VERIGUI: VERIFIABLE LONG-CHAIN GUI DATASET

VeriGUI Team

ABSTRACT

Recent studies have delved into constructing autonomous agents capable of performing complex Graphical User Interface (GUI)-based computer tasks, with the potential to revolutionize human-computer interaction. Despite encouraging results, existing efforts mainly focus on short-term interactions and rely on outcome-only verification, thereby limiting their scalability in real-world GUI applications that demand long-horizon task decomposition and execution. In this work, we introduce VeriGUI, a novel verifiable long-chain GUI dataset designed to facilitate the development and evaluation of generalist GUI agents operating in realistic computer environments. Our dataset emphasizes two critical dimensions: (1) *long-chain complexity*, with tasks decomposed into a sequence of interdependent subtasks spanning hundreds of steps, explicitly designed to allow any subtask to serve as a valid starting point; and (2) *subtask-level verifiability*, which enables diverse exploration strategies within each subtask, while ensuring that each subtask-level goal remains verifiable and consistent. The dataset consists of GUI task trajectories across both desktop and web, *annotated by human experts*. Extensive experiments on VeriGUI using various agents with different foundation models reveal significant performance gaps in handling long-horizon tasks, highlighting the need for more robust planning and decision-making capabilities in GUI agents.

https://github.com/VeriGUI-Team/VeriGUI
https://huggingface.co/datasets/2077AIDataFoundation/VeriGUI

1 INTRODUCTION

Autonomous Graphical User Interface (GUI) agents have recently demonstrated extraordinary capabilities in interactive computer tasks by following high-level instructions (Wang et al., 2024; Zhang et al., 2024a; Nguyen et al., 2024), supporting diverse workflows from web browsing to desktop applications (Ning et al., 2025; Hu et al., 2024). Recent breakthroughs in Multimodal Large Language Models (MLLMs) (Zhang et al., 2024c; Team et al., 2023; Achiam et al., 2023; Bai et al., 2025; Liu et al., 2023) have enabled promising prototypes of such agents that can perform complex decision-making tasks without relying on hard-coded automation or domain-specific scripting (Tan et al., 2024; Xie et al., 2023). However, developing such general-purpose GUI agents involves multiple complex processes, as it requires the ability to perceive complex visual layouts (Hong et al., 2024; Gou et al., 2024; Cheng et al., 2024), plan over long action sequences (Zhang et al., 2024d; Agashe et al., 2024), and generalize across dynamic and heterogeneous platforms (Wu et al., 2024; Zhang et al., 2025). This also poses a new challenge: how to obtain high-quality datasets that capture diverse, realistic human-computer interactions at scale to evaluate these agents effectively (Deng et al., 2023; Li et al., 2025; Liu et al., 2024b).

To address this challenge, various datasets and benchmarks have been released to facilitate the development of autonomous GUI agents (Zhang et al., 2025; Yang et al., 2025; He et al., 2024). Despite encouraging results, existing GUI datasets still suffer from two major limitations. First, most recent datasets focus on relatively *short-term interactions* (Lù et al., 2024; Chen et al., 2025), where the agent can complete a task in just a few steps (*e.g.*, mostly less than 10 steps), typically by identifying a UI element and executing a corresponding action (Li et al., 2025; Deng et al., 2023). For example, a task like "Search for an email about the invoice" can typically be completed in just three steps: open the email app, click the search bar, and type the keyword. Such interactions rarely require long-horizon planning or multi-step reasoning (Gao et al., 2024; Bonatti et al., 2024; Zheng et al., 2024), both of which are essential for solving real-world workflows involving conditional

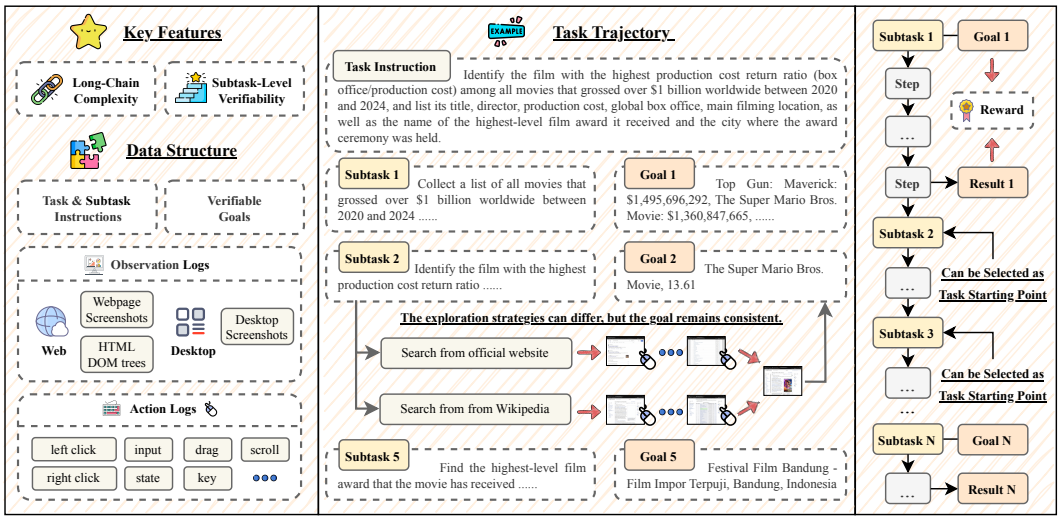


Figure 1: An overview of the VeriGUI dataset, which emphasizes (1) long-chain complexity, where each tasks consist of interdependent subtasks that span hundreds of steps, with each subtask serving as a valid starting point; and (2) subtask-level verifiability, enabling diverse exploration strategies while ensuring that the goal of each subtask is verifiable and consistent.

task dependencies and intermediate state tracking (Deng et al., 2023; Yang et al., 2025). Second, existing evaluation protocols typically rely on *outcome-only validation* such as checking whether the final page URL has been reached (Zhou et al., 2023a; Xie et al., 2024; Zhao et al., 2025). This coarse-grained supervision fails to capture the quality of intermediate subtasks, especially when tasks involve multiple interdependent subtasks (Pan et al., 2024). In such cases, when agents fail to achieve the desired goal, it is often unclear where or why the failure occurred, thereby making it difficult to support improvements to agent capability.

In this work, we introduce VeriGUI, a new verifiable long-chain dataset tailored for the development and evaluation of GUI agents. VeriGUI encompasses various richly annotated GUI task trajectories across desktop and web. All trajectories are carefully created and annotated by human experts, ensuring long-chain complexity and subtask-level verifiability, as shown in Fig. 1. (1) The long-chain complexity of VeriGUI features tasks that require agents to perform sequences of 4-8 interdependent subtasks with hundreds of GUI operations, often involving transitions across multiple applications or webpages. Notably, each subtask is designed to serve as a valid starting point, enabling agent evaluation across different task stages. To succeed, agents must engage in adaptive reasoning to manage dynamic task flows. This setup encourages the development of agents with robust planning, memory, and decision-making abilities across a wide range of complex GUI environments. (2) The subtask-level verifiability of VeriGUI enables a fine-grained assessment of intermediate results at every subtask rather than solely at the final outcome. Note that a subtask consists of multiple steps with specific GUI operations. Instead of verifying the low-level steps, the dataset focuses on evaluating whether the goal of each subtask has been correctly achieved, providing a more informative supervision signal. Thus, the dataset also supports open-ended interaction within each subtask, encouraging agents to explore diverse strategies to accomplish the goal of each subtask, rather than adhering to a fixed sequence of steps. Our core contributions are summarized as follows:

- We present VeriGUI, a large-scale, human-annotated dataset of verifiable long-chain GUI tasks designed to support research on autonomous agents in real-world computer environments.
- We design a comprehensive benchmark on top of VeriGUI, supporting multiple levels of evaluation, including task success rate, task completion rate, and action efficiency. This enables fine-grained analysis of agent capabilities across different stages of task execution and provides deeper insights into failure modes and planning bottlenecks.
- Extensive experiments with a range of various agents using state-of-the-art foundation models reveal substantial performance gaps on long-chain tasks, underscoring current limitations in complex planning and decision-making in GUI agents.

Table 1: Comparison of existing GUI datasets and benchmarks with VeriGUI. Platform indicates whether the benchmark supports web or desktop applications. #Steps refers to the average or range of steps per task. Verifiability describes how task trajectories are validated. Human demonstration indicates the presence of collected expert trajectories. Executability denotes whether an executable environment is available. Interaction defines the structure of the action space. Note that for VeriGUI, the #Steps reflects the average number of GUI operations in the human demonstration dataset.

Datasets and Benchmarks	Platform	#Steps	Verifiability	Human Demonstration	Executability	Interaction
VisualWebArena (Koh et al., 2024)	Web	9.6	Outcome	✗	✓	Web Element
VisualWebBench (Liu et al., 2024b)	Web	1.0	Outcome	✓	✗	Grounding
WebArena (Zhou et al., 2023a)	Web	–	Outcome	✗	✓	Web Element
Mind2Web (Deng et al., 2023)	Web	7.3	Step	✓	✓	Web Element
WebShop (Yao et al., 2022)	Web	11.3	Outcome	✗	✓	Web Element
WebVoyager (He et al., 2024)	Web	[3, 15]	Outcome	✓	✓	Web Element
WebCanvas (Pan et al., 2024)	Web	8.4	Step	✗	✓	Web Element
WebWalker (Wu et al., 2025)	Web	4.6	Outcome	✗	✓	Web Element
WebLINX (Lù et al., 2024)	Web	43.0	Outcome	✓	✗	Web Element
OSWorld (Xie et al., 2024)	Desktop + Web	[1, 15]	Outcome	✗	✓	GUI Operations
AgentStudio (Zheng et al., 2024)	Desktop + Web	[1, 30]	Outcome	✗	✓	GUI Operations
GUI-World (Chen et al., 2025)	Desktop + Web	–	Outcome	✓	✗	GUI Operations
WindowsAgentArena (Bonatti et al., 2024)	Desktop + Web	8.1	Outcome	✗	✓	GUI Operations
WorldGUI (Zhao et al., 2025)	Desktop + Web	–	Outcome	✗	✓	GUI Operations
TongUI (Zhang et al., 2025)	Desktop + Web	[1, 9]	Outcome	✓	✗	GUI Operations
GUI-Robust (Yang et al., 2025)	Desktop + Web	–	Step	✓	✗	GUI Operations
AssistGUI (Gao et al., 2024)	Desktop	[10, 25]	Outcome	✗	✓	GUI Operations
ScreenSpot-Pro (Li et al., 2025)	Desktop	1.0	Outcome	✓	✗	Grounding
VeriGUI (Ours)	Desktop + Web	214.4	Subtask	✓	✓	GUI Operations

2 Related Works

2.1 GUI Datasets & Benchmarks

Large-scale GUI datasets and benchmarks are fundamental for training and evaluating autonomous agents in realistic human-computer interaction settings (Liu et al., 2024b; He et al., 2024; Chen et al., 2025; Zhang et al., 2025; Gao et al., 2024; Pan et al., 2024), as summarized in Tab. 1. Early web datasets and benchmarks (Shi et al., 2017; Liu et al., 2018; Yao et al., 2022) relied on simplified simulations, while recent efforts (Deng et al., 2023; Zhou et al., 2023a; Koh et al., 2024) shift toward real-world browser environments for more realistic evaluation. VisualWebBench (Liu et al., 2024b) emphasizes visual grounding and reasoning via webpage screenshots but lacks interaction capabilities. On the desktop side, OSWorld (Xie et al., 2024) and WindowsAgentArena (Bonatti et al., 2024) evaluate agents in full-featured OS environments with programmatic feedback. Other datasets and benchmarks, such as GUI-Robust (Yang et al., 2025) and WorldGUI (Zhao et al., 2025), explore robustness under varied and abnormal conditions, while ScreenSpot (Li et al., 2025) focuses on spatial element grounding rather than full task execution. However, most existing datasets rely on outcome-only verification. Several datasets (Deng et al., 2023; Yang et al., 2025; Pan et al., 2024) provide step-level annotations (*e.g.*, specific GUI actions or URL match), but require agents to strictly follow predefined action sequences. This design restricts the exploration capabilities of agents required in real-world applications. Moreover, these datasets emphasize short-term interactions, offering limited insight into agent decision-making quality over long, interdependent task sequences. VeriGUI addresses these gaps by enabling subtask-level supervision and open-ended exploration across long-horizon GUI workflows.

