GPT-4V(ISION) IS A GENERALIST WEB AGENT, IF GROUNDED

Anonymous authors
Paper under double-blind review

ABSTRACT

The recent development on large multimodal models (LMMs), especially GPT-4V(ision) and Gemini, has been quickly expanding the capability boundaries of multimodal models beyond traditional tasks like image captioning and visual question answering. In this work, we explore the potential of LMMs like GPT-4V as a generalist web agent that can follow natural language instructions to complete tasks on any given website. We propose SEEACT, a generalist web agent that harnesses the power of LMMs for integrated visual understanding and acting on the web. We evaluate on the recent MIND2WEB benchmark. In addition to standard offline evaluation on cached websites, we enable a new online evaluation setting by developing a tool that allows running web agents on live websites. We show that GPT-4V presents a great potential for web agents—it can successfully complete 51.1% of the tasks on live websites if we manually ground its textual plans into actions on the websites. This substantially outperforms text-only LLMs like GPT-4 or smaller models (FLAN-T5 and BLIP-2) specifically fine-tuned for web agents. However, grounding still remains a major challenge. Existing LMM grounding strategies like set-of-mark prompting turns out to be not effective for web agents, and the best grounding strategy we develop in this paper leverages both the HTML structure and visuals. Yet, there is still a substantial gap with oracle grounding, leaving ample room for further improvement.[1]

Figure 1: SEEACT leverages an LMM like GPT-4V to visually perceive websites and generate plans in textual forms. The textual plans are then grounded onto the HTML elements and operations to act on the website.

1 INTRODUCTION

Large multimodal models (LMMs; Li et al. (2023); Alayrac et al. (2022); Liu et al. (2023b)), especially recent ones such as GPT-4V(ision) OpenAI (2023) and Gemini Anil et al. (2023), have shown a remarkable capability on standard vision-and-language understanding and reasoning benchmarks Kazemzadeh et al. (2014); Goyal et al. (2016); Hendrycks et al. (2020); Saikh et al. (2022); Lu et al. (2022); Zhong et al. (2023); Yue et al. (2023). While web content has been a primary source

[1]All code, data, and evaluation tools will be released.

of training data, a largely overlooked part of the web is the websites themselves—every website is designed to be rendered visually for easy consumption by human users. This poses a new challenge and a new opportunity for LMMs. On the one hand, screenshots of rendered websites, which could contain thousands of elements with rich relations, are more complex than the images in most existing benchmarks, which are usually object- or scene-centric. On the other hand, if LMMs can accurately comprehend websites, it will open the door for numerous applications on the web.

In this work, we aim to investigate the potential of LMMs as generalist web agents Deng et al. (2023). A generalist web agent, as defined in MIND2WEB Deng et al. (2023), is expected to follow natural language instructions and complete tasks on any given real-world website (e.g., Figure 1). The tasks can be fairly diverse and complex, with one task possibly taking 10+ actions across multiple dynamically rendered webpages. Existing work Deng et al. (2023); Liu et al. (2023d) primarily uses large language models (LLMs) such as GPT-4 OpenAI (2023) on the raw HTML input. However, HTML code is noisier than the rendered visuals and has a lower information density. For example, the screenshot in Figure 1 contains 423 HTML elements that would require $186\,490$ textual tokens with the GPT-2 Tokenizer, while requiring only 1445 visual tokens using GPT-4V's visual tokenizer. Furthermore, HTML alone provides incomplete information and misses critical semantics from, e.g., embedded images.

To this end, we propose SEEACT, a generalist web agent that harnesses the power of LMMs for integrated visual understanding and acting on the web. We will focus on GPT-4V, the most advanced LMM publicly available to date, and compare it with smaller LMMs such as BLIP-2 Li et al. (2023) and LLaVA-1.5 Liu et al. (2023a;c). We find that GPT-4V exhibits a strong capability in visually understanding rendered webpages and generate the right plans in textual forms across a wide range of websites and tasks. However, *grounding* Chandu et al. (2021); Gu et al. (2023), *i.e.*, converting the textual plan into precise actions on the website, remains a major challenge. It involves selecting the right HTML element to interact with as well as the right operation (e.g., `Click`, `Type`, or `Select`). We propose multiple grounding methods, including superpositioning bounding boxes and index labels onto the image, similar to set-of-mark prompting Yang et al. (2023a) that has been shown effective on object- or scene-centric images. However, we find that on complex images with rich semantic and spatial relationships like webpage screenshots, severe hallucination is observed from GPT-4V. The most effective grounding strategy leverages the known correspondence between HTML element and their visual rendering, a unique property for websites compared to natural images.

