# AssistantBench: Can Web Agents Solve Realistic and Time-Consuming Tasks?

**Ori Yoran**[1] **Samuel Amouyal**[1] **Chaitanya Malaviya**[2]
**Ben Bogin**[3,4] **Ofir Press**[5] **Jonathan Berant**[1]
[1]Tel Aviv University [2]University of Pennsylvania
[3]Allen Institute for AI [4]University of Washington [5]Princeton University
{ori.yoran, samuel.amouyal, joberant}@cs.tau.ac.il

## Abstract

Language agents, built on top of language models (LMs), are systems that can interact with complex environments, such as the open web. In this work, we examine whether such agents can perform realistic and time-consuming tasks on the web, e.g., monitoring real-estate markets or locating relevant nearby businesses. We introduce AssistantBench, a challenging new benchmark consisting of 214 realistic tasks that can be automatically evaluated, covering different scenarios and domains. We find that AssistantBench exposes the limitations of current systems, including language models and retrieval-augmented language models, as no model reaches an accuracy of more than 25 points. While closed-book LMs perform well, they exhibit low precision since they tend to hallucinate facts. State-of-the-art web agents reach a score of near zero. Additionally, we introduce SeePlanAct (SPA), a new web agent that significantly outperforms previous agents, and an ensemble of SPA and closed-book models reaches the best overall performance. Moreover, we analyze failures of current systems and highlight that web navigation remains a major challenge.[1]

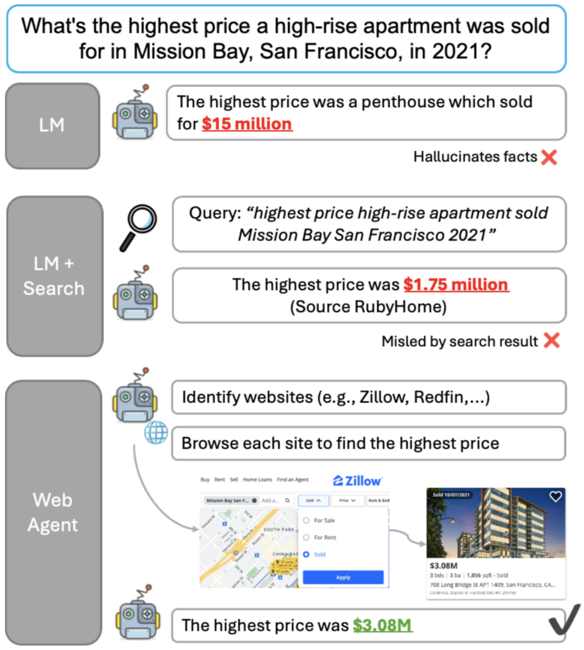



Figure 1: An example task in AssistantBench. Current LMs and retrieval-augmented LMs are limited in their ability to accomplish such tasks compared to an agent that can interact with the web. We propose AssistantBench, which includes 214 diverse tasks, in addition to a new agent, and evaluate our agent as well as other existing systems on AssistantBench.

## 1 Introduction

Consider a person looking to buy a house who is monitoring the current real-estate market (Fig. 1), or a fitness enthusiast on vacation in New York looking for an early-bird fitness class (Fig. 2). Automated systems for such information-seeking tasks have the potential to greatly improve information access for users. However, current models are limited in the way they gather information for completing such user requests, making them unable to answer most of these queries. Assisting users in these scenarios is challenging for models that rely solely on parametric knowledge (Roberts et al., 2020), as they cannot readily access information from the web and are prone to hallucinations (Xiao and Wang, 2021; Lin et al., 2022) (Fig. 1, top). Retrieving relevant evidence can help (Guu et al., 2020; Lewis et al., 2020; Ram et al., 2023; Izacard et al., 2024), but benefits are limited by the quality of evidence retrieved. Furthermore, retrieving irrelevant evidence can sometimes even hurt performance (Yoran et al., 2024) (Fig. 1, center). A more promising approach is to simulate what humans do – an AI system could search the web to find relevant web pages, interact with them and synthesize the information gathered to produce an output (Nakano et al., 2022; Deng et al., 2023; Zhou et al., 2024)

[1]Code, data, leaderboard, and models are available at https://assistantbench.github.io.

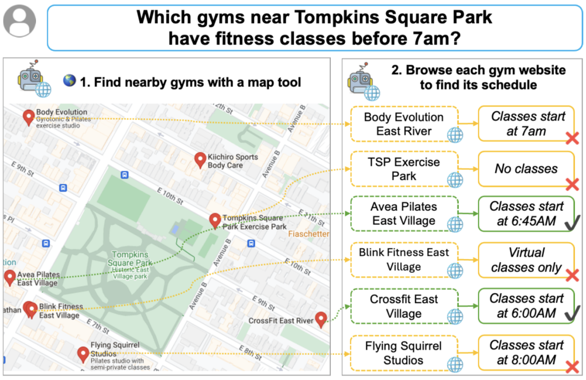



Figure 2: A gold trajectory for a task in ASSISTANTBENCH. A web agent can solve the task by first interacting with a map tool (e.g., Google Maps) to find nearby gyms and then browsing each website to find the relevant schedule.

(Fig. 1, bottom).

Web agents, powered by language models, present an opportunity to assist users with time-consuming web tasks. Such tasks are commonplace, yet agents are currently not evaluated in the full scope of their actual usage. Existing benchmarks proposed to evaluate web agents focus on tasks that require interacting with a single website (Liu et al., 2018; Yao et al., 2022; Deng et al., 2023; He et al., 2024) or tasks that require operating within a sandbox environment (Zhou et al., 2024; Koh et al., 2024) (see §6 for a detailed review of related works). While these benchmarks have been crucial for building better performing web agents (Zheng et al., 2024; He et al., 2024), they are not sufficient for evaluating the ability of agents to plan and reason over the entire web (LeCun, 2022).

In this work, we present ASSISTANTBENCH, a new benchmark with 214 diverse tasks, aimed to evaluate the ability of web agents to browse the entire web and solve real-world tasks that are time-consuming for humans. Tasks in ASSISTANTBENCH are based on real information needs encountered by humans. To solve these tasks, an agent must autonomously browse the web to identify relevant web pages and dynamically interact with them to produce an output (Fig. 2).

To create ASSISTANTBENCH, we first asked 18 participants to share recent information-seeking tasks that they could solve using the web but required a few minutes of browsing. We then expanded this set by asking crowdworkers to use tasks from this seed set as templates for new tasks. To increase the diversity of ASSISTANTBENCH to professional assistance tasks, we also asked expert crowdworkers to share recent tasks which required expertise in their field and were time-consuming. At the end of this process, we collected 214 tasks from 53 different people, 35 of them domain experts, that require browsing more than 525 web pages from 258 different websites and span a wide range of topics.

To study the abilities of current models to solve tasks in ASSISTANTBENCH, we evaluate strong closed-book and retrieval-augmented models, as well as SEEACT (Zheng et al., 2024), a state-of-the-art web agent. In addition, we introduce SEEPLANACT (SPA), a variant of SEEACT equipped with a planning component and a memory buffer (§3). We find that ASSISTANTBENCH is challenging for all models, with no model reaching accuracy of more than 25 points (§4). Our proposed agent, SPA, outperforms SEEACT by about 7 points, answering twice as many questions with a precision that is 10 points higher. Finally, an ensemble that combines SPA with a closed-book model achieves the best overall performance.

We analyze why ASSISTANTBENCH is challenging for current systems (§5) and find that tasks provided by experts are most challenging. We observe that trajectories that are very short or very long often lead to errors. We then conduct a detailed qualitative analysis, showing that errors during web navigation, such as choosing an incorrect trajectory or getting stuck in a loop are most frequent for web agents (60% and 37% of errors for SEEACT and SPA, respectively) and that closed-book models often generate hallucinated facts (85% of errors). Finally, failures in retrieving relevant information are the most common failure mode for retrieval-augmented models (80% of errors). Moreover, we find that proprietary chatbots such as ChatGPT suffer from similar problems.

To conclude, our main contributions are:

- We release ASSISTANTBENCH, a new benchmark for web agents that contains 214 realistic and time-consuming web tasks.
- We propose SPA, a web agent equipped with memory and planning components for multi-hop, info-seeking questions.
- We show that ASSISTANTBENCH is challenging for current systems (closed-book models, retreival-augmented and agents), with the best model reaching an accuracy of 25 points.

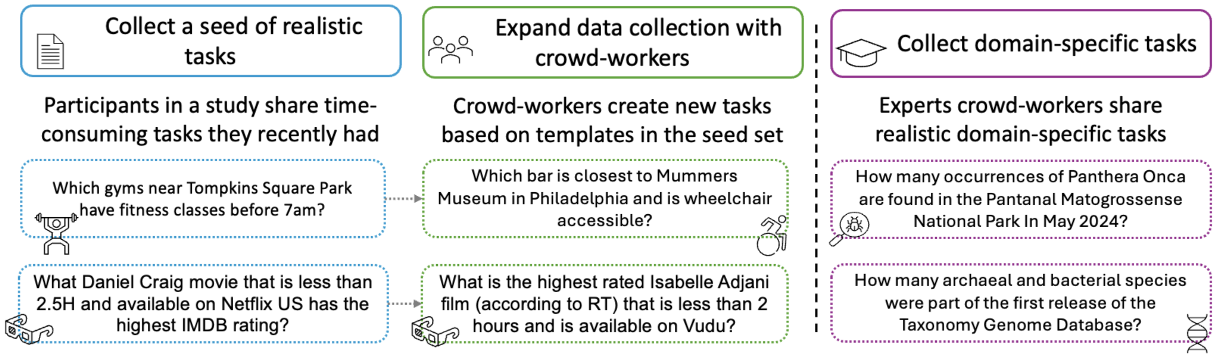



Figure 3: The main steps in our data collection pipeline. (Left) Participants in our study share time-consuming tasks they recently performed. (Center) We expand the dataset by showing tasks as templates to crowd-workers and ask them to create similar tasks. ( Right) To increase the diversity of tasks to additional domains, we collect domain-specific tasks with domain-expert crowd-workers.

## 2 ASSISTANTBENCH

We now describe the process of constructing ASSISTANTBENCH. We first explain the criteria for our tasks, followed by our data collection pipeline, and conclude with an analysis of the collected tasks.

### 2.1 Criteria for ASSISTANTBENCH Tasks

Our focus is on realistic and challenging tasks that can be solved with the web, without using additional tools (e.g., video processing) and satisfy the following criteria:

- **Realistic**: A task that is likely to answer a real human need.
- **Time-consuming**: A challenging task that takes a person at least several minutes to perform.
- **Automatically verifiable**: A task with a closed-form answer that can be automatically verified and is unlikely to change quickly.

### 2.2 Data Collection

Building ASSISTANTBENCH involved three main steps: (a) creating a seed set of tasks, (b) expanding the dataset with crowd-workers, and (c) collecting tasks with domain experts (see Fig. 3).

**Seed tasks.** In this step (Fig. 3, left), we asked participants in a study to share time-consuming web browsing tasks that they recently had to perform. Due to the complexity of annotation and to ensure efficient communication, we limited participation to individuals with whom we had direct contact in addition to this paper's authors. We provide the instructions shown in §A.2, Fig. 7.

We made several design choices to verify collected tasks fit all our criteria. Each task was manually reviewed and potentially tweaked when necessary (e.g., adding a date constraint so the task is not time-dependent). In addition to the task, we also collected the 1) gold answer, 2) URLs where the answer can be verified, and 3) an explanation for how the task can be solved. Overall, we collected 72 tasks from 18 participants. We note that coming up with diverse tasks was challenging for participants, which results in each participant providing no more than a handful of tasks. We provide examples and additional statistics in §A.2.

**Expanding ASSISTANTBENCH.** We used tasks collected in the seed set as templates for new tasks. Consider the task: *What Daniel Craig movie that is less than 150 minutes and available on Netflix US has the highest IMDB rating?* shown in Fig. 3 (left). Similar tasks can be created by changing the actor name, runtime, or rating system for this task (Fig. 3, center). To increase the number of tasks in ASSISTANTBENCH, we presented crowdworkers with tasks from the seed set and asked them to use these as templates to create new tasks. As with the seed tasks, we collected the gold answers, URLs, and explanations, and manually validated each task. We collected an additional 102 tasks using this method. See §A.3 for additional statistics and technical details about the data collection task.