2.2 GUI Agents

The emergence of MLLMs like GPT-4V (Achiam et al., 2023), Gemini-Pro (Team et al., 2023), and Qwen-VL (Bai et al., 2025) has catalyzed progress in generalist GUI agents capable of interpreting screen content and executing natural language instructions. Recent agent architectures such as Show-UI (Lin et al., 2025) and UI-TARS (Qin et al., 2025) extend MLLMs with task planning modules, visual grounding techniques, and hierarchical memory (Zheng et al., 2024; Zhang et al., 2024b; Hong et al., 2024; You et al., 2024; Tan et al., 2024). These systems highlight two critical capabilities: element grounding, *i.e.*, recognizing actionable UI components from raw pixels or accessibility metadata (Li et al., 2025); and long-horizon planning, *i.e.*, decomposing high-level instructions into coherent action sequences (Zhao et al., 2025). Several works improve agent planning and reasoning capabilities via prompt engineering (Tan et al., 2024; Zheng et al., 2024; Zhou et al., 2023b; 2024), supervised fine-tuning (Lin et al., 2025; Qin et al., 2025), or reinforcement learning (Luo et al.,

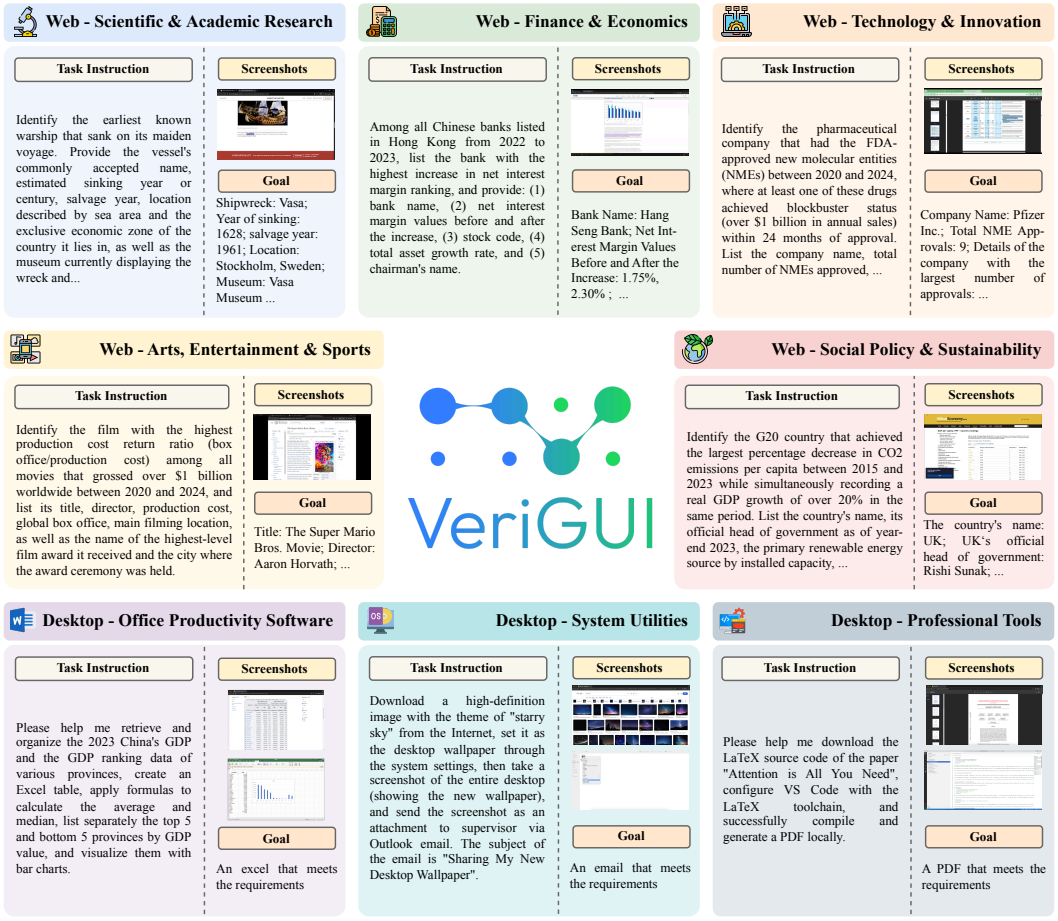


Figure 2: The VeriGUI dataset consists of various GUI tasks spanning both desktop and web.

2025; Zhou et al., 2025). In parallel to GUI agents, substantial progress has been made in deep research agents (Song et al., 2025; Jin et al., 2025; Zheng et al., 2025; Zhu et al., 2025a; Shi et al., 2025; Zhu et al., 2025b) that perform multi-hop web search and synthesis via search tool-augmented LLMs. Unlike GUI agents, these systems interact through textual APIs rather than visual interfaces. Despite promising results on existing tasks, our experiments show that current agents struggle with multi-step decision-making and error recovery in complex workflows, underscoring the need for benchmarks like VeriGUI that explicitly test long-chain generalization.

3 VeriGUI Dataset

In this section, we present the task formulation, data collection procedure, and statistical analysis of the VeriGUI dataset. As shown in Fig. 2, VeriGUI comprises two primary categories: web and desktop tasks. Specifically, the web tasks focus on deep research requiring multi-hop information retrieval and reasoning[1], whereas the desktop tasks emphasize application operation involving intricate GUI interactions and systematic state management.

3.1 Task Formulation

We formulate GUI-based tasks in VeriGUI as a Partially Observable Markov Decision Process (POMDP), defined by the tuple $\langle \mathcal{S}, \mathcal{O}, \mathcal{A}, P, O, R \rangle$, where $\mathcal{S}$ is the set of environment states,

[1]The current version of VeriGUI focuses on deep research tasks. Future versions will support a wider range of interactive tasks involving interface manipulation, such as filling out forms and setting preferences.

representing the full underlying system configuration. $\mathcal{O}$ is the observation space, and $O : \mathcal{S} \to \mathcal{O}$ is the observation function, which models the partial observations the agent receives from the environment. $\mathcal{A}$ is the action space of GUI operations. $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the state transition function, modeling the (often non-deterministic) dynamics of the GUI environment in response to actions. R is the reward function, which is defined through subtask-level verifiable goals.

For each GUI task in VeriGUI with an instruction Q, we obtain a complete task trajectory $\tau = (o_0, a_0, o_1, a_1, \ldots, o_T)$, where T denotes the number of steps in the trajectory. To capture intermediate results and provide dense supervision, we decompose τ into a sequence of K subtasks $\tau^{(1)}, \tau^{(2)}, \ldots, \tau^{(K)}$, such that $\tau = \tau^{(1)} \circ \tau^{(2)} \circ \cdots \circ \tau^{(K)}$, where $\circ$ denotes trajectory concatenation. The subtask $\tau^{(k)} = (o_{t_k}, a_{t_k}, \ldots, a_{t_{k+1}-1}, o_{t_{k+1}})$ corresponds to a contiguous segment of the full trajectory, where t_k and t_{k+1} denote the start and end timesteps. Each subtask $\tau^{(k)}$ is associated with a sub-instruction $Q^{(k)}$ and an independent subtask-level goal function $G^{(k)} : \tau^{(k)} \to \{0, 1\}$, which determines whether the agent has correctly completed the k-th goal. The corresponding subtask-level reward is defined as $R^{(k)} = G^{(k)}(\tau^{(k)}) \in \{0, 1\}$.

Observation Space. We consider two types of GUI tasks in VeriGUI: (1) web tasks, where the observation $\mathcal{O}$ includes a webpage screenshot and the HTML DOM tree. These provide both visual and structural cues for decision-making. (2) desktop tasks, where the observation $\mathcal{O}$ only consists of a desktop GUI screenshot. Compared to web environments, desktop tasks often lack structured DOM data, making perception more dependent on visual signals.

Action Space. The action space $\mathcal{A}$ defines a unified set of GUI operations applicable across both web and desktop tasks, as shown in Tab. 2. These actions cover common interaction modalities such as click, input, and key events. During execution, the agent selects one action per step from this action set. In some cases, the `result_state()` action is used by the model to output the final result. The specific mapping between the actions recorded during data collection and the GUI actions is provided in Appendix A.

Table 2: GUI actions in VeriGUI

Action Type	Description
`left_click([x, y])`	Left-click at given coordinates
`right_click([x, y])`	Right-click at given coordinates
`drag([x, y])`	Drag at given coordinates
`scroll()`	Scroll vertically
`input(text)`	Type a string input
`key_down(key)`	Press down a single key
`result_state()`	Output result statement

Goal Space. For web tasks, the goal is defined as obtaining a correct textual answer through interaction with the webpage. The agent must actively search, navigate, and reason over web content to extract the required information. A goal is considered successfully completed if the final output text matches the expected answer. For desktop tasks, goals are defined as reaching specific system states, such as enabling a configuration or launching an application. Goal completion is determined by verifying whether the current GUI or system state satisfies the intended task outcome, based on screenshots or accessibility properties. Subtask-level goal functions $G^{(k)}$ provide binary supervision for each sub-instruction and define the subtask-level reward $R^{(k)}$ used for training and evaluation.