We evaluate SEEACT on the MIND2WEB dataset Deng et al. (2023) and compare it with text-only large language models (LLMs) like GPT-4 OpenAI (2023) as well as smaller models (FLAN-T5 Chung et al. (2022) and BLIP-2 Li et al. (2023) for supervised fine-tuning, LLaVA-1.5 Liu et al. (2023a;c) and CogAgent Hong et al. (2023) for in-context learning). In addition to the standard offline evaluation setting on cached websites, we further establish a new *online evaluation* setting by developing a tool that allows for running web agents on live websites. The major findings from our exploration are summarized below:

- SEEACT with GPT-4V is a strong generalist web agent, if oracle grounding is provided. In online evaluation, it can successfully complete 51.1% of tasks on different websites, substantially outperforming existing methods like GPT-4 (13.3%) or FLAN-T5 (8.9%). This strongly demonstrates the potential of LMMs like GPT-4V for web agents.
- However, grounding is still a major challenge. The best grounding strategy still has a 20-25% gap with oracle grounding. Among the various grounding strategies, the best one organically leverages both HTML text and visuals, substantially outperforming image annotation strategies Yang et al. (2023a) by up to 10%.
- In-context learning with large models (both LMMs and LLMs) shows better generalization to unseen websites, while supervised fine-tuning still has an edge on websites seen during training.
- There is a non-negligible discrepancy between online and offline evaluation because there can often be multiple viable plans for completing the same task. Online evaluation is more indicative of a model's true performance.

Figure 2: An example of the element grounding process for a single action during completing the given task with three different methods. In this action step, the model needs to click the "Find Your Truck" button to perform a search. For grounding with textual choices, some element candidates represented with HTML text are given, the model is required to generate the choice index of the target element. For image annotation, bounding boxes and index labels are added to the image. The model is required to generate the label on the bottom-left of the target element. For grounding with element attributes, the model needs to predict the text and type of the target element.

2 SeeAct

In this section, we first explain the problem formulation of web agents and then introduce SeeAct, a generalist web agent based on LMMs. Specifically, given a web-based task (e.g., “Rent a truck with the lowest rate” in the car rental website), we examine two essential capabilities of LMMs as a generalist web agent: (i) **Action Generation** to produce an action description at each step (e.g., “Move the cursor over the ‘Find Your Truck’ button and perform a click”) towards completing the task, and (ii) **Element Grounding** to identify an HTML element (e.g., “[button] Find Your Truck”) at the current step on the webpage.

2.1 Formulation

Given a website $\mathcal{S}$ (*e.g.,* a car rental website) and a task T (*e.g., “Rent a truck with the lowest rate”*), the web agent should generate a sequence of executable actions $A = [\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_n]$ to complete the task. Specifically, at time step t, the agent should generate an action $\boldsymbol{a}_t$ based on the current environment observation s_t, the previous actions $\{\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_{t-1}\}$, and the task T:

$$\boldsymbol{a}_t = \pi(s_t, T, \{\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_{t-1}\})$$