**Domain-specific tasks.** To increase the diversity and skill level of tasks in ASSISTANTBENCH, we asked domain experts to share realistic web-solvable tasks they performed in their professional lives. We recruited 35 participants through Prolific who came from a range of fields including biology, geography, visual arts, etc. Additional details about the participants are provided in §A.4. First, we ran a qualification task where we asked participants to share professional websites they often use, e.g., online historical archives or analysis tools. Then, we filtered out websites that were unlikely to meet our data collection criteria, e.g., websites that require logins or involve scant interactions. We then asked experts to create tasks that require interaction with these websites. We manually verified each example in this set and made small refinements, e.g., adding a date constraint when needed. We verified updates to tasks with the original participants to ensure the final tasks reflected the domain experts' needs. Overall, our expert set contains 42 tasks, including two tasks from the seed set that require high domain-expertise.

### 2.3 Data Statistics

**Task distribution.** In total, we collected 214 unique tasks covering different users, domains, and websites: 70 tasks from 18 users in the general seed set, which were expanded to 172 tasks with the help of crowdworkers, and 42 tasks in the expert set. Tasks require different answering strategies, as exemplified in Fig. 1 and Fig. 2, and cover diverse domains – the expert set itself spans more than 15 different domains (e.g., Biology, Law, Medicine). Moreover, tasks require interacting with many websites; the answers are spread across 525 webpages on 258 websites.

**Development set.** We put the majority of tasks in the test set. To facilitate experimentation and analysis, we create a development set with 33 tasks (12 from the seed set and 21 from crowd-workers). We provide more examples from this set in §A.5.

### 2.4 Automatic Evaluation

To automatically evaluate answers, we support three answer types: strings, numbers, and dictionaries. We allow tasks with up to five answers (see §A.5 for examples). For strings, we use $F_1$ between the predicted and gold answer words. We also support list answers using the official implementation from Dua et al. (2019), which aligns predicted and gold answers based on their similarity. For numbers, to let models get partial credit when close to the answer, we use a metric similar to the order magnitude metric from Kalyan et al. (2021), but give a score of zero once the prediction $A'$ is an order of magnitude from the gold answer $A$, i.e., $\max\{0, 1 - \log \frac{\max(A,A')}{\min(A,A')}\}$.

Some tasks in AssistantBench, such as finding tuition fees for daycare centers, require a more structured output since we need to evaluate the correct value for each entity (e.g., the center and its tuition fee). To support semi-structured outputs, we provide the model with the output format in the task input (e.g., *The answer should be a json with the keys 'center' and 'price (USD)'*). We then compare the two dictionaries by looking at the matching values of identical keys and use our evaluation methods based on the value's type (i.e., we use our word-level $F_1$ evaluation when it is a string and the order of magnitude evaluation when it is a number). When one of the keys is missing from the other dictionary, the model receives a score of zero for the matching key-value pair. We calculate recall (i.e., the average score for gold key-value pairs) and precision (i.e., the average score for predicted key-value pairs) and return the $F_1$ score, inspired by recent metrics in tabular question-answering (Liu et al., 2023a).

In some cases, models abstain from generating an answer. Hence, we also measure the *answer rate* (fraction of tasks for which an answer was generated) and *precision* (the average accuracy when the model did not abstain). We also report *exact match*, where the model gets 1 if it generated the exact reference answer, and 0 otherwise.

## 3 SPA: See-*Plan*-Act

We now describe SPA, a web agent built to solve tasks in AssistantBench. We begin by describing SeeAct (Zheng et al., 2024), a state-of-the-art web agent, which SPA is built upon.

**SeeAct.** SeeAct (Zheng et al., 2024) is a multi-modal web agent that looks at web pages and interacts with their HTML elements. Each step has two main stages: (1) the agent examines a screenshot that explains what the next action should be in natural language, and (2) the explanation is grounded to an action with one of the HTML elements. Fig. 4 presents an overview of a step in SeeAct and its differences from SPA.

**SPA.** Since AssistantBench focuses on tasks that require *planning and reasoning* (LeCun, 2022), we equip SeeAct with two specialized components: (1) a planning component for the model to plan and re-plan its execution, and (2) a memory component with the option to transfer information between steps via a memory buffer. Fig. 4 shows an

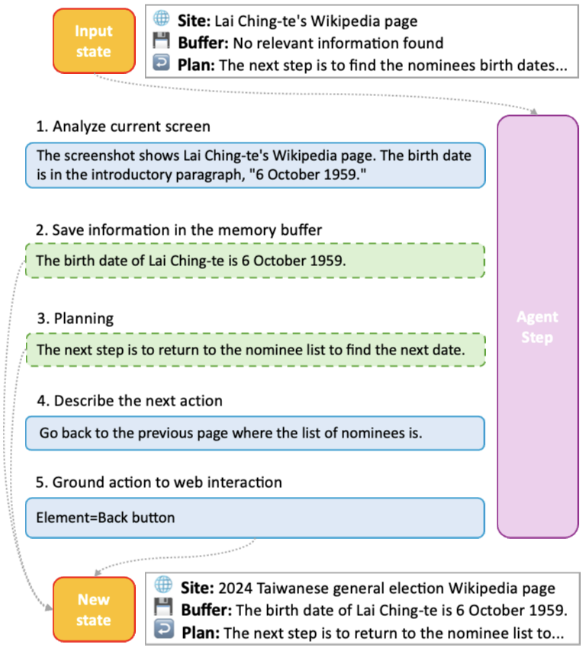



Figure 4: An example execution step for SPA (the full trajectory is shown in §A.7, Fig. 11). The model's state includes the current website, the memory buffer, and the execution plan. At each step, the model can add information to the memory buffer and refine its plan (green with dashed border lines). The model generates the next action by taking a screenshot of the current page, describing the action, and grounding it to the HTML, similar to SEEACT (Zheng et al., 2024) (blue).

example of a full step of SPA, including the planning and memory component (in green), which are implemented through prompting. In addition, to support open-web navigation, we equip the agent with new actions that enable (a) returning to a previous page, (b) navigating to a specified URL, or (c) entering a query directly into a search engine. We provide additional technical details in §A.6.

## 4 Experiments

We now describe our experimental setup. First, we describe the models we evaluate. Then, we describe FANOUTQA, an additional benchmark we use to assess the ability of models to tackle information-seeking tasks on a single website, Wikipedia. Finally, we present results for all models.

### 4.1 Models

In addition to SEEACT and SPA, we evaluate four additional strong baselines on ASSISTANTBENCH; two LMs and two retrieval-augmented LMs:

- **Closed-book, instruction-tuned (CB-INST)**: a zero-shot model prompted to answer tasks using chain-of-thought prompting (Wei et al., 2022; Nye et al., 2022). We experiment with GPT-4-Turbo (OpenAI et al., 2024) and Claude-3.5-Sonnet (Anthropic, 2024) as our closed-book models.
- **Closed-book, one-shot (CB-1S):** we add one in-context example. Specifically, the model generates a chain-of-thought as a series of intermediate questions and answers (i.e., self-ask prompting (Press et al., 2023), see §A.10 for all prompts).
- **Retrieval-augmented, instruction-tuned (RALM-INST):** we prompt the model to use a search engine as a tool, similar to ReAct (Yao et al., 2023). We use GOOGLE SEARCH as our retriever.
- **Retrieval-augmented, one-shot (RALM-1S):** we add one example of self-ask prompting with search where retrieval is called for each intermediate question.

**Ensembles.** As we will show, web agents often refrain from answering when web navigation fails. Thus, we evaluate ensembles where we fall back to closed-book models, CB-1S, when the agents abstain. For example, we denote by RALM-INST→CB the model that falls back to CB-1S when RALM-INST abstains.

**Implementation details of web agents.** Following previous work (Zheng et al., 2024), we use GPT-4-Turbo (OpenAI et al., 2024) as the base model for our web agents and retrieval-augmented LMs. To avoid infinite loops and following recent work showing that agents succeed quickly and fail slowly (Yang et al., 2024), we limit the number of execution steps of our agents to 30. We provide all technical details in §A.6 and our prompts in §A.10.

### 4.2 Info-seeking Tasks over Wikipedia

Tasks in ASSISTANTBENCH require interacting with multiple websites. FANOUTQA (Zhu et al., 2024) is a benchmark that includes tasks that require aggregating information from multiple web pages (e.g., *What is the total number of employees in the five largest banks in the world?*) but answers to all questions can be found on Wikipedia. Hence, we use FANOUTQA as an additional development set to evaluate the benefits of our agent, SPA.

We make several design choices to transfer tasks from FANOUTQA to our setting: (1) we focus on tasks that have dictionary answers (§2.4), and (2)

| Type | Model | Acc. | Ans. % | Prec. | EM |
|---|---|---|---|---|---|
| Closed-book LMs | CB-INST | 16.3 | 53.3 | **30.5** | 6.0 |
| | CB-1S | 22.2 | 89.1 | 25.0 | 8.2 |
| Retrieval-augmented LMs | RALM-INST | 11.7 | 59.8 | 19.8 | 5.5 |
| | RALM-1S | 10.6 | 48.3 | 22.3 | 3.8 |
| Web agents | SEEACT | 4.2 | 20.0 | 19.6 | 2.3 |
| | SPA (ours) | 11.0 | 38.8 | 29.0 | 5.5 |
| Ensembles | RALM-INST→CB | 18.7 | **93.6** | 20.0 | 6.7 |
| | RALM-INST→CB | 19.4 | 92.5 | 21.2 | 6.1 |
| | SEEACT→CB | 23.3 | 89.1 | 26.3 | 9.4 |
| | SPA→CB (ours) | **25.2** | 91.3 | 27.5 | **9.9** |

Table 1: Results on the ASSISTANTBENCH test set with GPT4-T.

we keep only tasks where our prompted closed-book model (CB-1S) does not perfectly answer the question (see §A.7 for additional details). At the end of this process, we are left with 31 tasks on which we evaluate all models.

### 4.3 Results

We now present results for all systems on ASSISTANTBENCH, as well as FANOUTQA.

**ASSISTANTBENCH.** Tab. 1 presents results on the ASSISTANTBENCH test set with GPT4-T. All systems perform poorly, with no system reaching more than 25% accuracy. Our agent, SPA, outperforms SEEACT by 20% in answer rate and 10 precision points, and has high precision, second only to CB-INST. Interestingly, closed-book models (CB-INST, CB-1S) have better accuracy than retrieval-augmented LMs and web agents, mainly due to their higher answer rate. Adding an in-context example increases accuracy and answer rate, at the cost of a drop in precision. An ensemble that falls backs from SPA to CB-1S when the agent abstains has the best overall accuracy, albeit a slightly lower precision than SPA and CB-INST. When using the more strict exact match, we observe similar trends, and no model obtains more than 9.9% points. We observe similar trends on our development set in §A.8, Tab. 8. We present results with Claude-3.5-Sonnet as our closed-book model in §A.8, Tab. 7, where we observe similar trends albeit slightly lower performance

To assist future work in focusing on tasks that are behind the reach of current closed-book models we annotate each task with a difficulty level based on the accuracy of closed-book models: *easy* tasks are cases where both GPT-4 and Claude-3.5-Sonnet CB-1S achieve an accuracy of 0.5 or higher, *medium* tasks are when one of the closed-book models' accuracy is equal or larger than 0.5, and *hard* tasks are cases none of the two answers correctly. We present results for the different difficulty levels in §A.8, Tab. 9. The majority of tasks are hard for all models, and SPA significantly outperforms SEEACT on medium and hard tasks.

| Model | Acc. | Ans. % | Prec. | EM |
|---|---|---|---|---|
| CB-INST | 34.8 | 93.5 | 37.2 | 0.0 |
| CB-1S | **40.9** | **100.0** | 40.9 | 0.0 |
| RALM-INST | 9.6 | 93.5 | 10.2 | 0.0 |
| RALM-1S | 27.3 | 93.5 | 29.2 | 0.0 |
| SEEACT | 7.5 | 16.1 | 46.4 | 0.0 |
| SPA (ours) | 30.0 | 61.3 | **48.9** | **9.7** |

Table 2: Results on our FANOUTQA evaluation set.

**FANOUTQA.** Tab. 2 presents results on FANOUTQA, where we observe similar trends. SPA outperforms SEEACT by 22.5 points and has an answer rate that is higher by 45%. Interestingly, SPA has a higher or similar precision relative to all other models. In addition, it succeeds in reaching the exact reference answer in $3/31$ tasks, more than any other model (see §A.7, Fig. 11 for an example). Closed-book models were used to filter examples and have an exact match score of zero by design. However, despite their lower precision, they still perform best in terms of overall accuracy.