3.2 Data Collection.

Data Source. The VeriGUI dataset is constructed from a wide range of real-world GUI environments encompassing both web and desktop platforms. (1) For web tasks, we specifically focus on deep research scenarios involving information retrieval and reasoning. Thus, we curate data from publicly accessible and authoritative sources, including official websites of government agencies, academic institutions, online encyclopedias, financial databases, and news portals. These tasks cover five primary thematic domains: scientific and academic research; finance and economics; technology and innovation; arts and entertainment; and social policy and sustainability. This categorization ensures diverse topical coverage and reflects realistic user intentions in complex information-seeking scenarios. (2) For desktop tasks, we include three representative domains of applications commonly used in professional and everyday workflows: office productivity software (*e.g.*, Word, Excel, and PowerPoint), system utilities (*e.g.*, settings configuration and file management), and professional tools (*e.g.*, VS Code and Adobe applications). These tasks capture multi-step GUI interactions that require structured reasoning, interface navigation, and sequential decision-making.

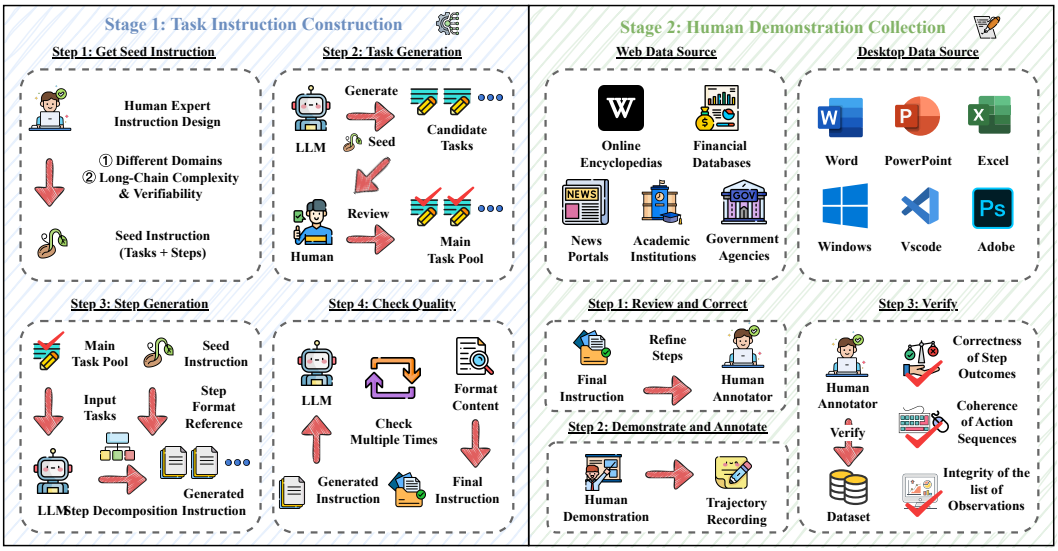


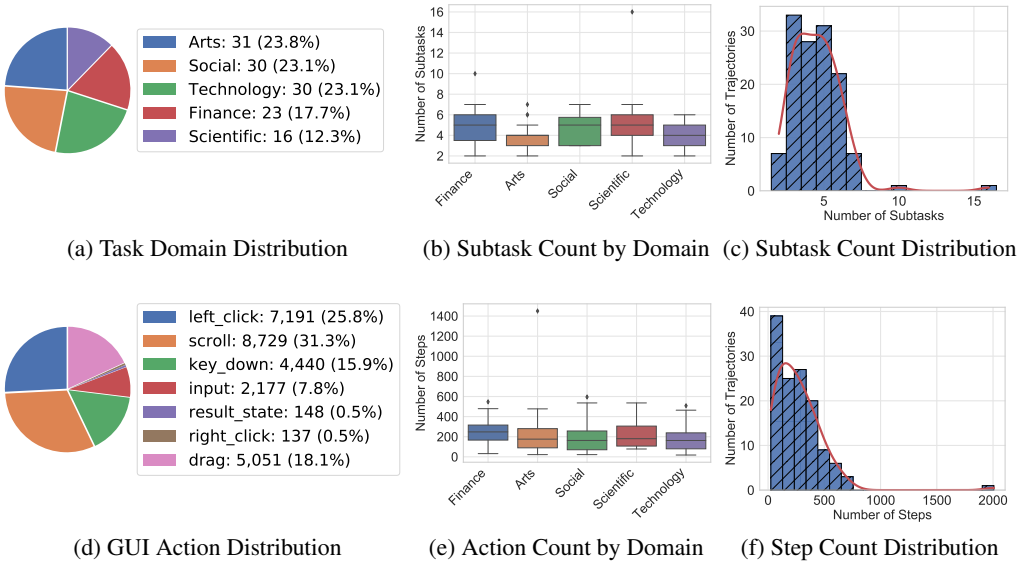
Figure 3: An overview of the proposed VeriGUI framework, consisting of two stages: task instruction construction and human demonstration collection. The framework combines LLM-based generation with human annotation to ensure realistic, high-quality GUI tasks and demonstrations.

Task Instruction Generation. To generate realistic and executable instructions, we develop a multi-stage pipeline combining language model generation with human curation, as shown in the left part of Fig. 3. Initially, a small batch of seed instructions is manually selected for each topical domain. These seed instructions, representing high-level user intents, are input to a language model to generate a large number of candidate tasks. Human annotators then review these outputs, selecting only those that are grammatically clear, semantically meaningful, and practically feasible. Once a vetted pool of main tasks is established, the language model is prompted to perform subtask decomposition to obtain complete task instructions, including detailed sub-instructions of each subtask. This process is guided by seed instructions and strict formatting constraints. After generation, each batch of instructions undergoes automated filtering, followed by a second, stricter verification phase involving multiple passes of model-based evaluation. Only those tasks that pass all verification rounds are retained. This procedure enables efficient instruction generation while maintaining the factual correctness, diversity, and task feasibility necessary for GUI datasets.

Human Demonstration Collection. Human annotators manually execute each task based on the given final instruction and record the complete trajectory demonstration, as shown in the right part of Fig. 3. Before execution, human annotators refine the subtask sequence to ensure feasibility and smooth operation, allowing adjustments as needed during interaction. Demonstrations are recorded using screen capture tools, with detailed annotations including action logs, observation logs, and subtask-level goals. To ensure high-quality supervision and accurate benchmarking, all trajectory demonstrations undergo strict quality control. This includes both automatic checks and manual review to verify the correctness of subtask outcomes, coherence of action sequences, and integrity of observations. Only demonstrations that meet all criteria are retained. This guarantees that VeriGUI provides reliable and verifiable supervision for long-horizon GUI agents.

3.3 Data Statistics

To better understand the characteristics of the VeriGUI dataset, Figure 4 and Table 3 present statistical summaries of the collected web task trajectories. These statistics provide insights into the composition and structure of GUI-based tasks collected from a variety of real-world web environments. The domain distribution of task trajectories is shown in Fig. 4a, which demonstrates that the dataset covers a wide range of domains. This ensures broad coverage and diversity across real-world tasks. Each task is decomposed into a sequence of multiple subtasks, where each subtask corresponds to a verifiable goal. Figure 4b and 4c show the distribution of the number of subtasks per trajectory, typically ranging from 4 to 8. This subtask-level structure allows agents to receive intermediate supervision, supporting more fine-grained evaluation and learning. VeriGUI further



(a) Task Domain Distribution (b) Subtask Count by Domain (c) Subtask Count Distribution

(d) GUI Action Distribution (e) Action Count by Domain (f) Step Count Distribution

Figure 4: The detailed data statistics of the proposed VeriGUI dataset based on 130 collected web task trajectories, with additional data including desktop task trajectories currently in process.

emphasizes long-chain complexity. Figure 4d illustrates the distribution of GUI action types, capturing a wide range of low-level behaviors such as clicks, scrolls, and drag operations. As shown in Fig. 4e and 4f, many tasks require executing hundreds of steps to reach completion. This highlights the need for agents to reason over long action sequences and handle extended workflows beyond simple UI manipulation. Overall, these statistics demonstrate that VeriGUI tasks are both subtask-verifiable and long-chain, offering a realistic and challenging benchmark for long-horizon planning and interaction in GUI environments.

Table 3: The overall data statistics of VeriGUI based on 130 collected web task trajectories.

Statistic	Value
Total number of tasks	130
Total number of subtasks	587
Average number of subtasks per task	4.5
Total number of steps	27,873
Average number of steps per task	214.4
Average number of steps per subtask	49.8

4 Experiments

To demonstrate the effectiveness of the proposed VeriGUI dataset in evaluating long-horizon reasoning and subtask-level verifiability in GUI tasks, we conduct experiments across a diverse set of agent frameworks and foundation models. The current experiments are based on the currently available 130 web task trajectories. Additional experiments with more comprehensive data, including ongoing desktop task trajectories, will be included in a future version of this work.

4.1 Experimental Settings

Baselines. For web tasks in VeriGUI, we evaluate different web agents: (1) deep research agents: closed-source agents with built-in search capabilities, including OpenAI Deep Research (OpenAI, 2025) and Gemini Deep Research (Google, 2025). (2) search engine agents: different foundation models combined with an open-source search tool[2] based on the model context protocol[3]. (3) browser-use agents: different foundation models using the Browser-Use framework (Müller & Žunič, 2024). (4) multi-agent system: Camel OWL (Hu et al., 2025) with OpenAI-o3. We

[2] https://github.com/searxng/searxng-docker
[3] https://modelcontextprotocol.io/

Table 4: Comparison of different agents on the VeriGUI benchmark based on 130 web tasks. SR denotes the task success rate, while CR denotes the task completion rate. **Bold** and underline mean the best and the second-best results in each column within agent type.