The environment observation s_t comprises an HTML document h_t and a screenshot image i_t. LLMs can only be grounded on the HTML document, while LMMs can be grounded on both the HTML document and the screenshot image. The website status is updated accordingly after each action:

$$s_{t+1} = \mathcal{S}(\boldsymbol{a}_t) = \{h_{t+1}, i_{t+1}\}$$

For simplicity, in subsequent step-wise formulations, the time step notation t is omitted. An action $\boldsymbol{a}$ corresponds to a browser event provided by the website environment. Therefore, we formulate an action as a triplet of three necessary variables for a browser event (e, o, v). $e \in \mathcal{E}$ identifies the target webpage element to operate on, such as the "Find Your Truck" button in Figure 2. $\mathcal{E}$ represents the set of webpage elements within the environment $\mathcal{S}$. The operation $o \in \mathcal{O}$ is the action to be performed on the target element, with $\mathcal{O}$ encompassing all possible operations in $\mathcal{S}$ (e.g., `Click`, `Type`). The variable v denotes the additional value needed for a certain operation (e.g., the date *12/10/2023* for a `Type` operation).

However, agents based on LLMs or LMMs are typically unable to directly generate the three variables (e, o, v) required for a browser event. Instead, they generate a textual description of the intended action $\tilde{\boldsymbol{a}}$, containing information about these variables as $(\tilde{e}, \tilde{o}, \tilde{v})$. This process is referred to as **Action Generation**. To interact with the environment, a further step is required to convert $\tilde{\boldsymbol{a}}$ into $\boldsymbol{a}$, which we refer to as **Action Grounding**.

2.2 Action Generation

We explicitly instruct GPT-4V to imitate humans browsing a webpage and analyze the task, webpage, and previous actions. It is asked to generate an action description $\tilde{a}$ based on its analysis and reasoning. We take the screenshot image i as the visual context without utilizing the HTML document h for action generation.

2.3 Action Grounding

Despite the capability of GPT-4V in identifying and describing the next action to complete the given task in natural language, it is still challenging to convert the action description $\tilde{\boldsymbol{a}}$ into an executable action $\boldsymbol{a}$ within the environment. Deriving operation type o and value v from the action description $\tilde{\boldsymbol{a}}$ can be solved through string parsing reasonably well. The key challenge is to identify the target element e from the generated $\tilde{e}$, which we refer to as **Element Grounding**. To address this challenge, we explore three approaches using different types of information: Grounding via Element Attributes, Grounding via Textual Choices, and Grounding via Image Annotation, as depicted in Figure 2. The prompting details of action generation and grounding are included in Appendix F.

Grounding via Element Attributes. This approach involves prompting the model to generate as detailed attributes of the target element as possible, thereby providing more information to precisely match with the target HTML element. Specifically, we prompt the model to not only describe the element e, but also specify the target element's type and the textual content in $\tilde{e}$. For example, as illustrated in Figure 2, the model would generate element text as "Find Your Truck" and identify its type as a "BUTTON." Following this, a heuristic search is performed across the DOM elements, using the element text and type to locate matching elements. In cases where a single match is found, it is automatically selected. Otherwise, when multiple matches arise, the model is further prompted to select the final selection.

Grounding via Textual Choices. The above approach demands precise and sufficient attribute descriptions from GPT-4V and accurate matching by the heuristic search, which can be highly demanding. For instance, many elements may have no textual content or have textual information in a nearby element instead of itself. Alternatively, we provide the model with textual representations of elements as choices to facilitate grounding, which has already been proven effective in MindAct Deng et al. (2023). Specifically, MindAct utilizes a ranking model to select top-k candidate elements $(e_1, e_2, ..., e_k)$ with a pretrained cross-encoder. Each candidate element is represented as a choice in a multi-choice question with its HTML text, as illustrated in Figure 2. After generating the action description $\tilde{\boldsymbol{a}}$, the model is further asked a multi-choice question to choose its intended element from the given multiple choices (including a 'none' option).

Grounding via Image Annotation. Textual representations alone are sometimes insufficient to distinguish similar or identical elements, as illustrated in Appendix I. Therefore, in this approach, we propose to overlay a bounding box for each candidate element e selected by the ranker as well as a label around the bounding box[2] with a label assignment method to avoid overlapping between markups. The model is expected to generate the label corresponding to the target element.

Oracle Action Grounding. Ideally, the action description $\tilde{a}$ must encompass all necessary details to precisely identify each variable (e, o, v) of the action triplet. To assess the performance of action generation, an oracle grounding method, which ensures the variables be identified as long as they are mentioned in the action description, is desired. Here we approximate the oracle grounding method by asking human annotators to identify the model's intended actions.