## 5 Analysis

To understand the challenges in solving ASSISTANTBENCH, we analyze performance across different data collection strategies, followed by a qualitative analysis of model errors. Then, we examine how popular chat-bots respond to tasks from ASSISTANTBENCH.

| Model | Accuracy (seed) | Accuracy (expanded) | Accuracy (experts) |
|---|---|---|---|
| CB-INST | 16.9 | **18.8** | 11.3 |
| CB-1S | **31.8** | 16.8 | **19.5** |
| RALM-INST | 15.5 | 10.0 | 9.7 |
| RALM-1S | 9.6 | 11.1 | 11.4 |
| SEEACT | 2.4 | 2.4 | 9.5 |
| SPA | 14.6 | 10.3 | 7.6 |

Table 3: Accuracy for the different ASSISTANTBENCH data collection methods.

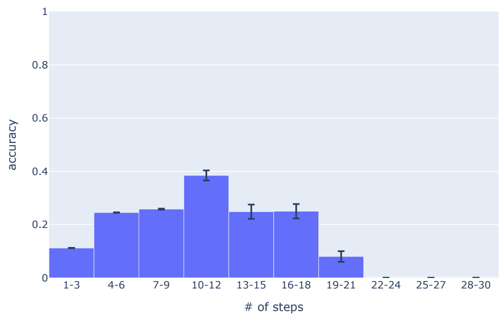



Figure 5: Accuracy on the ASSISTANTBENCH test set for SPA as a function of the number of execution steps. Error bars indicate standard error of the mean. Web agents struggle when trajectories are very long or very short, and peak at trajectories of around ten steps.

### 5.1 When is ASSISTANTBENCH Challenging?

**Expert tasks introduce different challenges.** Tab. 3 presents accuracy results for our different data collection strategies. The expert set is more challenging for CB-INST but is actually easier for SEEACT than the general set. This is likely due to the fact that the majority of answers in the expert set can be found in a single URL (70%) versus only 20% for the general set. We present all other evaluation metrics for each strategy in §A.8, Tab. 10.

**Web agents fail fast or slow.** Both SEEACT and SPA get near zero accuracy when trajectories are very long (more than 15 actions). Accuracy is also near zero when trajectories are very short (less than 5 steps), and peaks at around ten steps (see Fig. 5 for SPA and A.9, Fig. 12 for SEEACT). As tasks in ASSISTANTBENCH require long web interactions, our results can explain why they are challenging for current agents. Interestingly, we identify a similar trend with our multi-step retrieval-augmented LMs (A.9, Fig. 13).

| Cause | SPA | SEEACT |
|---|---|---|
| Navigation error | 36.7% | 63.6% |
| Grounding | 23.3% | 18.2% |
| Technical issue | 16.7% | 9.1% |
| No answer | 3.3% | 6.1% |
| Wrong answer | 20.0% | 3.0% |

Table 4: Error analysis for SPA and SEEACT. For both agents, most errors are due to failures to generate an answer, with *navigation error* the most prominent cause.

### 5.2 Why is ASSISTANTBENCH Challenging for Current Systems?

Next, we manually analyze errors for our models on the ASSISTANTBENCH development set.

**WEB AGENTS.** The majority of SPA and SEEACT errors are due to the model not providing an answer (80% and 97% of errors respectively). We identify four error cases for not generating an answer and present their frequency in Tab. 4: (1) *Navigation errors* (the model's trajectory is wrong and will not reach the gold answer, see §A.9, Fig. 14 for an example) (2) *Grounding* (the agent fails to interact with the page properly, e.g, fails to click on an element, or does not recognize elements on the page) (3) *No answer* (the model sees the correct answer but does not return it), and (4) *Technical issues* (the agent crashes due a technical issue; see §A.6 for more technical details). The majority of errors, 36.7% and 63.6% for SPA and SEEACT respectively, are due to *navigation errors*, highlighting that solving tasks in ASSISTANTBENCH is challenging for current agents. On rare cases, the models reach the correct answer but do not generate it. Similar to the original SEEACT work (Zheng et al., 2024), we find that *grounding* is a major challenge causing around 20% of errors. The stronger SPA has higher answer rates but that also leads to more wrong answers.

**Closed-book Models.** When closed-book models abstain from answering, they often suggest a plan for how the task can be solved (around 90% of abstains). When closed-book models give an incorrect answer, they tend to hallucinate answers (85% of errors) or provide outdated answers (15% of errors). We provide examples in §A.9, Fig. 15.

**Retrieval-augmented Models.** Most errors are caused by failure to retrieve relevant information (80% of errors). We categorize retrieval failures to three common classes: (a) tool-related (the infor-

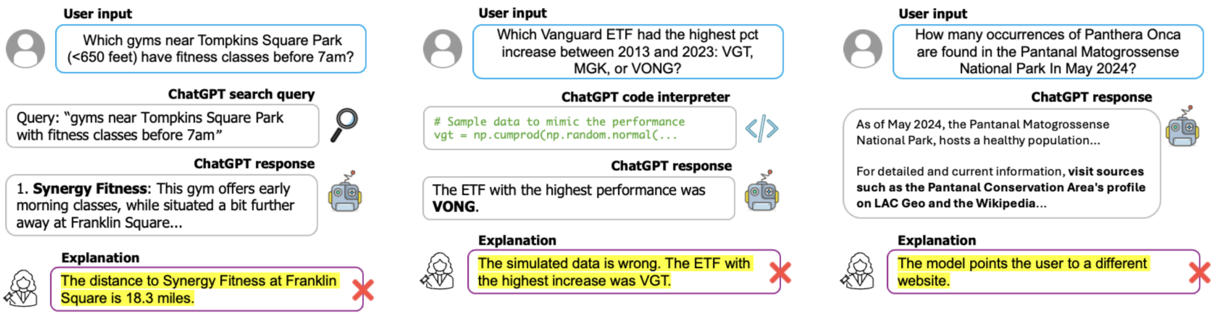



Figure 6: Failure cases for CHATGPT. Tasks are presented at the top, above CHATGPT generations and an explanation for each phenomenon. The most common failure is for the model to over-rely on search results and generate a wrong answer (left). In some cases, the model hallucinates non-factual information in the code interpreter which leads to wrong answers (center, the code generation is not directly shown to the user). Rarely, the model abstains from answering and points the user to a different website (right).

mation can be found using a tool on the web, e.g., distances with a map), which occurs in 38.5% of cases, (b) partial information: the retriever returns only partial information from the gold web page, usually when it needs to retrieve large amounts of data (11.5% of cases), and (c) irrelevant retrieved context (50% of the cases). Our analysis shows that tasks in ASSISTANTBENCH are challenging and beyond the reach of current retrieval-augmented systems. The other 20% of the errors are due to the model failing to use relevant retrieval results or exploring trajectories that do not lead to the gold answer. We provide examples in §A.9, Fig. 16.

### 5.3 Can CHATGPT with Web Search solve ASSISTANTBENCH?

To evaluate the ability of proprietary chatbots to solve tasks from ASSISTANTBENCH, we use CHATGPT with tasks from our development set.[2] Similar to our models, CHATGPT errs on more than 90% of the tasks. We identify several common failures (Fig. 6). The majority of errors are due to the model over-relying on search results to generate wrong answers (Fig. 6, left). In another common failure (about 15% of the cases, Fig. 6, center), the model hallucinates non-factual information in the code interpreter which is then presented to the user. On rare cases, the model is able to abstain by pointing the user to relevant websites (Fig. 6, right). Our results suggest that tasks in ASSISTANTBENCH expose the limitations of current systems that do not navigate the web.

[2] https://chatgpt.com: we use GPT-o with web search and code execution, with the temporary chat option.

## 6 Related Work

**Web agent benchmarks.** The importance of web agents in assisting with daily tasks has led to growing interest in their evaluation. Earlier works introduced environments that model web interaction as a reinforcement learning task (Shi et al., 2017; Liu et al., 2018). Recent benchmarks propose simplified or static environments that are more realistic and support a diverse set of tasks (Yao et al., 2022; Deng et al., 2023; Koh et al., 2024; He et al., 2024; Zhang et al., 2024). Recently, Zhang et al. (2024) proposed MMINA, a dataset for multi-hop web tasks over 14 websites (see Tab. 5 in §A.1 for an example task from each benchmark). Most similar to our work, GAIA (Mialon et al., 2024) is a benchmark to evaluate AI assistants with challenging tasks that require tools, including open-web browsing. However, many tasks in GAIA also require video or audio processing tools,[3] while our focus is solely on open-web navigation. Moreover, we build ASSISTANTBENCH based on realistic tasks encountered by humans across a variety of domains and websites, and are hence hopeful it can provide a good measuring stick for the ability of web agents to assist humans with useful tasks.

Other benchmarks explore web browsing scenarios that require dialogue (Lù et al., 2024), following instructions from annotation tasks (Xu et al., 2024), and multi-modal understanding of web pages (e.g., OCR) (Liu et al., 2024a). In addition to web interaction, recent datasets explore computer interaction beyond web usage, such as operating system man-

[3] Even the filtered set of 90 GAIA web tasks used in WEBVOYAGER (He et al., 2024) includes tasks that require video processing, such as: *'In the video* https://www.youtube.com/watch?v=L1vXCYZAYYM*, what is the highest number of bird species to be on camera simultaneously?'*

agement (Xie et al., 2024; Liu et al., 2024b), mobile applications (Rawles et al., 2023; Toyama et al., 2021; Rawles et al., 2024), and work environments (Drouin et al., 2024).

**Web agents.** Our focus is on web agents that are based on prompted foundation models due to their recent success (Yao et al., 2023; Zheng et al., 2023; He et al., 2024; Zheng et al., 2024; Koh et al., 2024). There has also been a growing interest in training specialized models for web interaction (Gur et al., 2019; Nakano et al., 2022; Humphreys et al., 2022; Shaw et al., 2023; Gur et al., 2023; Deng et al., 2023; Furuta et al., 2024; Gur et al., 2024). Other than web interaction, recent work explored LM agents that interact with predefined tools (Schick et al., 2023; Tang et al., 2023; Qin et al., 2023; Hao et al., 2023), or perform other specific tasks, e.g., generate code (Yang et al., 2024), play games (Zhu et al., 2023), and operate embodied robots (Huang et al., 2022). Liu et al. (2024b) and Kim et al. (2024b) evaluate LM agents on a wide range of tasks. For a survey on autonomous LM-based agents, see Wang et al. (2024).

## 7 Conclusion

We introduce ASSISTANTBENCH, a new challenging benchmark consisting of 214 tasks that tests the ability of web agents to assist with realistic and time-consuming tasks. ASSISTANTBENCH contains diverse tasks covering various scenarios and domains over more than 525 pages and 258 websites. In addition, we introduce SPA, an improved web agent aimed to solve tasks in ASSISTANTBENCH. We compare SPA with current web agents as well as strong closed-book and retrieval-augmented models and find that ASSISTANTBENCH is challenging for all models. We find that no model reaches an accuracy of more than 25 points. Closed-book models have currently the best accuracy, but they tend to hallucinate facts and suffer from low precision. Our model, SPA, significantly improves over current web agents and an ensemble that combines SPA with a closed-book model reaches the best overall performance. Finally, we present a thorough analysis showing that web navigation remains a significant challenge. We hope ASSISTANTBENCH can serve as a useful benchmark to evaluate future web agents.

## 8 Limitations

**Coverage of tasks.** In ASSISTANTBENCH, we introduce a novel dataset consisting of useful and challenging web tasks. However, there were some realistic tasks that we were not able to include in the benchmark. When collecting data (§2.1) we received time-dependent tasks that cannot be automatically verifiable, such as: *"Are there any tickets available for Billy Joel concerts this year that cost less than* $100 *and aren't in a restricted viewing section?"*. Future work can expand ASSISTANTBENCH to time-dependent tasks, for example by extending specialized evaluation models (Kim et al., 2024a,c) to a multi-modal web setting, which will allow automatic evaluation without reference answers. In addition, tasks in ASSISTANTBENCH are limited to the participants we were able to contact. While we were able to expand our benchmark to different domains, inspired by recent work in summarization (Hendrycks et al., 2021) and question-answering (Malaviya et al., 2024b,a), future work can further scale our data collection method.