Method	Scientific		Finance		Technology		Arts		Social		Average	
	SR (%)	CR (%)	SR (%)	CR (%)	SR (%)	CR (%)	SR (%)	CR (%)	SR (%)	CR (%)	SR (%)	CR (%)
Deep Research Agent												
OpenAI-o3	12.5	**31.9**	0.0	18.7	**10.0**	**26.3**	**16.1**	**43.9**	3.3	**21.7**	**8.5**	**28.8**
OpenAI-o4-mini	0.0	8.1	0.0	17.0	6.7	20.7	12.9	30.6	3.3	19.0	5.4	20.5
Gemini-2.5-Flash	6.2	19.4	0.0	14.3	3.3	16.7	**16.1**	41.0	**6.7**	17.7	6.9	22.6
Gemini-2.5-Pro	**18.8**	**31.9**	0.0	**22.2**	**10.0**	23.7	**16.1**	41.6	0.0	21.0	**8.5**	28.1
Search Engine Agent												
GPT-4o	0.0	3.1	0.0	3.0	3.3	10.3	0.0	3.9	0.0	4.3	0.8	5.2
GPT-4.1	0.0	**13.1**	0.0	**14.8**	3.3	14.3	9.7	23.5	0.0	8.0	3.1	15.0
OpenAI-o3	0.0	5.0	0.0	13.5	10.0	19.0	**12.9**	**35.2**	0.0	**11.0**	**5.4**	**18.3**
Gemini-2.5-Flash	0.0	5.0	0.0	7.4	0.0	8.3	6.5	28.1	0.0	6.7	1.5	12.1
Gemini-2.5-Pro	0.0	4.4	0.0	8.7	3.3	12.0	**12.9**	28.1	0.0	7.7	3.8	13.3
Claude-3.7-Sonnet	0.0	8.1	0.0	10.9	**13.3**	**23.7**	9.7	30.0	0.0	8.0	**5.4**	17.4
Claude-4.0-Sonnet	0.0	11.9	0.0	11.3	6.7	13.7	**12.9**	21.9	0.0	**11.0**	4.6	14.4
Deepseek-Chat	0.0	4.4	0.0	2.2	3.3	10.7	**12.9**	24.8	0.0	4.7	3.8	10.4
Browser-Use Agent												
GPT-4o	0.0	1.9	0.0	1.7	3.3	8.3	3.2	13.5	0.0	5.7	1.5	7.0
GPT-4.1	0.0	3.8	0.0	7.0	3.3	9.0	16.1	29.7	0.0	9.7	4.6	13.1
OpenAI-o3	**6.2**	**20.6**	0.0	**11.3**	0.0	**18.7**	16.1	33.5	0.0	**12.3**	4.6	**19.7**
Gemini-2.5-Flash	0.0	1.9	0.0	6.1	0.0	2.0	0.0	19.7	0.0	7.3	0.0	8.2
Gemini-2.5-Pro	**6.2**	10.6	0.0	6.1	**6.7**	9.7	12.9	36.1	0.0	10.0	5.4	15.5
Claude-3.7-Sonnet	0.0	7.5	0.0	9.6	0.0	15.3	16.1	36.8	0.0	10.3	3.8	17.3
Claude-4.0-Sonnet	**6.2**	13.8	0.0	6.5	0.0	11.3	**19.4**	**45.8**	**3.3**	9.3	**6.2**	18.5
Qwen-VL-Max	0.0	2.5	0.0	0.9	0.0	3.0	6.5	11.6	0.0	4.3	1.5	4.9
Multi-Agent System												
OWL with OpenAI-o3	6.2	18.8	0.0	6.5	3.3	11.3	16.1	32.3	6.7	16.3	6.9	17.5

use the following foundation models across settings: OpenAI-o3, OpenAI-o4-mini, GPT-4.1, GPT-4o (Hurst et al., 2024), Gemini-2.5-Pro, Gemini-2.5-Flash (Team et al., 2023), Claude-3.7-Sonnet, Claude-4.0-Sonnet (Anthropic, 2024), DeepSeek-Chat (Liu et al., 2024a), and Qwen-VL-Max (Bai et al., 2025). Note that except for the closed-source deep research agents, the remaining web agents follow two different interaction paradigms: (1) search engine agents use a search tool for retrieval without interacting with webpages and accept only text as input. (2) browser-use agents and OWL agents adopt web element operations as their action space and accept both visual and text inputs.

Evaluation Metrics. To comprehensively evaluate agent performance, we consider three complementary metrics[4]. (1) The *task Success Rate (SR)* measures whether the agent achieves the overall task goal. (2) The *task Completion Rate (CR)* measures the extent to which the agent achieves the overall task goal. Since our tasks often involve multiple subtasks, CR estimates the completion level by calculating the proportion of correct elements in the output. For example, if the expected result contains ten keywords and the agent outputs one correctly, the CR is 10%. We introduce this metric because the tasks are highly challenging, and using only SR makes it difficult to distinguish the performance differences between agents. (3) The *Action Efficiency (AE)* quantifies the planning effectiveness of agents by measuring the number of steps required to arrive at the final answer. Note that AE is only defined for tasks that are successfully completed. Moreover, AE is not directly comparable across different interaction paradigms due to their inherently distinct action spaces. For both the SR and the CR, we report the LLM-as-a-Judge score (Gu et al., 2024). Specifically, we utilize GPT-4.1 as the judge to semantically evaluate the correctness of the agents' final answers. Without further clarification, each experiment is conducted once to obtain the final results. Detailed prompts are provided in Appendix B.

[4]We further introduce a metric that treats a subtask as a valid starting point, allowing agent evaluation at different task stages. Specifically, the *task Success Rate under k-subtask oracle (SR@k)* indicates whether the agent can achieve the overall task goal when provided with the goal outcomes of the first k subtasks. Experiments on this will be presented in a future version.

4.2 MAIN RESULTS

Table 4 summarizes the performance of all evaluated agents on the VeriGUI benchmark across five web domains, measuring both task success rate and completion rate. Notably, across all agent types and foundation models, no configuration achieves an average success rate above 10% or a completion rate above 30%. This consistently low performance highlights the challenging nature of the VeriGUI tasks, which require long-horizon planning, multi-step reasoning, and complex decision-making under diverse web scenarios. We analyze the results from three perspectives: foundation model capability, interaction paradigm, and domain-specific behavior.

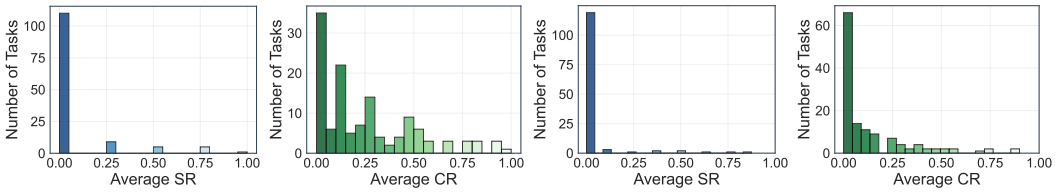
Foundation Model Comparison. We observe notable differences in agent performance across foundation models. Within the deep research agent setting, OpenAI-o3 and Gemini-2.5-Pro achieve the highest average SRs at 8.5%, with CRs of 28.8% and 28.1%, respectively. These results suggest that both models possess relatively stronger reasoning capabilities and better generalization across tasks. In contrast, OpenAI-o4-mini performs worst in this setting, indicating limitations in handling complex web tasks despite being a reasoning model. In the search engine and browser-use settings, where most models are shared, we observe similar model-wise trends. OpenAI-o3, Claude-3.7-Sonnet, and Claude-4.0-Sonnet demonstrate stronger completion rates across both settings. GPT-4o shows consistently low SRs (0.8–1.5%) and CRs (5.2–7.0%) across both settings, indicating limitations in handling complex multi-step tasks. Although GPT-4.1 performs slightly better, it still lags behind Claude and Gemini. Besides, Deepseek-Chat and Qwen-VL-Max also show weaker performance. These results suggest that foundation model differences continue to play a key role in determining agent effectiveness.

Impact of Interaction Paradigms. The design of the interaction paradigm has a substantial impact on agent performance. Agents using the search engine paradigm achieve the weakest results across both SR and CR metrics. Most models under this setting have average SRs between 0.8–5.4% and CRs below 18.3%. This is likely due to their reliance on passive text-based retrieval without the ability to interact directly with web page structures. In comparison, agents using the browser-use paradigm generally obtain slightly higher scores. While the improvements in SR are often modest, the average CR is higher for several models. For instance, Claude-4.0-Sonnet improves from 14.4% CR in the search engine setting to 18.5% in the browser setting, and Gemini-2.5-Pro improves from 13.3% to 15.5%. These gains suggest that having access to page-level structure and the ability to simulate user actions can provide meaningful advantages, especially for tasks involving dynamic interfaces or multiple steps. The deep research agent setting achieves the highest completion rates overall, with top models reaching over 28% average CR. Although their interaction mechanisms are less transparent, the results suggest that strong built-in retrieval, summarization, or planning capabilities may contribute to their relative success. The multi-agent system also performs competitively, achieving an SR of 6.9% with OWL and OpenAI-o3, indicating that orchestrated agent collaboration can be beneficial in certain task types.

Performance Across Domains. We also analyze performance across the five domains in the VeriGUI to explore how content type influences agent effectiveness. Tasks in *arts and entertainment* generally achieved the highest success and completion rates, likely due to more structured and predictable data formats such as lists or summaries. For example, the browser-use agent with Claude-4.0-Sonnet reaches 19.4% SR and 45.8% CR in this domain. In contrast, domains like *finance and economics* and *social policy and sustainability* proved more challenging, often requiring the agent to extract fragmented, abstract information from less standardized content. Most models show SRs near 0% and CRs below 20% in these domains. The *scientific and academic research* and *technology and innovation* domains showed intermediate difficulty, frequently involving dense technical descriptions or multi-attribute reasoning. These trends suggest that the complexity of information presentation is the key factor influencing agent success in web-based GUI tasks.

4.3 ANALYSIS

Analysis of Task Difficulty. To better understand the intrinsic difficulty of tasks in the VeriGUI-Web benchmark, we conduct a fine-grained statistical analysis of SR and CR distributions across all tasks, comparing results from different agent frameworks. The distribution curves in Fig. 5 reveal



(a) Deep Research SR (b) Deep Research CR (c) Search Engine SR (d) Search Engine CR

(e) Browser-Use SR (f) Browser-Use CR (g) Multi-Agent SR (h) Multi-Agent CR

Figure 5: Distribution of task success rate (SR) and completion rate (CR) across 130 web tasks.

that for both agent types, the majority of tasks yield low SR and CR values, with a long tail of near-zero success, underscoring the challenge posed by VeriGUI's multi-step reasoning requirements.

To systematically categorize task difficulty, we define five levels based on the average SR and CR across all models and agents: (1) Level 1 includes tasks with SR above 0%, indicating they are relatively tractable for current agents. (2) Level 2 includes tasks with zero SR but CR above 20%. (3) Level 3 includes tasks with zero SR but CR between 5% and 20%. (4) Level 4 includes tasks with zero SR but CR between 0% and 5%. (5) Level 5 includes tasks where both SR and CR are zero, indicating no model was able to make progress. The results in Fig. 6 show that the majority of VeriGUI tasks fall into Level 2–5 with zero SR, highlighting the prevalence of high-complexity, partially achievable tasks. Only a small fraction of tasks fall into Level 1, indicating that few tasks are straightforward for current agents. This categorization provides a practical framework for future benchmarking and curriculum design in GUI agent training.

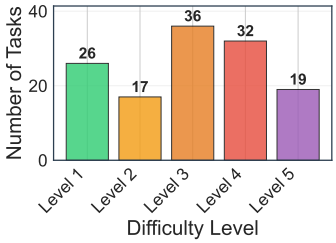


Figure 6: Task Difficulty Level.