3 Experiments

3.1 Dataset

We evaluate our methods on Mind2Web Deng et al. (2023), a comprehensive dataset encompassing over 2000 complex web tasks with annotated actions. This dataset spans 137 websites across 31 low-level domains, categorized into 12 high-level domains. It supports three primary operations:

[2]We use the Supervision library for image annotation: `https://supervision.roboflow.com/`

Table 1: Dataset statistics. The average # of visual tokens is based on OpenAI visual token calculator.

Split	# Tasks	# Domains	# Websites	Avg # Actions	Avg # Visual Tokens	Avg # HTML	
						Elements	Tokens
Train	1009	31	73	7.7	4240	602	128 827
Cross-Domain	750	13	54	5.9	4404	502	92 200
Cross-Task	177	17	64	7.6	4172	607	123 274
Cross-Website	142	9	10	7.2	4653	612	114 358

`Click`, `Type`, and `Select`, with `Hover` and `Press Enter` operations integrated into `Click` to avoid ambiguity. The dataset's test sets aim to measure the generalization of web agents across different tasks, websites, and domains. Specifically, the Cross-Task setting focuses on evaluating agents on tasks that are new to the training data but within included domains and websites. The Cross-Website setting evaluates agents with tasks across 10 new websites for each of the top-level domains in the training data. The Cross-Domain setting assesses agent performance on tasks in two top-level domains held out from the training data.

We align each HTML document in the dataset with its corresponding webpage screenshot image from the MIND2WEB raw dump, which undergoes human verification to confirm correct rendering and element visibility. Mind2Web screenshot images might have some potential issues, including wrong capture time stamp, and wrong rendering, which can make it hard to make decisions, even for humans. To remove these wrong images, we conduct manual filtering and results in the following dataset. Detailed statistics of these subsets are presented in Table 1. We will release this *cleaned multimodal Mind2Web* dataset for future research.

3.2 METHODS

SeeAct. In grounding via image annotation and textual choices, we first employ the DeBERTa-base cross-encoder from MindAct Deng et al. (2023) to rank the top 50 elements for better comparison with its text-only counterparts. Then, we cluster elements into groups of 17 options for inference. In grounding via element attributes, no candidate element is provided. We leverage GPT-4V API for all three methods and Gemini Pro Vision Anil et al. (2023), and LLaVA-1.5 Liu et al. (2023a;c) for the best-performing grounding method.

MindAct. To compare with SEEACT, we also implement methods based on text-only LLMs and BLIP-2 Li et al. (2023) following the two-stage strategy of MindAct Deng et al. (2023). Firstly, we employ the ranker above to pick the top 50 elements. Subsequently, the action generation problem is formulated as a multi-choice question answering problem, with the candidate elements as options, including a "None" option if the target element is absent. During inference, elements are clustered into groups of 5 elements, with iterative refinement, until a single choice is made or all options are discarded. We evaluate supervised fine-tuning (SFT) methods using FLAN-T5 Chung et al. (2022) and BLIP-2-T5 and in-context learning (ICL) methods using GPT-3.5, GPT-4.

Pixel-Level Grounding. LMMs can generate target element coordinates in the image via training on datasets augmented with object coordinates, especially for open-sourced models Hong et al. (2023); Cheng et al. (2024); You et al. (2023). We choose CogAgent Hong et al. (2023) as a representative model for the experiment. Details of each method can be found in Appendix B.

3.3 OFFLINE EVALUATION

We adopt the evaluation metrics utilized in MIND2WEB. **Element Accuracy** (Ele. Acc) compares the predicted element with the ground-truth elements. **Operation F1** (Op. F1) calculates the token-level F1 score for the predicted operation comprised of action and input value. **Step Success Rate** (Step SR) measures the success of each action step. A step is successful only if the selected element and the predicted operation are correct. We report macro averages across tasks for these step-wise metrics. **Success Rate** (SR) measures the success of an entire task. A task is regarded successful only if all steps have succeeded. This metric is stringent without allowing the model the space for exploration and error correction. Therefore, we focus on the first three metrics for offline evaluation and further conduct online evaluation on live websites for better evaluation on the whole task success rate.