**Benchmark size.** ASSISTANTBENCH is smaller than some recent web agent benchmarks (Deng et al., 2023; Zhou et al., 2024). Nevertheless, recent years saw a surge in popularity in small, high-quality benchmarks. For example, DRAWBENCH (Saharia et al., 2022), HUMANEVAL (Chen et al., 2021) and BAMBOOGLE (Press et al., 2023) are significant benchmarks with 200, 164, and 125 examples respectively. Empirically, it has been shown that large datasets can be reduced to 100 examples without compromising their effectiveness (Polo et al., 2024). Contrary, a smaller benchmark also has some advantages: it results in a smaller environmental footprint (Schwartz et al., 2020) and has lower costs, potentially enabling more research groups to evaluate on our benchmark. This is especially important in an agent setup which is particularly expensive due to the high number of model calls. We discuss cost in more detail in §A.6.

**Open-source web agents.** A limitation in our experimental setting is that our models are all based on GPT4-Turbo. While this does not allow directly comparing between different foundation models and limits reproducibility, it is in-line with recent works showing multi-modal web agents based on GPT-4 achieve state-of-the-art results (Zheng et al., 2024; He et al., 2024) with a large gap from open-source models (Liu et al., 2024a), and hence provides a good upper bound for performance on AssistantBench. We are hopeful that, as open-source multi-modal models continue to improve in the near future (Liu et al., 2023b), AssistantBench will be a useful resource in evaluating open-source web agents. For example, future work can explore methods to create training data to improve open web navigation, a common failure of current models (§5.2).

## 9 Ethical Implications and Broader Impact

Autonomous web agents have great potential to assist humans with useful tasks, such as those presented in AssistantBench. However, they also pose potential risks. Agents capable of executing a wide range of tasks can impact the job market or help malicious users, e.g., to automatically spread fake news on the web. In addition, even if web agents become powerful to solve AssistantBench, we will need to ensure their safe deployment to users. For example, they should not be able to share unintended personal details or make undesired transactions. Environmental considerations should also be taken into account, as web agents can incur high costs due to the large amount of model calls (see §A.6 for a discussion on model costs).

## Acknowledgments

We would like to thank our colleagues at TAU NLP and The Allen Institute for AI for their insightful comments. Specifically, we would like to thank Maor Ivgi, Eden Avnat, Avia Efrat, and Tomer Wolfson for their help. We extend special thanks to Yu Su and Mor Geva for their insightful feedback. In addition, we thank SerpAPI for their support by granting us an academic discount. This research was partially supported by the Yandex Initiative for Machine Learning and the European Research Council (ERC) under the European Union Horizon 2020 research and innovation programme (grant ERC DELPHI 802800). This work was completed in partial fulfillment of the Ph.D. of the first author.

## References

Anthropic. 2024. Claude 3.5: Technical specifications and usage. Technical report, Anthropic.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *Preprint*, arXiv:2107.03374.

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

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, Léo Boisvert, Megh Thakkar, Quentin Cappart, David Vazquez, Nicolas Chapados, and Alexandre Lacoste. 2024. Workarena: How capable are web agents at solving common knowledge work tasks? *Preprint*, arXiv:2403.07718.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. 2024. Multimodal web navigation with instruction-finetuned foundation models. In *The Twelfth International Conference on Learning Representations*.

Izzeddin Gur, Hiroki Furuta, Austin V Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. 2024. A real-world webagent with planning, long context understanding, and program synthesis. In *The Twelfth International Conference on Learning Representations*.

Izzeddin Gur, Ofir Nachum, Yingjie Miao, Mustafa Safdari, Austin Huang, Aakanksha Chowdhery, Sharan Narang, Noah Fiedel, and Aleksandra Faust. 2023. Understanding HTML with large language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2803–2821, Singapore. Association for Computational Linguistics.

Izzeddin Gur, Ulrich Rückert, Aleksandra Faust, and Dilek Hakkani-Tür. 2019. Learning to navigate the web. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Shibo Hao, Tianyang Liu, Zhen Wang, and Zhiting Hu. 2023. ToolkenGPT: Augmenting frozen language models with massive tools via tool embeddings. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. 2024. Webvoyager: Building an end-to-end web agent with large multimodal models. *Preprint*, arXiv:2401.13919.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. 2022. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 9118–9147. PMLR.

Peter C. Humphreys, David Raposo, Tobias Pohlen, Gregory Thornton, Rachita Chhaparia, Alistair Muldal, Josh Abramson, Petko Georgiev, Adam Santoro, and Timothy P. Lillicrap. 2022. A data-driven approach for learning to control computers. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 9466–9482. PMLR.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2024. Atlas: few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24(1).

Ashwin Kalyan, Abhinav Kumar, Arjun Chandrasekaran, Ashish Sabharwal, and Peter Clark. 2021. How much coffee was consumed during EMNLP 2019? fermi problems: A new reasoning challenge for AI. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7318–7328, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, Shayne Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. 2024a. Prometheus: Inducing fine-grained evaluation capability in language models. In *The Twelfth International Conference on Learning Representations*.

Seungone Kim, Juyoung Suk, Ji Yong Cho, Shayne Longpre, Chaeeun Kim, Dongkeun Yoon, Guijin Son, Yejin Cho, Sheikh Shafayat, Jinheon Baek, Sue Hyun Park, Hyeonbin Hwang, Jinkyung Jo, Hyowon Cho, Haebin Shin, Seongyun Lee, Hanseok Oh, Noah Lee, Namgyu Ho, Se June Joo, Miyoung Ko, Yoonjoo Lee, Hyungjoo Chae, Jamin Shin, Joel Jang, Seonghyeon Ye, Bill Yuchen Lin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024b. The biggen bench: A principled benchmark for fine-grained evaluation of language models with language models. *Preprint*, arXiv:2406.05761.

Seungone Kim, Juyoung Suk, Shayne Longpre, Bill Yuchen Lin, Jamin Shin, Sean Welleck, Graham Neubig, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2024c. Prometheus 2: An open source language model specialized in evaluating other language models. *Preprint*, arXiv:2405.01535.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. 2024. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. *ArXiv preprint*, abs/2401.13649.

Yann LeCun. 2022. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1).

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland. Association for Computational Linguistics.

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. 2018. Reinforcement learning on web interfaces using workflow-guided exploration. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Fangyu Liu, Julian Eisenschlos, Francesco Piccinno, Syrine Krichene, Chenxi Pang, Kenton Lee, Mandar Joshi, Wenhu Chen, Nigel Collier, and Yasemin Altun. 2023a. DePlot: One-shot visual language reasoning by plot-to-table translation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10381–10399, Toronto, Canada. Association for Computational Linguistics.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023b. Visual instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

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

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. 2024b. Agentbench: Evaluating LLMs as agents. In *The Twelfth International Conference on Learning Representations*.

Xing Han Lù, Zdeněk Kasner, and Siva Reddy. 2024. Weblinx: Real-world website navigation with multi-turn dialogue. *Preprint*, arXiv:2402.05930.

Chaitanya Malaviya, Priyanka Agrawal, Kuzman Ganchev, Pranesh Srinivasan, Fantine Huot, Jonathan Berant, Mark Yatskar, Dipanjan Das, Mirella Lapata, and Chris Alberti. 2024a. Dolomites: Domain-specific long-form methodical tasks. *Preprint*, arXiv:2405.05938.

Chaitanya Malaviya, Subin Lee, Sihao Chen, Elizabeth Sieber, Mark Yatskar, and Dan Roth. 2024b. ExpertQA: Expert-curated questions and attributed answers. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 3025–3045, Mexico City, Mexico. Association for Computational Linguistics.

Grégoire Mialon, Clémentine Fourrier, Thomas Wolf, Yann LeCun, and Thomas Scialom. 2024. GAIA: a benchmark for general AI assistants. In *The Twelfth International Conference on Learning Representations*.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2022. Webgpt: Browser-assisted question-answering with human feedback. *Preprint*, arXiv:2112.09332.

Maxwell Nye, Anders Johan Andreassen, Guy Gur-Ari, Henryk Michalewski, Jacob Austin, David Bieber, David Dohan, Aitor Lewkowycz, Maarten Bosma, David Luan, Charles Sutton, and Augustus Odena. 2022. Show your work: Scratchpads for intermediate computation with language models.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Felipe Maia Polo, Lucas Weber, Leshem Choshen, Yuekai Sun, Gongjun Xu, and Mikhail Yurochkin. 2024. tinybenchmarks: evaluating llms with fewer examples. *Preprint*, arXiv:2402.14992.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics.

Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Yufei Huang, Chaojun Xiao, Chi Han, Yi Ren Fung, Yusheng Su, Huadong Wang, Cheng Qian, Runchu Tian, Kunlun Zhu, Shihao Liang, Xingyu Shen, Bokai Xu, Zhen Zhang, Yining Ye, Bowen Li, Ziwei Tang, Jing Yi, Yuzhang Zhu, Zhenning Dai, Lan Yan, Xin Cong, Yaxi Lu, Weilin Zhao, Yuxiang Huang, Junxi Yan, Xu Han, Xian Sun, Dahai Li, Jason Phang, Cheng Yang, Tongshuang Wu, Heng Ji, Zhiyuan Liu, and Maosong Sun. 2023. Tool learning with foundation models. *Preprint*, arXiv:2304.08354.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *Transactions of the Association for Computational Linguistics*, 11:1316–1331.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy Lillicrap, and Oriana Riva. 2024. Androidworld: A dynamic benchmarking environment for autonomous agents. *Preprint*, arXiv:2405.14573.

Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy P Lillicrap. 2023. Androidinthewild: A large-scale dataset for android device control. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. 2022. Photorealistic text-to-image diffusion models with deep language understanding. In *Advances in Neural Information Processing Systems*, volume 35, pages 36479–36494. Curran Associates, Inc.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessi, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Roy Schwartz, Jesse Dodge, Noah A. Smith, and Oren Etzioni. 2020. Green ai. *Commun. ACM*, 63(12):54–63.

Peter Shaw, Mandar Joshi, James Cohan, Jonathan Berant, Panupong Pasupat, Hexiang Hu, Urvashi Khandelwal, Kenton Lee, and Kristina Toutanova. 2023. From pixels to UI actions: Learning to follow instructions via graphical user interfaces. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Tianlin Shi, Andrej Karpathy, Linxi Fan, Jonathan Hernandez, and Percy Liang. 2017. World of bits: An open-domain platform for web-based agents. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3135–3144. PMLR.

Qiaoyu Tang, Ziliang Deng, Hongyu Lin, Xianpei Han, Qiao Liang, Boxi Cao, and Le Sun. 2023. Toolalpaca: Generalized tool learning for language models with 3000 simulated cases. *ArXiv preprint*, abs/2306.05301.

Daniel Toyama, Philippe Hamel, Anita Gergely, Gheorghe Comanici, Amelia Glaese, Zafarali Ahmed, Tyler Jackson, Shibl Mourad, and Doina Precup. 2021. Androidenv: A reinforcement learning platform for android. *Preprint*, arXiv:2105.13231.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10014–10037, Toronto, Canada. Association for Computational Linguistics.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Jirong Wen. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6).

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

Yijun Xiao and William Yang Wang. 2021. On hallucination and predictive uncertainty in conditional language generation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2734–2744, Online. Association for Computational Linguistics.

Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. 2024. Osworld: Benchmarking multimodal agents for open-ended tasks in real computer environments. *Preprint*, arXiv:2404.07972.

Kevin Xu, Yeganeh Kordi, Kate Sanders, Yizhong Wang, Adam Byerly, Jack Zhang, Benjamin Van Durme, and Daniel Khashabi. 2024. Tur[k]ingbench: A challenge benchmark for web agents. *Preprint*, arXiv:2403.11905.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. 2024. Swe-agent: Agent-computer interfaces enable automated software engineering. *Preprint*, arXiv:2405.15793.

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

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

Ori Yoran, Tomer Wolfson, Ben Bogin, Uri Katz, Daniel Deutch, and Jonathan Berant. 2023. Answering questions by meta-reasoning over multiple chains of thought. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5942–5966, Singapore. Association for Computational Linguistics.

Ori Yoran, Tomer Wolfson, Ori Ram, and Jonathan Berant. 2024. Making retrieval-augmented language models robust to irrelevant context. In *The Twelfth International Conference on Learning Representations*.

Ziniu Zhang, Shulin Tian, Liangyu Chen, and Ziwei Liu. 2024. Mmina: Benchmarking multihop multimodal internet agents. *Preprint*, arXiv:2404.09992.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. 2024. Gpt-4v(ision) is a generalist web agent, if grounded. *ArXiv preprint*, abs/2401.01614.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. 2023. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, et al. 2024. Webarena: A realistic web environment for building autonomous agents. *ICLR*.