Analysis of Action Efficiency. The analysis of action efficiency reveals clear differences in the planning capabilities of browser-use agents powered by various foundation models. Notably, in the browser-use setting, agents operate over web elements as their action space rather than GUI operations, making their action counts not directly comparable to those in human demonstrations. As shown in Tab. 5, models such as GPT-4.1 and Claude-3.7-Sonnet generally required more actions, suggesting a more exploratory or cautious execution style. In contrast, models like OpenAI-4o and Claude-4.0-Sonnet completed tasks with fewer steps, indicating more direct strategies. However, lower action counts did not always align with better outcomes, as some models exhibited brittle reasoning despite efficient execution. Conversely, higher action counts sometimes reflected more thorough exploration, particularly in tasks with complex or ambiguous goals. These results suggest that while action efficiency provides insight into planning behavior, it must be considered alongside success rate to fully assess agent performance.

Table 5: Comparison of the average action efficiency for browser-use agents with different foundation Models on the VeriGUI benchmark.

Method	**Average AE**
Browser-Use Agent	
OpenAI-o3	29.7
GPT-4o	22.8
GPT-4.1	36.0
Gemini-2.5-Pro	41.2
Gemini-2.5-Flash	65.7
Claude-3.7-Sonnet	35.7
Claude-4.0-Sonnet	24.7
Qwen-VL-Max	29.8

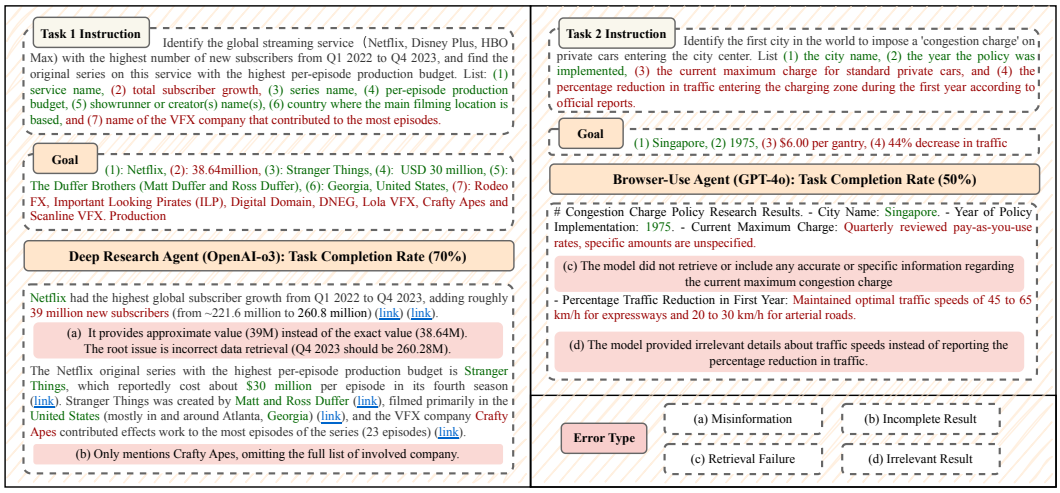


Figure 7: Case studies of agent performance on two web tasks in VeriGUI.

4.4 Case Studies

To better understand the behaviors and limitations of different agent types in long-horizon GUI reasoning tasks, we present two representative cases from the VeriGUI benchmark. These examples illustrate retrieval fidelity, multi-step reasoning quality, and typical failure modes across four defined error types: Misinformation, Incomplete Result, Retrieval Failure, and Irrelevant Result.

For Task 1 in the left part of Fig. 7, the agent must identify the global streaming service with the highest subscriber growth and provide detailed metadata about its highest-budget original series. The Deep Research Agent (OpenAI-o3) achieves a relatively high Completion Rate, correctly identifying Netflix, Stranger Things, and most relevant metadata. However, it exhibits two key errors. First, it commits misinformation by reporting an approximate subscriber growth of 39 million instead of the exact 38.64 million, due to being misled by media reports and mistakenly recording Q4 2023 as 260.8 million instead of the official 260.28 million. Second, it demonstrates an incomplete result by mentioning only one VFX company, while omitting six others that contributed significantly.

For Task 2 in the right part of Fig. 7, the agent is asked to identify the first city to implement a congestion charge and extract key policy details. The Browser-Use Agent (GPT-4o) correctly identifies Singapore and the implementation year 1975, but fails in other aspects. It encounters a retrieval failure by not providing any specific value for the congestion charge, instead returning vague descriptions. Additionally, it provides an irrelevant result by discussing average traffic speeds rather than reporting the required percentage reduction in traffic during the first year. These issues suggest that although browser-based agents can navigate webpages, they still struggle with precise data extraction and generating structured, goal-oriented output, leading to a lower completion rate.

Beyond individual examples, our experiments reveal several systemic limitations. First, many chat-based agents demonstrate shallow search behavior: they invoke tools only a few times before prematurely terminating the output, even when the task clearly requires deeper investigation. This limits their ability to perform comprehensive, multi-hop retrieval in complex GUI environments. Second, browser agents often formulate web queries using full natural language sentences instead of distilled keywords. While sentence-level inputs may appear more natural, they frequently lead to suboptimal search results, reducing the likelihood of retrieving exact information needed for task completion.

5 Discussion

The current experimental results in VeriGUI are based on a limited subset of 130 web tasks, most of which focus on information-seeking scenarios. Interestingly, we observe that deep research agents generally outperform browser-use agents in this setting. This raises an important question: should we prioritize the development of deep research agents, or does the GUI agent paradigm still hold

promise for broader and more powerful generalist capabilities? We believe the latter remains highly compelling, and this observation should be interpreted from several perspectives.

GUI Agents Excel in Interactive Tasks. The nature of the tasks strongly influences performance. Most of the current web tasks in VeriGUI emphasize multi-hop information retrieval and factual synthesis, which align closely with the strengths of deep research agents. However, for many practical tasks involving interface manipulation, such as uploading files and logging into accounts, deep research agents are fundamentally limited. These agents lack the ability to interact with the visual layout of interfaces, which is essential for completing such tasks. In contrast, GUI agents are built to operate over both the visual and structural components of the environment, enabling them to tackle interactive workflows that go beyond passive information extraction. Thus, future versions of VeriGUI will include a broader set of web tasks that emphasize GUI interaction.

GUI Agent Performance is Underestimated. Most existing browser-based GUI agents rely on general-purpose multimodal models and relatively basic execution frameworks. They have not yet benefited from the same degree of domain-specific optimization or tool integration that powers deep research systems. As the field progresses, we expect that advances in environment modeling, long-horizon planning, multimodal understanding, and training with fine-grained subtask supervision as provided in VeriGUI will significantly improve the reasoning, robustness, and decision-making capabilities of GUI agents. The performance gap we see today should not be seen as a fundamental limitation, but rather as a reflection of the early stages of a promising technology.

GUI Agents may Offer a Path Toward Generalist Agents. One of the most exciting prospects for GUI agents is their potential to serve as a foundational tool in the development of more generalist AI systems. While deep research agents are currently focused on web-based tasks, GUI agents have the inherent ability to generalize across multiple computing environments, including web and desktop platforms. Their ability to interact with graphical interfaces makes them versatile, capable of performing tasks such as browsing, document editing, system configuration, and data entry, all without the need for domain-specific rules or pipelines. This extensibility and flexibility provide a promising path towards building generalist models that can seamlessly navigate and execute tasks across diverse digital environments. By offering a unified approach to task execution, GUI agents may become a critical enabler for the development of truly general-purpose interactive agents.

It is important to note that the current evaluation only reflects a portion of what VeriGUI aims to capture. We are actively expanding the dataset to include more web tasks with interactive requirements, as well as a significant number of desktop tasks involving complex software operations. Future experiments on this expanded data will enable a more balanced and complete understanding of GUI agent capabilities across task types and environments.

6 Conclusion

In this work, we introduce VeriGUI, a large-scale, human-annotated dataset designed to address the growing need for verifiable, long-horizon benchmarks in GUI agent research. Unlike prior datasets that focus on short-term interactions and outcome-only validation, VeriGUI emphasizes *long-chain complexity* and *subtask-level verifiability*, supporting the development and evaluation of agent capabilities in real-world GUI workflows. Our comprehensive experiments across a range of leading agent models highlight persistent challenges in long-horizon task decomposition and execution, underscoring the importance of datasets like VeriGUI in pushing the frontier of generalist agent intelligence. We have open-sourced the dataset and will continue to update it. We hope VeriGUI serves as a valuable resource for the community, fostering further research into GUI agents.

7 CONTRIBUTORS

Project Leaders

- Shunyu Liu, Nanyang Technological University
- Minghao Liu, 2077AI, M-A-P

Core Contributors

- Huichi Zhou, 2077AI
- Zhenyu Cui, Zhejiang University
- Yang Zhou, Zhejiang University
- Yuhao Zhou, Shanghai AI Lab

Contributors

- Wendong Fan, Camel AI
- Ge Zhang, M-A-P
- Jiajun Shi, M-A-P
- Weihao Xuan, The University of Tokyo
- Jiaxing Huang, Nanyang Technological University
- Shuang Luo, Zhejiang University
- Fang Wu, Stanford University
- Heli Qi, Waseda University
- Qingcheng Zeng, Northwestern University
- Ziqi Ren, 2077AI, Zhejiang University
- Jialiang Gao, 2077AI, Abaka AI
- Jindi Lv, Sichuan University
- Junjie Wang, Tsinghua University, 2077AI
- Aosong Feng, Yale University
- Heng Zhou, Shanghai AI Lab

Advisors

- Wangchunshu Zhou, OPPO
- Zhenfei Yin, Shanghai AI Lab
- Wenlong Zhang, Shanghai AI Lab
- Guohao Li, Camel AI
- Wenhao Yu, Tencent AI Lab
- Irene Li, The University of Tokyo
- Lei Ma, The University of Tokyo
- Lei Bai, Shanghai AI Lab
- Qunshu Lin, Abaka AI, Zhejiang University

Corresponding Authors

- Mingli Song, Zhejiang University
- Dacheng Tao, Nanyang Technological University

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Saaket Agashe, Jiuzhou Han, Shuyu Gan, Jiachen Yang, Ang Li, and Xin Eric Wang. Agent s: An open agentic framework that uses computers like a human. *arXiv preprint arXiv:2410.08164*, 2024.

Anthropic. The claude 3 model family: Opus, sonnet, haiku, 2024. URL https://www.anthropic.com.

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

Rogerio Bonatti, Dan Zhao, Francesco Bonacci, Dillon Dupont, Sara Abdali, Yinheng Li, Yadong Lu, Justin Wagle, Kazuhito Koishida, Arthur Bucker, et al. Windows agent arena: Evaluating multi-modal os agents at scale. *arXiv preprint arXiv:2409.08264*, 2024.

Dongping Chen, Yue Huang, Siyuan Wu, Jingyu Tang, Huichi Zhou, Qihui Zhang, Zhigang He, Yilin Bai, Chujie Gao, Liuyi Chen, et al. Gui-world: A video benchmark and dataset for multimodal gui-oriented understanding. In *ICLR*, 2025.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. In *ACL*, pp. 9313–9332, 2024.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *NeurIPS*, volume 36, pp. 28091–28114, 2023.

