurate, concise, and closely connected to the logical chain constructed in the query.

Please return in JSON format, structured as follows:
{
"sublink_reason": "Describe why these specific sublinks were chosen and how they are interrelated.",
"sublinks": ["Selected sublink URL", "Selected sublink URL"],
"reason": "Explain the reason for designing this query and how it encourages the user to engage in multi-step reasoning.",
"query": "Your query statement",
"answer": "The answer to the query"
}
Sublink 1 URL; Sublink 1 INFO
Sublink 2 URL; Sublink 2 INFO
...
Sublink n URL; Sublink n INFO

Question-Answer Verify

You will act as a strict judge. You need to evaluate whether the given query can be accurately answered only by combining the information from two documents (doc1 and doc2) and the provided answer. Additionally, check if the answer is concise (as an entity or a judgment) and correct.

If the answer is incorrect, can be answered using only one document, or is not concise enough, you should return false.
If any document (doc1 or doc2) does not contain the necessary key information for the answer and only provides context for the query, you should return false.
If any document merely provides query background information unrelated to the answer and does not require combining information from both documents, you should return false.
If the answer is a long answer and not of an entity type, you should return false.
If the query is unnatural, doesn't appear as a complete query, or has a harsh tone, you should return false.
Each question should require combining information from both documents, meaning the answer results from multi-hop reasoning or multi-step reasoning, and it is concise for you to return true.
You are very strict, and any case failing to meet the above criteria should result in a false. Please return your result in JSON format as follows:
{
"reason": "Consider each of the conditions above in sequence to assess whether the query and answer meet the criteria. If they do meet the criteria, list the helpful parts from each doc for answering the question.",
"decision": "true/false"
}
{Doc1 INFO}; {Doc2 INFO}

Prompts for Single-source Data Annotation

Question Generate

Question-Answer Verify

You will act as a strict judge. You need to assess whether current knowledge from doc2 is required to accurately answer the given query based on the two provided documents (doc1 and doc2) and the given answer. Doc1 represents known knowledge, while doc2 represents current knowledge. Your task is to determine if the answer relies on doc2 to be accurately provided. Additionally, evaluate whether the answer is short (an entity or judgment) and correct.

If the answer is incorrect or not concise, return false.
If the necessary key information is found in the known knowledge doc1, also return false.
If the answer is a long answer and not of entity type, return false.
If the query is unnatural, not a complete query, or awkwardly phrased, return false.

The answer should result from multi−hop reasoning or multi−step reasoning, where multi−step reasoning indicates that the generated query is challenging and requires reasoning or calculation to answer, and only if the answer is concise should you return true.
You are extremely strict, and any requirements not met should result in a return of false.

Please return the result in JSON format as follows:
{
”reason”: ”Evaluate against the above conditions step by step, considering whether the query and answer meet the conditions. Use English to justify, and if they do, list the sections from doc2 that assist in answering the query.”,
”decision”: ”true/false”
}

E.3 Details Prompts for Agents

The prompts for the **Expoloer Agent** and **Critic Agent** are shown below.

Prompts for WebWalker

The Expoloer Agent

Digging through the buttons to find quailty sources and the right information. You have access to the following tools:

{tool_descs}

Use the following format:

Question: the input question you must answer
Thought: you should always think about what to do
Action: the action to take, should be one of [{tool_names}]
Action Input: the input to the action
Observation: the result of the action
... (this Thought/Action/Action Input/Observation can be repeated zero or more times)

Begin!

{query}

The Critic Agent

Critic
You are a critic agent. Your task is to analyze the given observation and extract information relevant to the current query. You need to decide if the observation contains useful information for the query. If it does, return a JSON object with a ”usefulness” value of true and an ”information” field with the relevant details. If not, return a JSON object with a ”usefulness” value of false.
Input:
– Query: ”<Query>”
– Observation: ”<Current Observation>”
Output (JSON):
{
”usefulness”: true,
”information”: ”<Extracted Useful Information>”
}
Or, if the observation does not contain useful information:
{
”usefulness”: false
}
– Query: {Query}
– Observation: {Observation}
Answer

CoT-QA Evaluator

You are a teacher grading a quiz.
You are given a question, the context the question is about, and the student's answer. You are asked to score the student's answer as either CORRECT or INCORRECT, based on the context.
Write out in a step by step manner your reasoning to be sure that your conclusion is correct. Avoid simply stating the correct answer at the outset.

Example Format:
QUESTION: question here
CONTEXT: context the question is about here
STUDENT ANSWER: student's answer here
EXPLANATION: step by step reasoning here
GRADE: CORRECT or INCORRECT here

Grade the student answers based ONLY on their factual accuracy. Ignore differences in punctuation and phrasing between the student answer and true answer. It is OK if the student answer contains more information than the true answer, as long as it does not contain any conflicting statements.
Begin!

QUESTION: {query}
CONTEXT: {answer}
STUDENT ANSWER: {result}
EXPLANATION
GRADE:"""

Figure 9: The prompt for evaluation.

You are a critic agent. Your task is to evaluate whether the accumulated useful information is sufficient to answer the current query. If it is sufficient, return a JSON object with a "judge" value of true and an "answer" field with the answer.
If the information is insufficient, return a JSON object with a "judge" value of false.
Input:
- Query: "<Query>"
- Accumulated Information: "<Accumulated Useful Information>"

Output (JSON):
```
{
    "judge": true,
    "answer": "<Generated Answer>"
}
```
Or, if the information is insufficient to answer the query:
```
{
    "judge": false
}
```
- Query: {Query}
- Accumulated Information: {Information}

F Details for Evaluation

F.1 Evaluator

The evaluator prompt is shown in Figure 9.

Root Url	https://www.mrs.org/
Question	How many hours in total would a person spend if they attended the **Inclusive Connections Lounge** activities from December 1 to 6, 2024, at the MRS Fall Meeting?
Answer	66 hours
Source Website	https://www.mrs.org/meetings-events/annual-meetings/2024-mrs-fall-meeting/meeting-events/broadening-participation/inclusive-connections-lounge
Website Information	

Table 5: The case requiring reasoning capability in web traversal task.

Question	Where and when will the **2025** MRS Fall Meeting take place?
Answer	Boston, Massachusetts; November 30 to December 5, 2025.
Prediction	As of my knowledge cutoff in **October 2023**, the MRS has not yet announced the exact dates or location for the 2025 MRS Fall Meeting.

Table 6: The case of time cutoff in predictions generated by o1.

G Case Study

G.1 Reasoning Error

As shown in Table 5, this question requires first locating the webpage related to the **Inclusive Connections Lounge**, followed by a comprehensive understanding of the information on the page to calculate the required time. In such cases, it is also necessary to account for the system's ability to perform time calculations or reasoning. Consequently, even when the source page is successfully located, errors might still occur if the system fails to process the time correctly.

G.2 Time Cut-off

As shown in Table 6, the cutoff date for o1's temporal data is October 2023, rendering it unable to provide answers regarding web information published beyond this point.