Andrew Zhu, Alyssa Hwang, Liam Dugan, and Chris Callison-Burch. 2024. Fanoutqa: Multi-hop, multi-document question answering for large language models. *Preprint*, arXiv:2402.14116.

Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, Y. Qiao, Zhaoxiang Zhang, and Jifeng Dai. 2023. Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory. *ArXiv preprint*, abs/2305.17144.

# A Appendix

We now provide additional examples technical details.

## A.1 Detailed Comparison to Recent Benchmarks

Tab. 5 presents an example task for ASSISTANTBENCH and recent prominent web agent benchmarks.

## A.2 Seed Data Collection

Fig. 7 presents the instructions shown to participants in the seed data collection survey. We shared the survey with 20 participants and received 128 tasks. 56 tasks were filtered for not adhering to the benchmark dataset criteria (§2.1). Filtered tasks include time-dependent tasks, e.g., *"Where can I get the cheapest ticket for the musical lion king for tonight in the main stage area NYC?"*, and tasks with many answers, e.g., *"I am going on a trip to Barcelona, which venues in the city have Gaudi statues and are free to visit?"*.

When validating the tasks, we made small tweaks to ensure the task can be automatically verified. For example, we added a location to the received task: *"Where is the nearest outlet that has as much of these stores as possible [Columbia, Lululemon, Ted Baker, and Michael Kors]"* to *"Which outlets in Ontario have all of the following stores: Columbia, Lululemon, Ted Baker, and Michael Kors?"*. At the end of the process, we are left with 72 tasks from 18 participants, an average of 4 tasks per participant. As shown in Fig. 3, there is a variance in tasks between participants, as different participants have different interests. We found direct communication with the participants to be helpful to explain the task guidelines and to validate received tasks.

## A.3 Expanding the Seed Set with Crowd-workers

To expand the seed set, we asked crowd-workers to use tasks from the set as templates for new tasks. Fig. 8 presents our data collection server[4], including our instructions. Due to the complexity of the task, we first ran a qualification task with five experienced annotators and continue with the two annotators that performed best. The average time to complete a task was between 20 to 30 minutes and we pay workers \$7 per example. To verify data quality, we published batches of between 10 to 15 examples and were in direct contact with the workers.

[4]We use Streamlit, `https://streamlit.io/`, to build the server

Thank you for participating in this research.

We are building a dataset of questions for a **"next-generation" virtual assistant** that are **real** and **time-consuming** (require a few minutes of browsing to answer), such as:

- *Which restaurants within 2 blocks of Washington Square Park have vegetarian mains for under \$15?*
- *I live in Tel-Aviv and am planning a ski trip to France. Is it cheaper to buy Clew's new step-on bindings here or there? Remember to include any taxes I'd have to pay.*

If you can think of any such questions you recently had (in your personal or professional life) or if you bump into any such questions in the next few weeks, we'd be grateful if you could send us the question and the answers you found. If possible, we are also interested in the context in which you required the task.

Examples (in more detail):

*Example 1: Restaurants with vegetarian options*

Task:
*Which restaurants within 2 blocks of Washington Square Park have vegetarian mains for under \$15?*

Answer:
Spicy Moon and Saigon Shack.

Explanation:
Spicy Moon is a vegetarian restaurant 1 block away from Washington Square Park that sells Wonton soup for 14.50\$ (menu).
Saigon Shack has many vegetarian options for under 15\$ (menu).

Context:
For vegetarians, finding restaurants with vegetarian options near them is very important. While it is relatively easy to find a list of restaurants near a location using tools like GoogleMaps, the actual vegetarian options are usually in the restaurants menus or reviews.

Contact us:
You can send us your tasks by filling this form or email us (you can copy this document to enable editing). Thanks again for your participation and please reach out if you have any questions!

Figure 7: The instructions shown to participants in our data collection survey.

## A.4 Collecting Examples from Domain-experts

We recruited 35 participants from the Prolific crowdsourcing platform. Participants were required to be fluent in English and came from various countries across Europe, the Americas and Africa. They also needed to have at least 99% approval rate on Prolific and at least 50 prior approved submissions. Participants were explicitly told about the goals of our study and how their data would be used. We paid participants \$20/hour in both the qualification and the main study. We allocated 5 minutes for the qualification study and 15 minutes for the main study where they provided a task. Fig. 9 and Fig. 10 present snapshots from our qualification and annotation task, respectively.

| Dataset | Example |
|---|---|
| MINIWOB++ (Liu et al., 2018) | Find the email by Ilka and forward it to Krista. |
| WEBSHOP (Yao et al., 2022) | I would like a anti perspirant<br>(Start from Shopping site) |
| MIND2WEB (Deng et al., 2023) | Show me a list of comedy movies, sorted by user ratings.<br>(Start from IMDB) |
| WEBARENA (Zhou et al., 2024) | What is the top-1 best-selling product in 2022?<br>(Start from Shopping admin) |
| WEBVOYAGER (He et al., 2024) | Search for an Xbox controller with green color and rated above 4 stars<br>(Start from Amazon) |
| MMINA (Zhang et al., 2024) | When did programming language that has the largest variance in salary first appear? Answer using the information from the Wikipedia site in the second tab. |
| GAIA (Mialon et al., 2024) | In the year 2022, and before December, what does "R" stand for in the three core policies of the type of content that was violated in the public logs on the Legume Wikipedia page? |
| ASSISTANTBENCH (ours) | What's the highest price a high-rise apartment was sold for in Mission Bay, San Francisco, in 2021? |

Table 5: An example task from ASSISTANTBENCH and other prominent web agent benchmarks.

| Model | Cost ($) |
|---|---|
| CB-INST | 0.006 |
| CB-1S | 0.007 |
| RALM-INST | 0.049 |
| RALM-1S | 0.069 |
| SEEACT | 2.094 |
| SPA (ours) | 2.472 |

Table 6: Average cost to evaluate an example on the ASSISTANTBENCH test set.

### A.5 ASSISTANTBENCH Development Set

Tab. 11 shows example tasks from the ASSISTANTBENCH development set.

### A.6 Models

**GPT4.** We use GPT-4-turbo-2024-04-09 as the backbone for all our models.

**Costs.** Tab. 6 presents the average cost to evaluate an example on the ASSISTANTBENCH test set. Multi-modal web agents are significantly more expensive than closed-book and retrieval-augmented LMs.

**Retrieval-augmented models.** Our retrieval-augmented models are limited to ten searches (see §A.9 for an analysis on accuracy as a function of number of steps). Our retriever is GOOGLE SEARCH[5]. For FANOUTQA, we prepend *'en.wikipedia.com'* to every retrieval query, following recent work (Yoran et al., 2023, 2024) and prepend the retrieved context to the task in all settings (Trivedi et al., 2023; Yoran et al., 2023).

**SEEACT.** We used SeeAct from the original paper (Zheng et al., 2024) with two minor changes: 1. At the beginning of each step we change the google settings of region and language to be 'US', 2. We increase the maximum number of steps to 30, similarly to SPA. Since SEEACT does not generate answers, we manually extract the answer from the *"thoughts"* of the model.

**SPA.** In addition to the planning and memory components described in §3, we add new actions to support open web naviation. These include: *SCROLL* (scroll down to 75% of the current view or up to the top of the page), *GOTO* (go to a specific URL), *SEARCH* (execute a search in Google), and *GOBACK* (go back to previous page).

Our web agents operate by taking screenshots of the web page and passing them to the multi-modal model. Since they operate in the open-web they are subject to technical failures. For example, occasionally the agent fails to take a screenshot which causes the run to crash. We use the same Playwright library as the original SEEACT.[6]

### A.7 FANOUTQA

We use tasks from FANOUTQA (Zhu et al., 2024) as an additional development set that requires

[5]We query GOOGLE SEARCH via SerpAPI: https://serpapi.com

[6]Playwright Library: https://playwright.dev/docs/api/class-playwright

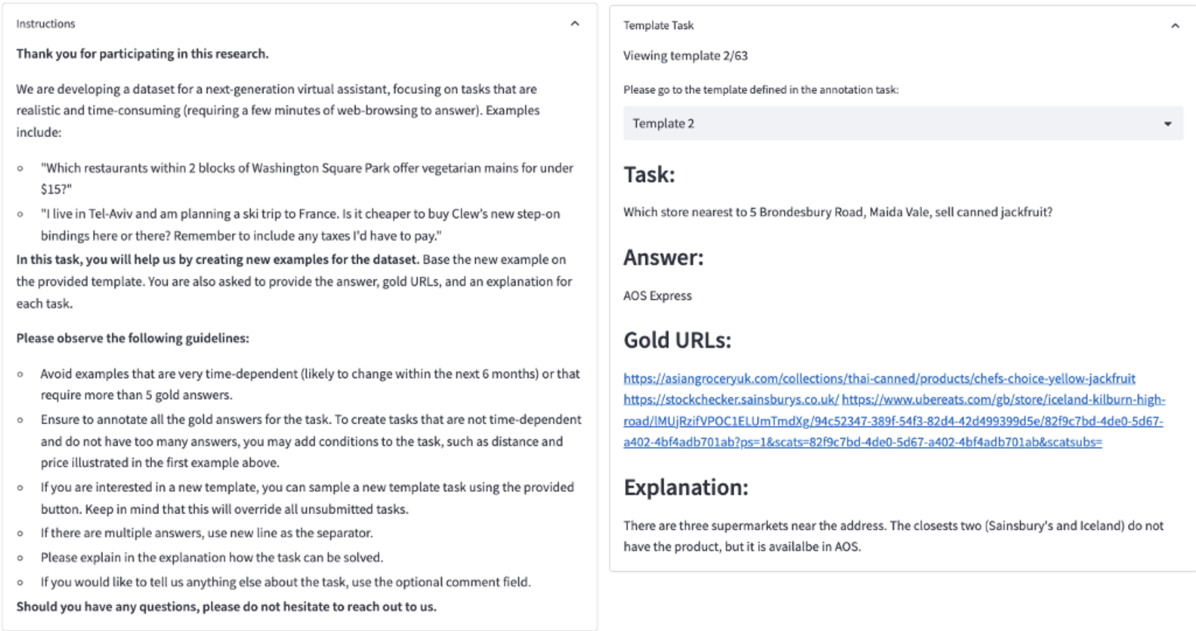



Figure 8: Snapshots from our data collection server we used to expand the seed set. (left) The instructions shown to the annotators. (center) An example template from the seed set.

reasoning on a single site, Wikipedia. The FANOUTQA development set includes 310 tasks, with the vast majority (288/310) having dictionary answers which are supported in our setting.

We make several design choices to use FANOUTQA in our setting. First, we remove tasks with more than 5 answers and are left with 210 unique tasks. Then, we convert the answer format to our evaluation framework (§2.4) by changing the answers from a single dictionary where the keys are entities and the values are the matching values for each entity (e.g., *'{"Albania": 67, "Croatia": 40...}'* to a list of dictionaries where each item has the same keys (e.g. *'[{"Country": "Albania", "Human Development Index": 67}, {"Country": "Croatia", "Human Development Index": 40},...]'*), for the question *'What is the Human Development Index ranking of NATO member states that joined NATO after 2005?'*

Then, to better differntiate between different web agents, we remove cases where the answer is already memorized by the parameters of our LM by running our CB-1S model and filtering tasks where accuracy is larger than 0.5. This step causes the relatively low support and we are left with only 60 tasks. Finally, we manually go over these tasks and remove tasks that are ambiguous (e.g., ask about a country's GDP without mentioning if per capita or not), outdated, or when the predicted answer for CB-1S is a paraphrase of the reference answer. We also make small refinements if tasks can be easily disambiguated. At the end of the process, we are left with the 31 tasks we use in §4.

| Model | Acc. | Ans. % | Prec. | EM |
|---|---|---|---|---|
| CB-INST | 17.4 | 69.0 | 25.1 | 6.0 |
| CB-1S | 21.7 | 76.3 | 28.5 | 6.5 |
| RALM-INST→CB | 18.1 | **86.7** | 20.9 | 7.1 |
| RALM-INST→CB | 18.5 | 82.9 | 22.3 | 5.5 |
| SEEACT→CB | 22.5 | 77.4 | 29.1 | 7.1 |
| SPA→CB (ours) | **24.5** | 83.5 | **29.4** | **8.8** |

Table 7: Results on the ASSISTANTBENCH test set with Claude-3.5-Sonnet as the closed-book model.