Difei Gao, Lei Ji, Zechen Bai, Mingyu Ouyang, Peiran Li, Dongxing Mao, Qinchen Wu, Weichen Zhang, Peiyi Wang, Xiangwu Guo, et al. Assistgui: Task-oriented pc graphical user interface automation. In *CVPR*, pp. 13289–13298, 2024.

Google. Gemini deep research, 2025. URL https://gemini.google/overview/deep-research.

Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and Yu Su. Navigating the digital world as humans do: Universal visual grounding for gui agents. *arXiv preprint arXiv:2410.05243*, 2024.

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

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 *ACL*, 2024.

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 *CVPR*, pp. 14281–14290, 2024.

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

Xueyu Hu, Tao Xiong, Biao Yi, Zishu Wei, Ruixuan Xiao, Yurun Chen, Jiasheng Ye, Meiling Tao, Xiangxin Zhou, Ziyu Zhao, et al. Os agents: A survey on mllm-based agents for computer, phone and browser use. *OpenReview*, 2024.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

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

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 *ACL*, 2024.

Kaixin Li, Ziyang Meng, Hongzhan Lin, Ziyang Luo, Yuchen Tian, Jing Ma, Zhiyong Huang, and Tat-Seng Chua. Screenspot-pro: Gui grounding for professional high-resolution computer use. *arXiv preprint arXiv:2504.07981*, 2025.

Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Stan Weixian Lei, Lijuan Wang, and Mike Zheng Shou. Showui: One vision-language-action model for gui visual agent. In *CVPR*, pp. 19498–19508, 2025.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024a.

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.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, volume 36, pp. 34892–34916, 2023.

Junpeng Liu, Yifan Song, Bill Yuchen Lin, Wai Lam, Graham Neubig, Yuanzhi Li, and Xiang Yue. Visualwebbench: How far have multimodal llms evolved in web page understanding and grounding? *arXiv preprint arXiv:2404.05955*, 2024b.

Xing Han Lù, Zdeněk Kasner, and Siva Reddy. Weblinx: Real-world website navigation with multi-turn dialogue. In *ICML*, 2024.

Run Luo, Lu Wang, Wanwei He, and Xiaobo Xia. Gui-r1: A generalist r1-style vision-language action model for gui agents. *arXiv preprint arXiv:2504.10458*, 2025.

Magnus Müller and Gregor Žunič. Browser use: Enable ai to control your browser, 2024. URL https://github.com/browser-use/browser-use.

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.

Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, et al. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. *arXiv preprint arXiv:2503.23350*, 2025.

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

Yichen Pan, Dehan Kong, Sida Zhou, Cheng Cui, Yifei Leng, Bing Jiang, Hangyu Liu, Yanyi Shang, Shuyan Zhou, Tongshuang Wu, et al. Webcanvas: Benchmarking web agents in online environments. *arXiv preprint arXiv:2406.12373*, 2024.

Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025.

Dingfeng Shi, Jingyi Cao, Qianben Chen, Weichen Sun, Weizhen Li, Hongxuan Lu, Fangchen Dong, Tianrui Qin, King Zhu, Minghao Liu, Jian Yang, Ge Zhang, Jiaheng Liu, Changwang Zhang, Jun Wang, Yuchen Eleanor Jiang, and Wangchunshu Zhou. Taskcraft: Automated generation of agentic tasks. *arXiv preprint arXiv:2506.10055*, 2025.

Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and Percy Liang. World of bits: An open-domain platform for web-based agents. In *ICML*, pp. 3135–3144, 2017.

Huatong Song, Jinhao Jiang, Wenqing Tian, Zhipeng Chen, Yuhuan Wu, Jiahao Zhao, Yingqian Min, Wayne Xin Zhao, Lei Fang, and Ji-Rong Wen. R1-searcher++: Incentivizing the dynamic knowledge acquisition of llms via reinforcement learning. *arXiv preprint arXiv:2505.17005*, 2025.

Weihao Tan, Wentao Zhang, Xinrun Xu, Haochong Xia, Ziluo Ding, Boyu Li, Bohan Zhou, Junpeng Yue, Jiechuan Jiang, Yewen Li, et al. Cradle: Empowering foundation agents towards general computer control. *arXiv preprint arXiv:2403.03186*, 2024.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Shuai Wang, Weiwen Liu, Jingxuan Chen, Yuqi Zhou, Weinan Gan, Xingshan Zeng, Yuhan Che, Shuai Yu, Xinlong Hao, Kun Shao, et al. Gui agents with foundation models: A comprehensive survey. *arXiv preprint arXiv:2411.04890*, 2024.

Jialong Wu, Wenbiao Yin, Yong Jiang, Zhenglin Wang, Zekun Xi, Runnan Fang, Linhai Zhang, Yulan He, Deyu Zhou, Pengjun Xie, et al. Webwalker: Benchmarking llms in web traversal. *arXiv preprint arXiv:2501.07572*, 2025.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, et al. Os-atlas: A foundation action model for generalist gui agents. *arXiv preprint arXiv:2410.23218*, 2024.

Tianbao Xie, Fan Zhou, Zhoujun Cheng, Peng Shi, Luoxuan Weng, Yitao Liu, Toh Jing Hua, Junning Zhao, Qian Liu, Che Liu, et al. Openagents: An open platform for language agents in the wild. *arXiv preprint arXiv:2310.10634*, 2023.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh J Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, et al. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *NeurIPS*, volume 37, pp. 52040–52094, 2024.

Jingqi Yang, Zhilong Song, Jiawei Chen, Mingli Song, Sheng Zhou, Xiaogang Ouyang, Chun Chen, Can Wang, et al. Gui-robust: A comprehensive dataset for testing gui agent robustness in real-world anomalies. *arXiv preprint arXiv:2506.14477*, 2025.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *NeurIPS*, volume 35, pp. 20744–20757, 2022.

Keen You, Haotian Zhang, Eldon Schoop, Floris Weers, Amanda Swearngin, Jeffrey Nichols, Yinfei Yang, and Zhe Gan. Ferret-ui: Grounded mobile ui understanding with multimodal llms. In *ECCV*, pp. 240–255, 2024.

Bofei Zhang, Zirui Shang, Zhi Gao, Wang Zhang, Rui Xie, Xiaojian Ma, Tao Yuan, Xinxiao Wu, Song-Chun Zhu, and Qing Li. Tongui: Building generalized gui agents by learning from multimodal web tutorials. *arXiv preprint arXiv:2504.12679*, 2025.

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

Chaoyun Zhang, Liqun Li, Shilin He, Xu Zhang, Bo Qiao, Si Qin, Minghua Ma, Yu Kang, Qingwei Lin, Saravan Rajmohan, et al. Ufo: A ui-focused agent for windows os interaction. *arXiv preprint arXiv:2402.07939*, 2024b.

Duzhen Zhang, Yahan Yu, Jiahua Dong, Chenxing Li, Dan Su, Chenhui Chu, and Dong Yu. Mm-llms: Recent advances in multimodal large language models. *arXiv preprint arXiv:2401.13601*, 2024c.

Shaoqing Zhang, Zhuosheng Zhang, Kehai Chen, Xinbei Ma, Muyun Yang, Tiejun Zhao, and Min Zhang. Dynamic planning for llm-based graphical user interface automation. In *EMNLP Findings*, pp. 1304–1320, 2024d.

Henry Hengyuan Zhao, Difei Gao, and Mike Zheng Shou. Worldgui: An interactive benchmark for desktop gui automation from any starting point. *arXiv preprint arXiv:2502.08047*, 2025.

Longtao Zheng, Zhiyuan Huang, Zhenghai Xue, Xinrun Wang, Bo An, and Shuicheng Yan. Agentstudio: A toolkit for building general virtual agents. *arXiv preprint arXiv:2403.17918*, 2024.

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

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

Wangchunshu Zhou, Yuchen Eleanor Jiang, Long Li, Jialong Wu, Tiannan Wang, Shi Qiu, Jintian Zhang, Jing Chen, Ruipu Wu, Shuai Wang, Shiding Zhu, Jiyu Chen, Wentao Zhang, Xiangru Tang, Ningyu Zhang, Huajun Chen, Peng Cui, and Mrinmaya Sachan. Agents: An open-source framework for autonomous language agents. *arXiv preprint arXiv:2309.07870*, 2023b.

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

Yuqi Zhou, Sunhao Dai, Shuai Wang, Kaiwen Zhou, Qinqlin Jia, et al. Gui-g1: Understanding r1-zero-like training for visual grounding in gui agents. *arXiv preprint arXiv:2505.15810*, 2025.

He Zhu, Tianrui Qin, King Zhu, Heyuan Huang, Yeyi Guan, Jinxiang Xia, Yi Yao, Hanhao Li, Ningning Wang, Pai Liu, Tianhao Peng, Xin Gui, Xiaowan Li, Yuhui Liu, Yuchen Eleanor Jiang, Jun Wang, Changwang Zhang, Xiangru Tang, Ge Zhang, Jian Yang, Minghao Liu, Xitong Gao, Jiaheng Liu, and Wangchunshu Zhou. Oagents: An empirical study of building effective agents. *arXiv preprint arXiv:2506.15741*, 2025a.

King Zhu, Hanhao Li, Siwei Wu, Tianshun Xing, Dehua Ma, Xiangru Tang, Minghao Liu, Jian Yang, Jiaheng Liu, Yuchen Eleanor Jiang, Changwang Zhang, Chenghua Lin, Jun Wang, Ge Zhang, and Wangchunshu Zhou. Scaling test-time compute for llm agents. *arXiv preprint arXiv:2506.12928*, 2025b.

Appendix

TABLE OF CONTENTS

A Action Space

A.1 Action Space for Web Task

We develop a screen capture tool to support human annotators in collecting detailed task trajectories. Each recorded trajectory logs all mouse and keyboard events, which can be systematically mapped to the predefined GUI action space. The mapping is as follows:

- Scroll wheel events (`WHEEL`) are mapped to the `scroll` action.
- Key press events (`KEY_DOWN`) are mapped to the `key_down` action.
- Text input events (`INPUT`) are mapped to the `input` action.
- Text output events (`RESULT_STATE`) are mapped to the `result_state` action.
- Right-click context menu events (`CONTEXT_MENU`) are mapped to the `right_click` action.
- Tab switching events (`TAB_CHANGE`) are interpreted to the `left_click` action at the corresponding coordinates.
- Mouse drag actions (`MOUSE_DRAG`) are mapped to the `drag` action.
- If a `MOUSE_DOWN` event is not followed by a `MOUSE_DRAG` event, it is interpreted as the `left_click` action.
- Additionally, `MOUSE_UP` events are recorded to help determine the end of drag actions or validate click completions, although they are not directly mapped to any action in the defined space.