Table 2: Performance of different models. All models under SEEACT utilize 'Choices" for grounding. Methods with * mark are conducted on a subset with 30 tasks for each task split.

Model	Cross-Task			Cross-Website			Cross-Domain		
	Ele. Acc	Op. F1	Step SR	Ele. Acc	Op. F1	Step SR	Ele. Acc	Op. F1	Step SR
Supervised Fine-Tuning									
FLAN-T5-XL	**57.1**	75.7	**53.5**	**43.8**	67.7	**41.1**	**41.7**	65.8	**39.2**
BLIP-2-T5-XL	50.1	**77.0**	47.0	39.4	**69.3**	37.0	41.5	**69.4**	39.1
In-Context Learning									
GPT-3.5*	19.4	59.8	16.8	14.9	56.5	14.1	25.5	57.9	24.2
GPT-4*	40.2	63.4	31.7	27.4	61.0	27.0	36.2	61.9	29.7
COGAGENT	22.4	53.0	17.6	18.4	42.2	13.4	20.2	42.1	15.0
SEEACT									
– LLAVA-1.5	9.7	65.6	8.1	9.1	60.8	7.5	10.7	63.7	8.5
– GEMINI PRO	21.5	67.7	19.6	17.1	61.3	15.4	20.7	64.1	18.0
– GPT-4V	**46.4**	**73.4**	**40.2**	**38.0**	**67.8**	**32.4**	**41.8**	**69.4**	**36.3**
– Oracle*	66.4	79.2	61.9	69.5	78.9	65.0	72.8	73.6	62.1

Table 3: Step success rate (%) of GPT-4V on a subset of 30 tasks for each task split with different grounding methods. "Attributes", "Choices", "Annotation", and "Oracle" refer to element grounding via Element Attributes, Textual Choices, Image Annotation, and Human Annotation, respectively, as described in subsection 2.3.

Grounding	Cross-Task	Cross-Website	Cross-Domain
Attributes	4.7	9.7	15.3
Annotation	22.8	18.7	24.3
Choices	35.4	38.1	43.2
Oracle	61.9	65.0	62.1

3.4 ONLINE EVALUATION

We develop a new online evaluation tool using Playwright[3] to evaluate web agents on live websites (instead of cached websites in offline evaluation). Our tool can efficiently tunnel multimodal inputs from the browser to the agent and convert the predicted action (e, o, v) into a browser event for execution. To adhere to ethical standards, our experiments are restricted to non-login tasks in compliance with user agreements, and we closely monitor agent activities during online evaluation to prevent any actions that have potentially harmful impacts, like placing an order or modifying the user profile. For a fair comparison between offline and online evaluations, we only re-write time-sensitive tasks to ensure they are still valid when the evaluation is conducted. For instance, we update the dates for flight-related tasks. Finally, we conduct the online evaluation on the same subset of total 90 tasks from the three test splits and report the mean performance across these tasks.

4 RESULTS AND ANALYSIS

4.1 OFFLINE EVALUATION RESULTS

GPT-4V can be a Generalist Web Agent with Oracle Action Grounding. Given an effective action grounding method, GPT-4V has the potential to serve as a generalist web agent. Specifically, as described in subsection 2.3, we provide GPT-4V with an oracle action grounding method ($\text{SEEACT}_{\text{Oracle}}$) through human annotation, the model achieves a step success rate of 61.9%, 65.0%, and 62.1% across three test splits, respectively. As shown in Table 2, this method substantially outperforms other models under all metrics across three test splits. Specifically, it achieves a 8.4% step success rate improvement over the second-best method in the Cross Task setting. The performance advantage is more pronounced under the Cross-Website and Cross-Domain settings, where it

[3]https://playwright.dev/

Table 4: Whole task success rate (%) under both offline and online evaluation. Offline_0 and Offline_1 refer to no tolerance for error at any step and allowing for error at one step, respectively.