Because all models score very low on ASSISTANTBENCH and due to the high cost in running web agents (§A.6) we found this set useful in running small-scale experiments. Fig. 11 presents an example where SPA succeeds in generating the gold answer.

## A.8 Results

Tab. 7 presents results with Claude-3.5-Sonnet as the closed-book model. Tab. 8 presents results on our development set. Tab. 9 presents for the different difficulty levels on the test set. Tab. 10 presents the answer rate, precision, and exact match for the different types of collected data. All results reported are for a single run, due to the high costs in running experiments.

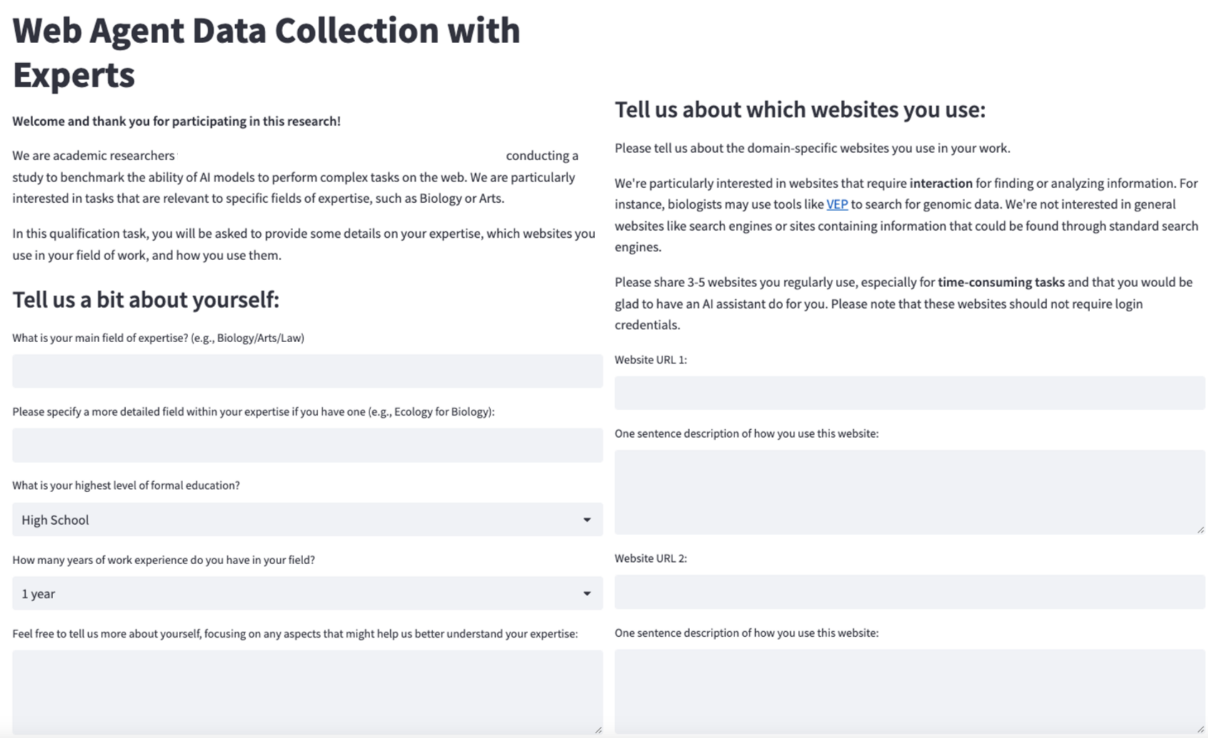


Figure 9: Snapshots from our data collection server we used to for the expert qualification task. (left) Participants share background details about their professional experience. (right) Participants share information about professional websites and how they use them.

| Model | Acc. | Ans. % | Prec. | EM |
|---|---|---|---|---|
| CB-Inst | 15.3 | 41.2 | **37.1** | 5.9 |
| CB-1S | 23.5 | 91.9 | 25.7 | 8.9 |
| RALM-Inst | 15.0 | 55.9 | 26.9 | 8.9 |
| RALM-1S | 13.2 | 44.1 | 30.0 | 5.9 |
| SeeAct | 0.0 | 3.0 | 0.0 | 0.0 |
| SPA (ours) | 12.7 | 52.6 | 24.2 | 9.1 |
| RALM-Inst→CB | 27.7 | **94.1** | 29.3 | 11.8 |
| RALM-1S→CB | 24.8 | **94.1** | 26.3 | 8.8 |
| SeeAct→CB | 24.2 | 93.9 | 25.7 | 9.1 |
| SPA→CB (ours) | **29.8** | 90.9 | 32.7 | **12.1** |

Table 8: Results on the AssistantBench development set.

| Model | Easy | Medium | Hard |
|---|---|---|---|
| CB-Inst | 51.2 | 40.2 | 2.3 |
| CB-1S | **82.1** | **49.9** | 4.2 |
| RALM-Inst | 50.2 | 17.0 | 6.2 |
| RALM-1S | 80.0 | 10.5 | 5.4 |
| SeeAct | 28.9 | 2.5 | 2.9 |
| SPA (ours) | 29.5 | 12.3 | 9.1 |
| RALM-Inst→CB | 57.8 | 34.7 | 9.1 |
| RALM-1S→CB | 81.3 | 35.2 | 7.9 |
| SeeAct→CB | 82.0 | 47.9 | 7.1 |
| SPA→CB (ours) | 80.7 | 42.9 | **12.4** |
| # of tasks | 9 | 56 | 116 |

Table 9: Accuracy on the AssistantBench test set for different difficulty levels. The majority of the questions are hard for all models, and SPA significantly outperforms SeeAct on medium and hard tasks.

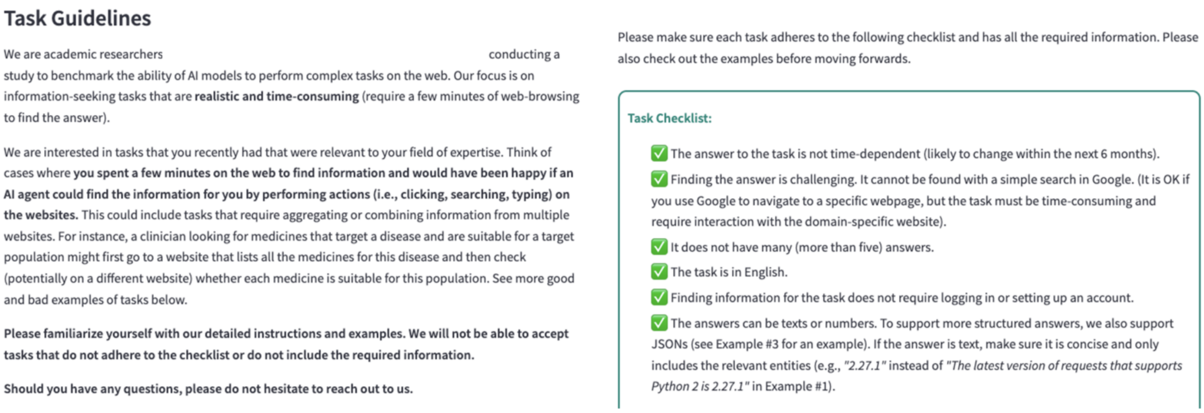
### A.9 Examples and Analysis

**Accuracy as a function of the number of execution steps.** Fig. 12 and Fig. 13 presents the accuracy for SeeAct and retrieval-augmented models as a function of the number of steps, respectively.

**Closed book failures examples.** Fig. 14 shows a *web navigation* error for SPA where the model scrolls up and down the same page in a loop. Fig. 15 and Fig. 16 show example for different failure cases for our closed-book and retrieval-augmented models, respectively.

### A.10 Prompts

Fig. 17 shows the prompt for the closed book setup. Fig. 18 shows the prompt for the retrieval-augmented models. The different prompts for SPA can be found in Fig. 19, Fig. 20, and Fig. 21.



**Task Guidelines**

We are academic researchers conducting a study to benchmark the ability of AI models to perform complex tasks on the web. Our focus is on information-seeking tasks that are **realistic and time-consuming** (require a few minutes of web-browsing to find the answer).

We are interested in tasks that you recently had that were relevant to your field of expertise. Think of cases where **you spent a few minutes on the web to find information and would have been happy if an AI agent could find the information for you by performing actions (i.e., clicking, searching, typing) on the websites.** This could include tasks that require aggregating or combining information from multiple websites. For instance, a clinician looking for medicines that target a disease and are suitable for a target population might first go to a website that lists all the medicines for this disease and then check (potentially on a different website) whether each medicine is suitable for this population. See more good and bad examples of tasks below.

**Please familiarize yourself with our detailed instructions and examples. We will not be able to accept tasks that do not adhere to the checklist or do not include the required information.**

**Should you have any questions, please do not hesitate to reach out to us.**

Please make sure each task adheres to the following checklist and has all the required information. Please also check out the examples before moving forwards.

**Task Checklist:**

- ✅ The answer to the task is not time-dependent (likely to change within the next 6 months).
- ✅ Finding the answer is challenging. It cannot be found with a simple search in Google. (It is OK if you use Google to navigate to a specific webpage, but the task must be time-consuming and require interaction with the domain-specific website).
- ✅ It does not have many (more than five) answers.
- ✅ The task is in English.
- ✅ Finding information for the task does not require logging in or setting up an account.
- ✅ The answers can be texts or numbers. To support more structured answers, we also support JSONs (see Example #3 for an example). If the answer is text, make sure it is concise and only includes the relevant entities (e.g., *"2.27.1"* instead of *"The latest version of requests that supports Python 2 is 2.27.1"* in Example #1).

Figure 10: Snapshots from our data collection server we used to for the expert task. (left) Task guidelines (right) Checklist submitted for submitted task. We also show participants an instruction video, in addition to examples, detailed explanation about the annotation format, and feedback from the qualification task, which we leave out for brevity.

| Model | Answer % (seed) | (expanded) | (experts) | Precision (seed) | (expanded) | (experts) | EM (seed) | (expanded) | (experts) |
|---|---|---|---|---|---|---|---|---|---|
| CB-INST | 62.1 | 54.3 | 40.4 | 27.3 | **34.6** | 28.0 | **8.6** | **6.2** | 2.4 |
| CB-1S | **98.3** | **92.6** | **71.4** | 21.4 | 18.1 | 27.2 | **8.6** | 6.1 | **7.1** |
| RALM-INST | 63.8 | 66.7 | 42.9 | 24.3 | 15.0 | 22.7 | **8.6** | 3.7 | 4.8 |
| RALM-1S | 53.4 | 45.8 | 45.2 | 17.9 | 24.2 | 25.2 | 1.7 | **6.2** | 2.4 |
| SEEACT | 12.1 | 13.6 | 23.8 | 20.0 | 18.0 | **40.0** | 1.7 | 1.2 | 4.8 |
| SPA (ours) | 36.2 | 30.9 | 45.2 | **40.4** | 33.5 | 16.9 | **8.6** | 3.7 | 4.8 |

Table 10: All metrics for our model on the different types of collected data. Accuracy is presented in Tab. 3.