This mapping ensures consistency between the raw recorded interactions and the unified action space $\mathcal{A}$, enabling accurate interpretation and reproduction of user behaviors by the model during both training and inference.

B Experimental Settings

B.1 Agent Prompt for Web Task

The agent prompt for different agent in web tasks is shown below:

Deep Research Agent Prompt

{question}

Search Engine Agent Prompt

You are the EXECUTOR agent. You will receive one task description at a time. Your role is to complete the task efficiently, using available tools via function calls when necessary.

Guidelines:
- Always think step by step before responding.
- Provide concise answers.
- If a tool is needed, respond only with the function call — no extra text.
- When the task is complete, respond with: FINAL ANSWER: [your answer here]

Browser-Use Agent Prompt

We follow the official agent prompt from Browser-Use (Müller & Žunič, 2024).

OWL Agent Prompt

You are a helpful assistant that can search the web, extract webpage content, simulate browser actions, and provide relevant information to solve the given task.

You are now working in 'working_dir'. All your work related to local operations should be done in that directory.

Mandatory Instructions
1. **Take Detailed Notes**: You MUST use the 'append_note' tool to record your findings. Ensure notes are detailed, well-organized, and include source URLs. Do not overwrite notes unless summarizing; append new information. Your notes are crucial for the Document Agent.

Web Search Workflow
1. **Initial Search**: Start with a search engine like 'search_google' or 'search_bing' to get a list of relevant URLs for your research if available.
2. **Browser-Based Exploration**: Use the rich browser toolset to investigate websites.
- **Navigation**: Use 'visit_page' to open a URL. Navigate with 'click', 'back', and 'forward'. Manage multiple pages with 'switch_tab'.
- **Analysis**: Use 'get_som_screenshot' to understand the page layout and identify interactive elements. Since this is a heavy operation, only use it when visual analysis is necessary.
- **Interaction**: Use 'type' to fill out forms and 'enter' to submit.
3. **Detailed Content Extraction**: Prioritize using the scraping tools from 'Crawl4AIToolkit' for in-depth information gathering from awebpage.

Guidelines and Best Practices
- **URL Integrity**: You MUST only use URLs from trusted sources (e.g., search engine results or links on visited pages). NEVER invent or guess URLs.
- **Thoroughness**: If a search query is complex, break it down. If a snippet is unhelpful but the URL seems authoritative, visit the page. Check subpages for more information.
- **Persistence**: If one method fails, try another. Combine search, scraper, and browser tools for comprehensive information gathering.
- **Collaboration**: Communicate with other agents using 'send_message' when you need help. Use 'list_available_agents' to see who is available.
- **Clarity**: In your response, you should mention the URLs you have visited and processed.

B.2 LLM-AS-A-JUDGE PROMPT FOR WEB TASK

For web tasks, the goal is defined as obtaining a correct textual answer through multi-turn information retrieval and reasoning. Thus, we use GPT-4.1 as a judge to semantically evaluate the correctness of agents' final answers based on the question, ground truth, and model response, and report the LLM-as-a-Judge score. The detailed evaluation prompt is provided as follows:

LLM-as-a-Judge Prompt for Web Task

You are a strict evaluator assessing answer correctness. You must score the model's prediction on a scale from 0 to 10, where 0 represents an entirely incorrect answer and 10 indicates a highly correct answer.

Input
Question:
```
{question}
```
Ground Truth Answer:

```
{answer}
```
Model Prediction:
```
{pred}
```

Evaluation Rules
- The model prediction may contain the reasoning process, you should spot the final answer from it.
- Assign a high score if the prediction matches the answer semantically, considering variations in format.
- Deduct points for partially correct answers or those with incorrect additional information.
- Ignore minor differences in formatting, capitalization, or spacing since the model may explain in a different way.
- Treat numerical answers as correct if they match within reasonable precision
- For questions requiring units, both value and unit must be correct

Scoring Guide
Provide a single integer from 0 to 10 to reflect your judgment of the answer's correctness.

Strict Output format example
4

C TASK EXAMPLE

C.1 WEB TASK EXAMPLE

The web tasks focus on deep research requiring multi-turn information retrieval and reasoning. In VeriGUI, these tasks span five key thematic domains: scientific and academic research; finance and economics; technology and innovation; arts and entertainment; and social policy and sustainability. Below are some examples of web tasks.

Web Task Example - Scientific and Academic Research

Task Instruction
Identify the earliest known warship that sank on its maiden voyage. Provide the vessel's commonly accepted name, estimated sinking year or century, salvage year, location described by sea area and the exclusive economic zone of the country it lies in, as well as the museum currently displaying the wreck and the official name of any anchor-related artifacts from it in the museum's collection.

Task Goal
Shipwreck: Vasa
Year of sinking: 1628
Salvage year: 1961
Location: Stockholm, Sweden
Museum: Vasa Museum
Artifact: Ankarstock

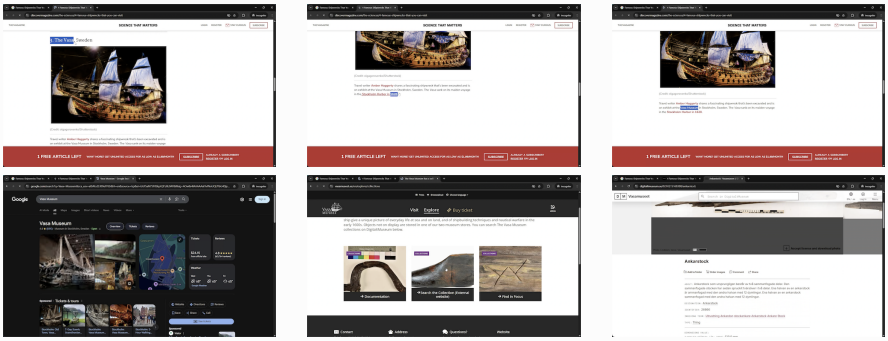


Figure 8: Human demonstration screenshots for the scientific and academic research web task example.

Subtask 1 Instruction
Compile a list of major shipwreck discoveries where the primary search and identification technology used was multibeam sonar.

Subtask 1 Goal

Name	Year
USS Kittiwake	2011
C-50 Naufragio Vicente Palacio Riva Ship	2000
The Vasa	1961
The Lusitania	1915

Subtask 2 Instruction
For each shipwreck on the list, find its estimated sinking date (year or century). Identify the wreck with the earliest sinking date.

Subtask 2 Goal

Name	Year
USS Kittiwake	1994
C-50 Naufragio Vicente Palacio Riva Ship	2000
The Vasa	1628 (oldest)
The Lusitania	1906

Subtask 3 Instruction
Confirm the full name of the organization, or institution responsible for the discovery of the Vasa.

Subtask 3 Goal
Organization: Vasa Museum

Subtask 4 Instruction
Determine The Vasa's location, specifying the sea or ocean body and the Exclusive Economic Zone (EEZ) of the relevant coastal nation.

Subtask 4 Goal
Location: Stockholm, Sweden

Subtask 5 Instruction
Search museum databases and archaeological reports to find a museum that currently exhibits The Vasa.

Subtask 5 Goal
Museum: Vasa Museum

Subtask 6 Instruction
From the Vasa Museum's official collection catalog or website, find the official name of the specific artifact related to the anchor stock in the museum's collection.

Subtask 6 Goal
Name: Ankarstock

Web Task Example - Finance and Economics

Task Instruction
Among all Chinese banks listed in Hong Kong from 2022 to 2023, list the bank with the highest increase in net interest margin ranking, and provide: (1) bank name, (2) net interest margin values before and after the increase, (3) stock code, (4) total asset growth rate, and (5) chairman's name.

Task Goal
Bank Name: Hang Seng Bank
Net Interest Margin Values Before and After the Increase: 1.75%, 2.30%
Stock Code: 0011.HK
Total Asset Growth Rate: -8.75%
Chairman's Name: Irene Lee

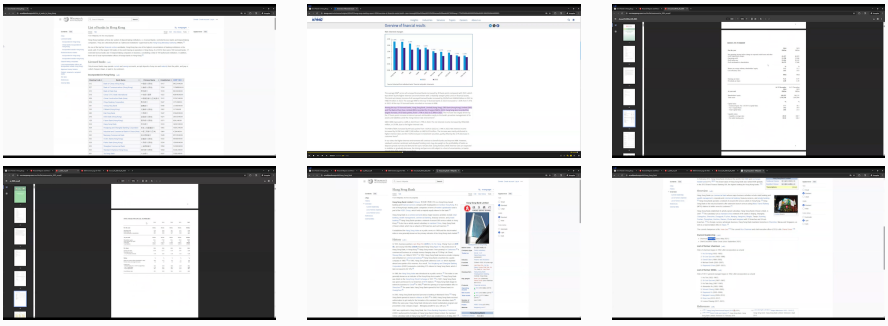


Figure 9: Human demonstration screenshots for the finance and economics web task example.

Subtask 1 Instruction
Collect the list of the top 10 licensed banks in Hong Kong by total assets in 2024 and their respective annual net interest margin (NIM) data.

Subtask 1 Goal
Hongkong and Shanghai Banking Corporation Limited (The): 10,500,393 HK$ million
Bank of China (Hong Kong) Limited: 3,685,578 HK$ million
Standard Chartered Bank (Hong Kong) Limited: 2,534,695 HK$ million
Hang Seng Bank, Limited: 1,692,094 HK$ million
Industrial and Commercial Bank of China (Asia) Limited: 915,960 HK$ million
Bank of East Asia, Limited (The): 860,361 HK$ million
Nanyang Commercial Bank, Limited: 555,149 HK$ million
China Construction Bank (Asia) Corporation Limited: 493,858 HK$ million
China CITIC Bank International Limited: 470,387 HK$ million
DBS Bank (Hong Kong) Limited: 467,621 HK$ million

Subtask 2 Instruction
Collect the list of the top 10 licensed banks in Hong Kong from 2022 to 2023 and their annual net interest margin data, and calculate the increase in net interest margin for each bank and identify the bank with the largest increase.

Subtask 2 Goal
Bank: Hang Seng Bank
NIM Increase: 55bp

Subtask 3 Instruction
Find the following for the bank: (1) name, (2) specific net interest margin values for 2022 and 2023, (3) stock code.

Subtask 3 Goal
Name: Hang Seng Bank
2022 NIM: 1.75%
2023 NIM: 2.30%
Stock Code: 0011.HK

Subtask 4 Instruction
Find the bank's(Hang Seng Bank) total asset data for 2022-2024 and calculate the total asset growth rate.

Subtask 4 Goal
2022 asset data: 1,854.4 HK$bn
2023 asset data: 1,692.1 HK$bn
Growth rate: -8.75%

Subtask 5 Instruction
Find the current chairman's name of the bank(Hang Seng Bank).