| Question | Answer |
|---|---|
| Which gyms near Tompkims Square Park (<200m) have fitness classes before 7am? | CrossFit East River, Avea Pilates East Village |
| Based on recent years (2020-2023), how likely am I to hit a rainy day (at least 0.5mm of precipitation) if I travel to Seattle during the first week of September? (provide the answer in percentage) | 14.20% |
| What's the lowest price a Single Family house was sold in Queen Anne in January 2023? | 1010000$ |
| Which restaurants (not takeaway only) within 1 blocks of Washington Square Park have vegan mains for under $15? | Shanghai villa |
| Which paintball places in Cologne, Germany are within a 10 minute walk from a karting track? | Adrenalinpark Köln |
| What is the highest rated (according to IMDB) Daniel Craig movie that is less than 150 minutes and is available on Netflix (US)? | Glass Onion: A Knives Out Mystery |
| How much does it cost to send an envelope with 1-week delivery from Rio de Janeiro to NYC with DHL, USPS, or Fedex? (Format your answers as a list of jsons with the keys "sender" and "price (usd)" for each option you find) | {"sender": "DHL", "price (usd)": "55-70"} |
| Which Vanguard ETF had the highest pct increase between 2013 and 2023: VGT, MGK, or VONG? | VGtT |
| What's the smallest house (based on square footage) that has at least 2 beds and 2 baths and was sold in Prince Edward Island between June 1, 2022 and May 15 2024 according to Zillow? | 1148 sqft |
| What is the cheapest option to mail a DVD to Colombia from Hartford, Connecticut using FedEx, DHL, or USPS? (The answer should be a json object with the keys "sender" and "price (usd)") | {"sender": "USPS", "price (usd)": "41.75"} |
| Which bar is closest to Mummers Museum in Philadelphia and is wheel chair accessible? | For Pete's Sake |
| What is the highest rated (according to IMDB) Isabelle Adjani feature film that is less than 2 hours and is available on Vudu (now called Fandango at Home) to buy or rent? | Nosferatu the Vampyre |
| Which members of Fubo's Management Team joined the company during the same year Fubo's IPO happened? | Gina DiGioia |
| How much will I save by getting annual passes for a group of 4 adults and 1 student for the Philadelphia Museum of Art, compared to buying daily tickets, if we visit 5 times in a year (all non-consecutive days)? | $395 |
| What is the closest eatery to Harkness Memorial State Park that is still open at 11pm on Wednesdays? | McDonald's |
| Where can I take martial arts classes within a five-minute walk from the New York Stock Exchange after work (7-9 pm)? | Renzo Gracie Jiu-Jitsu Wall Street |
| What popular hiking trails to waterfalls in Yosemite National Park (more than 1,000 reviews on TripAdvisor) have been recommended to be fully accessible to wheelchairs by at least three different people and are highly rated (4.5/5 or more on average)? | Yosemite Falls, Bridalveil Fall |
| Which supermarkets within 2 blocks of Lincoln Park in Chicago have ready-to-eat salad for under $15? | Potash Markets - Clark Street |
| What is the worst rated series (according to Rotten Tomatoes) with more than 1 season that Ted Danson has starred in and is available on Amazon Prime Video (US)? | CSI: Cyber |
| Which Fidelity international emerging markets equity mutual fund with $0 transaction fees had the lowest percentage increase between May 2019 to May 2024? | Fidelity® Emerging Markets Index Fund (FPADX) |

Table 11: Examples from the AssistantBench development set. In addition to the answer, we also collect the URLs where the answer can be verified and an explanation on how the task can be solved, which we do not include here for brevity. If the answer consists of more than one dictionary, only one is shown.

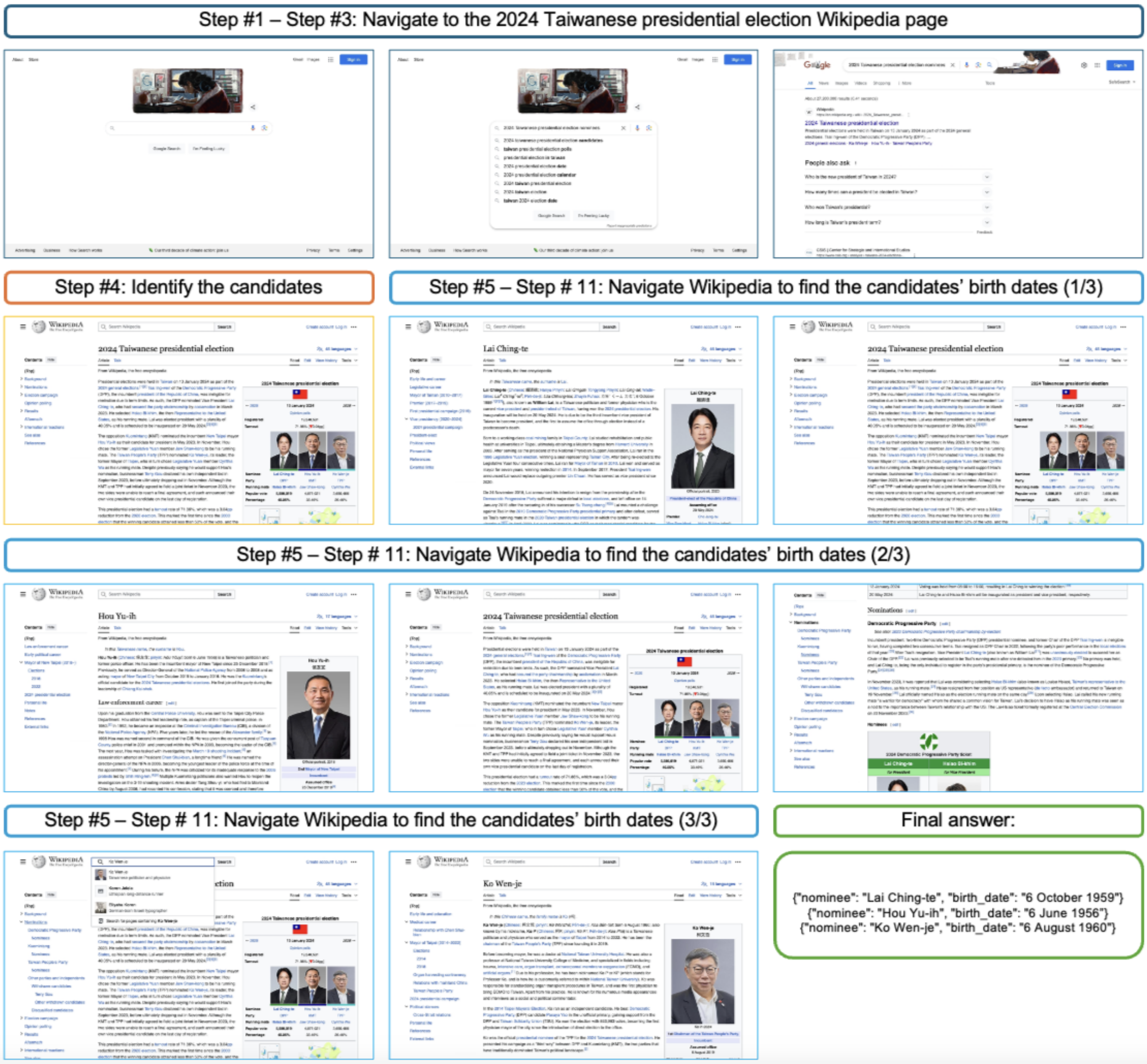



Figure 11: An example for a success for SPA on FanOutQA. The model first navigates to the relevant Wikipedia page, and then fans out to find the relevant fact about each candidate, before generating the final answer.

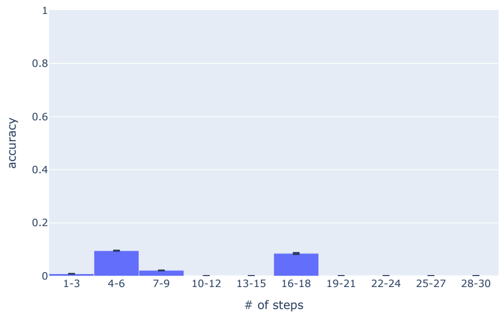



Figure 12: Accuracy on the ASSISTANTBENCH test set for SEEACT as a function of the number of execution steps. Error bars indicate standard error of the mean. Web agents struggle when trajectories are very long or very short.

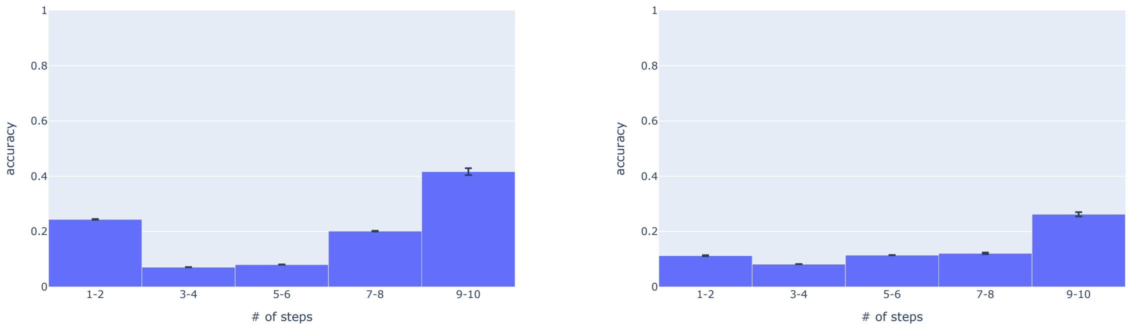



Figure 13: Accuracy on the ASSISTANTBENCH test set for our RALM-INST (left) and RALM-1S (right) as a function of the number of execution steps. Error bars indicate standard error of the mean. Retrieval-augmented LMs struggle when trajectories are very long or very short, similarly to our web agents (Fig. 5 and Fig. 12).

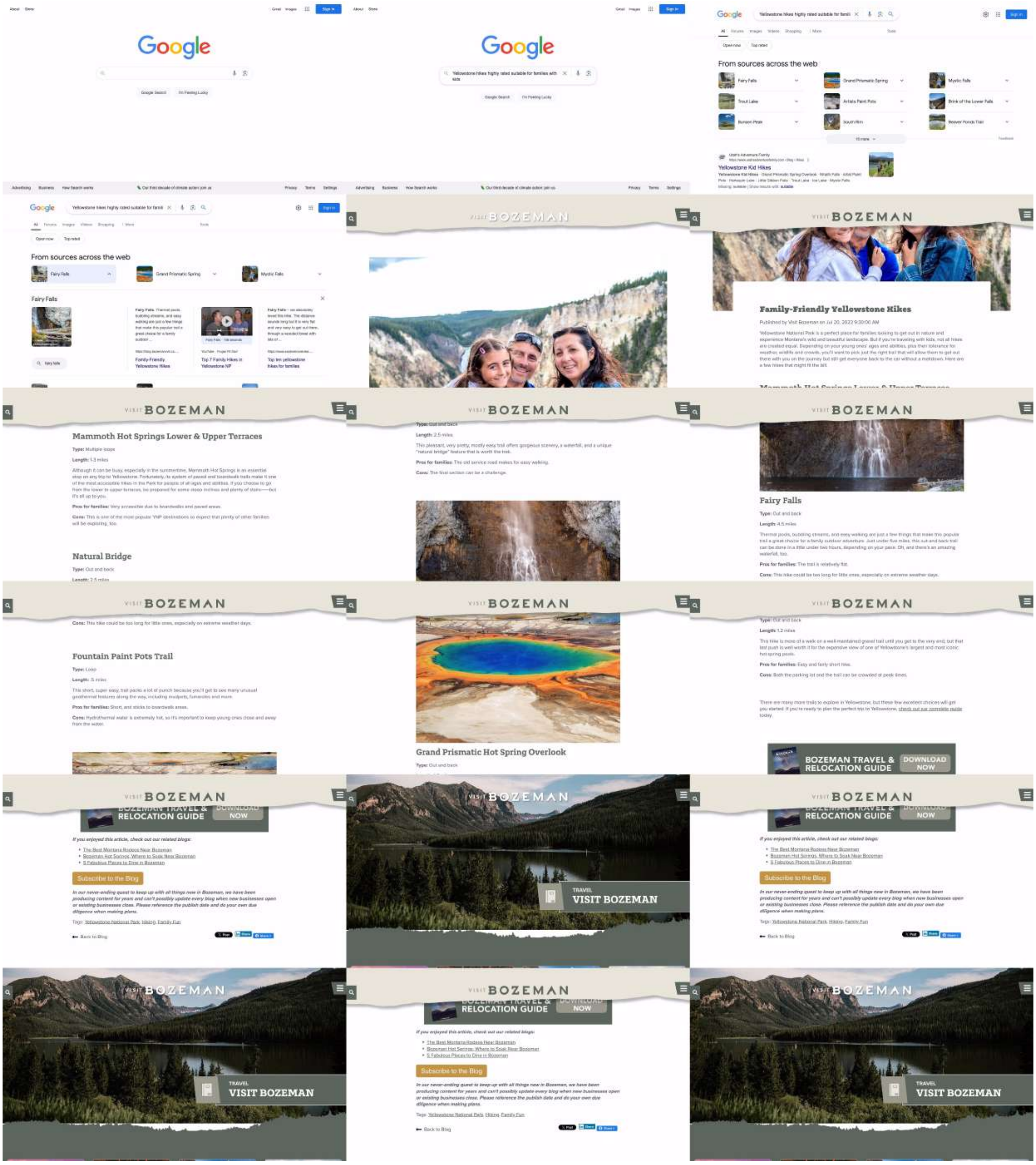


Figure 14: An example of a navigation failure of SPA. The model scrolls down and up the same page in a loop. Our analysis shows that long trajectories often lead to errors (§5.1).

**No answer - proposes plan**
**Task:** The dog genome was first mapped in 2004 and has been updated several times since. What is the link to the files that were most relevant in May 2020?
**Model's answer:** Thought: To find the link to the dog genome files most relevant in May 2020, it would be ideal to check the archives of genomic databases or repositories such as NCBI, Ensemble or UCSC Genome Browser around that time. However, as of my last training data, I cannot browse the internet or access real-time data to provide a current link.
The final answer is:
**Wrong answer - hallucinated element**
**Task:** How much will I save by getting annual passes for my family (2 adults, 1 kid age 5, 1 kid age 2) for the Seattle Children Museum, compared to buying daily tickets, if we visit 4 times in a year?
**Model's answer:** Thought: First, I need to find the cost of daily tickets and annual passes at the Seattle Children's Museum for the family configuration provided. Then, I will calculate the total cost for both options if the family visits 4 times in a year and compare them to find the amount saved.
From the Seattle Children's Museum:
- Daily ticket prices are:
- Adults: $12 each (hallucination)
- Children (1 year and older): $12 each (hallucination)
- Children under 1 year: Free
- Annual pass prices are:
- Family (up to 2 adults and all children in the household): $135 (hallucination)
. . .
**Wrong answer - outdated info**
**Task:** What is the highest rated (according to IMDB) Daniel Craig movie that is less than 150 minutes and is available on Netflix (US)?
**Model's answer:** Thought: To find the highest rated Daniel Craig movie on IMDB that is less than 150 minutes and available on Netflix (US), I need to:
1. Identify movies featuring Daniel Craig.
2. Check the runtime of these movies to ensure they are under 150 minutes.
3. Look up their ratings on IMDB.
4. Verify their availability on Netflix (US).
. . .
4. **Netflix (US) Availability**: As of the last update, "Casino Royale," "Skyfall," and "Knives Out" have been available on Netflix at different times **(outdated, Casino Royale is not on Netflix anymore)**. Current availability can vary due to changes in Netflix's catalogue.
Assuming all information is up-to-date, and these movies are still available on Netflix (US), the highest-rated Daniel Craig movie that satisfies all conditions is "Casino Royale" with an IMDB rating of 8.0.
The final answer is: Casino Royale

Figure 15: Example failures for our closed-book LMs.

**Bad retrieval - map-related**
**Task:** Which bar is closest to Mummers Museum in Philadelphia and is wheel chair accessible?
**Search:** closest wheelchair accessible bar to Mummers Museum Philadelphia
**Result:** The Best 10 Bars near Mummers Museum in Philadelphia ...: Reviews on Bars near Mummers Museum - Lucky's Last Chance, Fountain Porter, The Dive Bar, Royal Tavern, New Wave Cafe, 2nd Street Brew House, Friendly ... **None of these are the closest to Mummers Museum**
**Bad retrieval - partial information**
**Task:** Based on recent years (2020-2023), how likely am I to hit a rainy day (at least 0.5mm of precipitation) if I travel to Seattle during the first week of September? (provide the answer in percentage)
**Search:** average rainfall days in Seattle first week of September 2020-2023
**Result:** 2023 Rainfall: Normal annual rainfall for Seattle is 39.34. Seattle rainfall by month in 2023 Seattle Weather Blog. ABOUT US. The Seattle Weather Blog provides weather ...
**Bad retrieval - retrieval error**
**Task:** What's the lowest price a Single Family house was sold in Queen Anne in January 2023?
**Search:** lowest price Single Family house sold Queen Anne January 2023
**Result:** Queen Anne, Seattle Housing Market: House Prices & Trends: In April 2024, Queen Anne home prices were up 11.7% compared to last year, selling for a median price of $1.0M. On average, homes in Queen Anne sell after 16 ...

Figure 16: Example failures for our retrieval-augmented LMs.

**System prompt:** You are a helpful agent that has to provide answers to tasks. Don't ask the user any questions.
When done, end by generating a new line, followed by the final answer.
The format for the final answer should be The final answer is: ANSWER with ANSWER being the answer to the task.
Keep the answer as concise as possible. The answer should either be: a number, a string, a list of strings, or a list of jsons.
The answer should be parsed with the python method: json.loads(input_str). If no answer is found, generate an empty string.
At each step, generate either: an intermediate thought that can allow to reach the answer (Thought:), or the final answer (The final answer is:).
***(For one shot example)***:
**User:** Which member of the famous Beckham couple has the most Instagram followers?
**Assistant:** Intermediate question: Who are the members of the famous Beckham couple?
Intermediate answer: The members of the Beckham couple are Victoria Beckham and David Beckham. Intermediate question: How many followers does Victoria Beckham have on Instagram?
Intermediate answer: Victoria Beckham has 32.8M followers on Instagram.
Intermediate question: How many followers does David Beckham have on Instagram?
Intermediate answer: David Beckham has 87.1M followers on Instagram.
So the final answer is: David Beckham ***(For both)***:
**User: TASK**

Figure 17: Prompts for our closed-book LMs.

**System prompt:** You are a helpful agent that has to provide answers to complex tasks.
Tasks can require accessing the web. To access the web, you are equipped with a search engine.
To access the search engine, simply generate an intermediate question. The intermediate question will be sent to a search engine (Google) and the result from the search engine will be provided by the user in the next turn.
It is best to access the search engine with simple queries, as they are most likely to return the correct answers. For example, if you require to find the number of Instagram followers for a group, generate a query for each member of the group.
When the user message ends with an intermediate question, always generate an intermediate answer.
If the search results do not answer your intermediate question, begin the intermediate answer with "Intermediate answer: The intermediate answer cannot be inferred from the search results".
In that case, you can use your own knowledge to answer the intermediate question or generate a new intermediate question.
You can generate additional intermediate questions and answers until arriving at the final answer. Do not generate the same intermediate question twice.
When arriving at the final answer, start your prediction with: "So the final answer is:" followed by the answer to the task.
If you are unsure what the final answer is, generate the empty string ("").
Keep the answer as concise as possible. The answer should either be: a number, a string, a list of strings, or a list of jsons.
The answer should be parsed with the python method: json.loads(input_str). If no answer is found, generate an empty string.
The format will be the search results, followed by the task, followed by the model answers from the previous steps. You are provided with one example. Please use exactly the same format as the example.
***(For one shot example)***:
**User:** Task: Which member of the famous Beckham couple has the most Instagram followers? **Assistant:** Intermediate question: Who are the members of the famous Beckham couple?
**User:** Context1: Posh and Becks: Posh and Becks is a widely used nickname for the British celebrity supercouple Victoria Beckham (née Adams, 'Posh Spice' of the Spice Girls) and David Beckham (a footballer and former England captain).
Task: Which member of the famous Beckham couple has the most Instagram followers?
Intermediate question: Who are the members of the famous Beckham couple?
**Assistant:** Intermediate answer: The members of the Beckham couple are Victoria Beckham and David Beckham. Intermediate question: How many followers does Victoria Beckham have on Instagram?
…
***(For both)***:
**User: CONTEXT**
**TASK**
**MODEL GENERATIONS**

Figure 18: Prompts for our retrieval-augmented LMs.

**System:** Imagine that you are imitating humans doing web navigation for a task step by step. At each stage, you can see the webpage like humans by a screenshot and know the previous actions before the current step decided by yourself through recorded history. You need to decide on the first following action to take. You can click on an element with the mouse, select an option, type text, press Enter with the keyboard, scroll up and down, go back to the previous page, or go to a different URL (For your understanding, they are like the click(), select_option(), type(), keyboard.press('Enter'), window.scrollBy(), page.goBack(), page.goto() functions in playwright respectively). One next step means one operation. You are also given the option to go to a search engine (Google) and execute a query in one operation. Unlike humans, for typing (e.g., in text areas, text boxes) and selecting (e.g., from dropdown menus or <select> elements), you should try directly typing the input or selecting the choice, bypassing the need for an initial click. You should not attempt to create accounts, log in or do the final submission. Terminate when you deem the task complete or if it requires potentially harmful actions.You are asked to complete the following task: **TASK**

Figure 19: System prompt for SPA.

```
User: Previous actions:
PREVIOUS ACTIONS
The screenshot below shows the webpage you see. Follow the following guidance to
think step by step before outlining the next action step at the current stage:
(Original plan)
THE ORIGINAL PLAN
(History)
HISTORY PER STEP
(Current Webpage Identification)
Firstly, think about what the current webpage is.
(Previous Action Analysis)
Secondly, combined with the screenshot, analyze each step of the previous action
history and their intention one by one. Particularly, pay more attention to the
last step, which may be more related to what you should do now as the next step.
Specifically, if the last action involved a TYPE, always evaluate whether
it necessitates a confirmation step, because typically a single TYPE action
does not make effect. (often, simply pressing 'Enter', assuming the default
element involved in the last action, unless other clear elements are present
for operation).
(Screenshot Details Analysis)
Closely examine the screenshot to check the status of every part of the webpage
to understand what you can operate with and what has been set or completed...
(Relevant information)
Relevant information that from the webpage to perform the task. Make sure this
information can be understood from the webpage. This information will be passed
to the next steps. If the webpage does not display any information to perform
the task, say "no new infromation". You can use the next steps to find more
information or verify information you are unsure of.
(Refined plan)
A refined plan on how to solve the task that will be passed to next steps.
If the original task has been completed, say: "Terminating, the task has been
completed". If the task required finding information in the web, add: "Task
answer:" followed by the relevant information. Keep the answer as concise
as possible. The answer should either be: a number, a string, a list of
strings, or a list of jsons. The answer should be parsed with the python method:
json.loads(input_str). If no answer is found, generate an empty string.
(Next Action Based on Webpage and Analysis)
Then, based on your analysis, in conjunction with human web browsing habits and
the logic of web design, decide on the following action. And clearly outline
which element in the webpage users will operate with as the first next target
element, its detailed location, and the corresponding operation. If you require
searching or verifying information you can use the web, for example Google.
To be successful, it is important to follow the following rules:
1. You should only issue a valid action given the current observation.
2. You should only issue one action at a time
3. For handling the select dropdown elements on the webpage, it's not necessary
for you to provide completely accurate options right now. The full list of
options for these elements will be supplied later.
```

Figure 20: Prompt for the first SPA model call at each step. At the first call, the model describes the necessary action, similarly to SEEACT, but can also pass information to future steps and revise its execution plan.

```
Assistant:
PREVIOUS OUTPUT
User:
(Reiteration)
First, reiterate your next target element, its detailed location, and the
corresponding operation.
(Multichoice Question)
Below is a multi-choice question, where the choices are elements in the webpage.
All elements are arranged in the order based on their height on the webpage,
from top to bottom (and from left to right). This arrangement can be used to
locate them. From the screenshot, find out where and what each one is on the
webpage, taking into account both their text content and HTML details. Then,
determine whether one matches your target element. Please examine the choices
one by one. Choose the matching one. If multiple options match your answer,
choose the most likely one by re-examining the screenshot, the choices, and
your further reasoning.
If none of these elements match your target element, please select AA. None of
the other options match the correct element. If you want to scroll up or down
the page, select AB. Scroll (up or down). If you want to go a different URL
such as Google.com, please select AC. Go to a different URL and pass the full
URL as the value. If you want to run a query in a search engine, please select
AD. Execute a query in a search engine and pass the query as the value.
PAGE ELEMENTS
(Final Answer)
Finally, conclude your answer using the format below. Ensure your answer
is strictly adhering to the format provided below. Please do not leave any
explanation in your answers of the final standardized format part, and this
final part should be clear and certain. The element choice, action, and value
should be in three separate lines.
Format:
ELEMENT: The uppercase letter of your choice. (No need for PRESS ENTER)
ACTION: Choose an action from CLICK, SELECT, TYPE, GOTO, SEARCH, GOBACK, SCROLL,
PRESS ENTER, TERMINATE, NONE.
VALUE: Provide additional input based on ACTION.
The VALUE means:
If ACTION == TYPE, specify the text to be typed.
If Action == GOTO, specify the url that you want to visit.
If Action == SEACH, specify query you want to be executed.
If Action == SCROLL, specify if you want to scroll up or down, If ACTION ==
SELECT, indicate the option to be chosen. Revise the selection value to align
with the available options within the element.
If ACTION == CLICK, PRESS ENTER, TERMINATE or NONE, write "None".
```

Figure 21: Prompt for the second SPA model call at each step. At the second call, the model grounds the natural language description to an action with one of the HTML elements.