Subtask 5 Goal
Chairman: Irene Lee

Web Task Example - Technology and Innovation

Task Instruction
Identify the pharmaceutical company that had the FDA-approved new molecular entities (NMEs) between 2020 and 2024, where at least one of these drugs achieved blockbuster status (over $1 billion in annual sales) within 24 months of approval. List the company name, total number of NMEs approved, the name and indication of the fastest blockbuster drug, its peak annual sales figure, and the name and specialization of the lead scientist credited with its discovery.

Task Goal
Company Name: Pfizer Inc.
Total NME Approvals: 9
Details of the company with the largest number of approvals: Approval date drug trade name drug generic name 2021-11-05 Paxlovid™ nirmatrelvir/ritonavir, 2022-05-25 Cibinqo™ abrocitinib, 2023-01-30 Zavzpret® zavegepant, 2023-05-25 Paxlovid nirmatrelvir/ritonavir, 2023-06-05 Litfulo ritlegepitinib, 2023-08-22 Penbraya™ pentavalent meningococcal, 2023-10-12 Velsipity™ etrasimod, 2024-03-14 Rezdiffra* resmetirom, 2023-03-09 Zavzepant* zavegepant
Fastest Blockbuster Drug: Paxlovid (nirmatrelvir/ritonavir)
Indication: treatment of mild-to-moderate COVID-19 in adults and pediatric patients (12 years of age and older weighing at least 40 kg) who are at high risk for progression to severe COVID-19

Peak Annual Sales: $18.933 billion (2022)
Lead Scientist: Dafydd Owen
Specialization: medicinal chemist in the design and synthesis of drug-like molecules

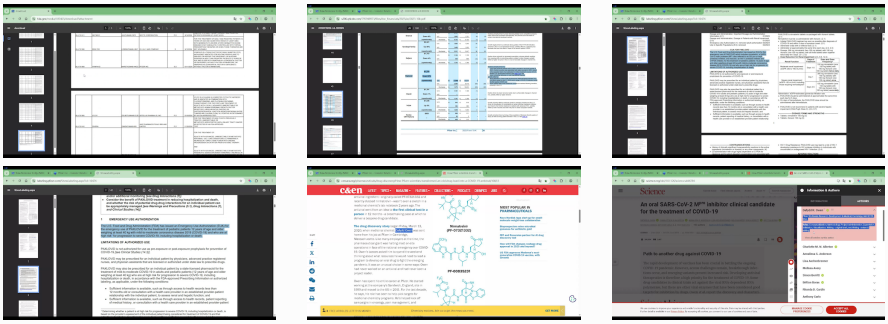


Figure 10: Human demonstration screenshots for the technology and innovation web task example.

Subtask 1 Instruction
Compile a statistical summary of all FDA-approved NMEs (2020-2024), identify the company with the highest number of approvals, and report its approved drugs with both brand and generic names.

Subtask 1 Goal
The ranking of the number of NME approvals of the company:

	Company name	Approved quantity
1	Pfizer Inc.	9
2	Novartis Pharmaceuticals	7
3	Bristol Myers Squibb	6
4	Merck Sharp & Dohme	5
5	Takeda Pharmaceuticals	4
6	Eli Lilly and Company	3

Company Name: Pfizer Inc.
Total NME Approvals: 9
Details of the company with the largest number of approvals:

Approval date	drug Trade Name	Generic Name
2021-11-05	Paxlovid™	nirmatrelvir/ritonavir
2022-05-25	Cibinqo™	abrocitinib
2023-01-30	Zavzpret®	zavegepant
2023-05-25	Paxlovid	nirmatrelvir/ritonavir
2023-06-05	Litfulo	ritlegepitinib
2023-08-22	Penbraya™	pentavalent meningococcal
2023-10-12	Velsipity™	etrasimod
2024-03-14	Rezdiffra*	resmetirom
2023-03-09	Zavzepant*	zavegepant

Subtask 2 Instruction
Among qualifying companies, identify Pfizer Inc. with the most FDA-approved NMEs and find which of their drugs reached blockbuster status fastest after approval and its peak annual sales figure.

Subtask 2 Goal
Fastest Blockbuster Drug: Paxlovid (nirmatrelvir/ritonavir)
Peak Annual Sales: $18.933 billion (2022)

Subtask 3 Instruction
Find the primary indication for Paxlovid (nirmatrelvir/ritonavir).

Subtask 3 Goal
Indication: treatment of mild-to-moderate COVID-19 in adults and pediatric patients (12 years of age and older weighing at least 40 kg) who are at high risk for progression to severe COVID-19

Subtask 4 Instruction
Search for the lead scientist or principal investigator credited with discovering Paxlovid (nirmatrelvir/ritonavir), including their full name and area of specialization.

Subtask 4 Goal
Lead Scientist: Dafydd Owen
Specialization: medicinal chemist in the design and synthesis of drug-like molecules

Web Task Example - Arts and Entertainment

Task Instruction
Identify the film with the highest production cost return ratio (box office/production cost) among all movies that grossed over $1 billion worldwide between 2020 and 2024, and list its title, director, production cost, global box office, main filming location, as well as the name of the highest-level film award it received and the city where the award ceremony was held.

Task Goal
Title: The Super Mario Bros. Movie
Director: Aaron Horvath
Production cost: $100,000,000
Global box office: $1,360,847,665
Main filming location: Paris, France
The name of the highest-level film award it received and the city where the award ceremony was held: Festival Film Bandung - Film Impor Terpuji / Commendable Imported Film, Bandung, Indonesia

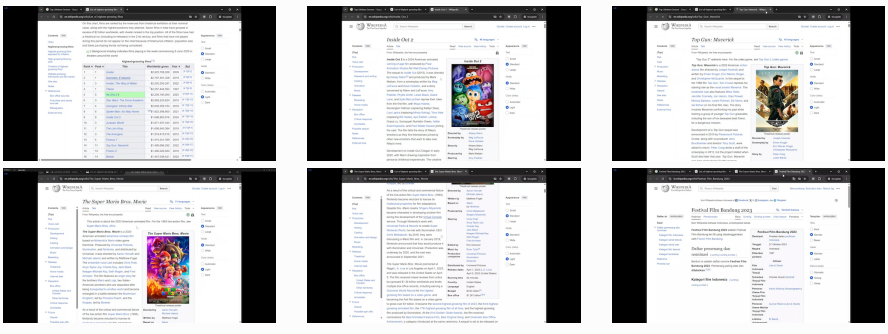


Figure 11: Human demonstration screenshots for the arts and entertainment web task example.

Subtask 1 Instruction
Collect a list of all films worldwide with box office earnings exceeding $1 billion from 2020

to 2024, along with their box office data.

Subtask 1 Goal
Avatar: The Way of Water: $2,320,250,281
Inside Out 2: $1,698,863,816
Spider-Man: No Way Home: $1,922,598,800
Top Gun: Maverick: $1,495,696,292
Barbie: $1,447,038,421
The Super Mario Bros. Movie: $1,360,847,665
Deadpool & Wolverine: $1,338,073,645
Moana 2: $1,059,242,164

Subtask 2 Instruction
Search the production cost of each film, and calculate the ratio of box office to production cost to identify the film with the highest return on investment. List only the highest-rated movies and their ratios.

Subtask 2 Goal
The Super Mario Bros. Movie, 13.61

Subtask 3 Instruction
Find the director's name of The Super Mario Bros. Movie, the specific production cost, and the exact global box office revenue.

Subtask 3 Goal
Aaron Horvath, $100,000,000, $1,360,847,665

Subtask 4 Instruction
Search for the main filming locations of The Super Mario Bros. Movie.

Subtask 4 Goal
Paris, France

Subtask 5 Instruction
Find all the film awards that The Super Mario Bros. Movie has received, identify the highest-level award among them, and find the host city of the corresponding award ceremony.

Subtask 5 Goal
Festival Film Bandung - Film Impor Terpuji / Commendable Imported Film, Bandung, Indonesia

Web Task Example - Social Policy and Sustainability

Task Instruction
Identify the G20 country that achieved the largest percentage decrease in CO2 emissions per capita between 2015 and 2023 while simultaneously recording a real GDP growth of over 20% in the same period. List the country's name, its official head of government as of year-end 2023, the primary renewable energy source by installed capacity, and the official title of its most recent Nationally Determined Contribution (NDC) report submitted to the UNFCCC.

Task Goal
The country's name: UK
UK's official head of government as of year-end 2023: Rishi Sunak
The primary renewable energy source by installed capacity: wind sources
The official title of its most recent Nationally Determined Contribution (NDC) report sub-

mitted to the UNFCCC: United Kingdom of Great Britain and Northern Ireland's 2035 Nationally Determined Contribution

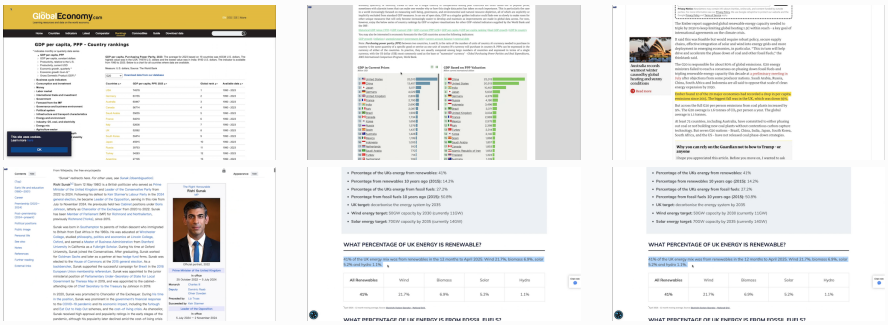


Figure 12: Human demonstration screenshots for the social policy and sustainability web task example.

Subtask 1 Instruction
Identify the G20 country that achieved a real GDP growth of over 20% between 2015 and 2023.

Subtask 1 Goal
USA, China, india, UK, Brazil, Russia, Canada, Mexico, Indonesia, Turkey, Saudi Arabia, Argentina

---

Subtask 2 Instruction
Identify the country with the largest percentage decrease in CO2 emissions per capita.

Subtask 2 Goal
The country name: UK

Subtask 3 Instruction
Find the full name of the head of government for UK, who was in office on December 31, 2023.

Subtask 3 Goal
The full name: Rishi Sunak

Subtask 4 Instruction
Research the energy profile of UK to determine its primary renewable energy source based on the latest available data for installed capacity (in MW or GW).

Subtask 4 Goal
UK's primary renewable energy source: wind sources

Subtask 5 Instruction
Search the official UNFCCC registry or UK's national environmental ministry website to find its most recently submitted Nationally Determined Contribution (NDC) report. Record its full official title and the year it was published/submitted.

Subtask 5 Goal
The official title of its most recent Nationally Determined Contribution (NDC) report submitted to the UNFCCC: United Kingdom of Great Britain and Northern Ireland's 2035 Nationally Determined Contribution