	Offline_0	Offline_1	Online
FLAN-T5-XL	4.4	24.4	8.9
GPT-4	1.1	12.2	13.3
$\text{SeeAct}_{\text{Choice}}$	3.3	12.2	37.8
$\text{SeeAct}_{\text{Oracle}}$	13.3	27.8	51.1

leads by 23.9% and 22.9% step success rates, demonstrating its generality compared with supervised fine-tuning. This observation is further corroborated within the online evaluation (Table 4).

Element Grounding Method Comparison. However, there is a noticeable gap between oracle grounding and all three proposed grounding methods, as shown in Table 3. This demonstrates that grounding, especially element grounding, is a major bottleneck. Element grounding via textual choice ($\text{SeeAct}_{\text{Choice}}$) demonstrates the best performance under all metrics across all settings, comparable to supervised fine-tuning and showing a substantial improvement over text-only LLMs.

Grounding via image annotation ($\text{SeeAct}_{\text{Annotation}}$) offers an intuitive approach and shows promising results in recent work that focuses on object- or scene-centric images Yang et al. (2023a). However, we find that on complex images with rich semantic and spatial relationships like webpage screenshots, severe hallucination is observed from GPT-4V. Specifically, it often fails to correctly map its generated element description (which is often correct according to oracle grounding) to the right bounding box and index label in the image, leading to a low element accuracy. This limitation primarily arises from GPT-4V's weakness in understanding image details and relative spatial location, a topic that we will further delve into in Appendix G. Markup types, like label type and location, can influence model performance Yang et al. (2023a); Yan et al. (2023), we leverage a number label on the bottom left of the bounding box as it shows the best performance in the ablation study in Appendix C. Grounding via element attributes ($\text{SeeAct}_{\text{Attribute}}$) also demonstrates inferior performance. This method's effectiveness is primarily limited by its heuristic-based element localization strategy, which depends on textual and locality characteristics. This becomes problematic as not all webpage elements contain text, and sometimes the relevant text is associated with a nearby but distinct element.

LMMs vs. LLMs. The $\text{SeeAct}_{\text{Choices}}$ model demonstrates a substantial performance advantage over the text-only GPT-4 under all three metrics across all three test splits. Specifically, it outperforms GPT-4 in step success rate of 8.5%, 5.4%, and 6.6% on three settings, respectively. Interestingly, fine-tuned BLIP-2-T5 does not show a noticeable gain over FLAN-T5, despite having additional visual input. Several factors may contribute to this. First, the CLIP model used as the image encoder may not be sufficiently adept at image details, as explored by Shen et al. (2021). This limitation is particularly relevant for our web navigation task, which demands a high level of image detail comprehension. Second, BLIP-2-T5 utilizes an off-the-shelf CLIP model that may not be optimal for webpage screenshots. Finally, although the screenshots in the test splits are error-free, some of the examples in the training set might contain issues such as rendering failures or inaccuracies in the timing of screenshot capture by annotators.

SFT vs. ICL. We compare SFT and ICL methods to offer insights for developing web agents in different scenarios. ICL (with SeeAct) demonstrates consistent and robust performance across three test splits. ICL is particularly advantageous in scenarios lacking annotations or requiring strong generalization capabilities for new domains and websites. As grounding methods improve towards oracle grounding, ICL is poised to show even stronger performance. On the other hand, SFT methods show better generalization across tasks on websites already seen during training. Considering the high cost of data annotation for web agents and the billions of websites on the Internet, ICL offers a more compelling solution for generalist web agents. However, if one only needs to develop a strong web agent for a certain website, SFT is still a competitive solution.

4.2 Online Evaluation Results

In online evaluation, we pair a web agent with a human annotator, where the human was tasked to monitor agent actions that may change real-world states and determine whether each task was successfully completed. For comparative analysis, we include success rates from offline evaluation, denoted as Offline_0 (allowing zero wrong action) and Offline_1 (allowing one wrong action). Table 4 shows that the whole task success rate in online evaluation substantially exceeds that of offline evaluation (Offline_0). This finding suggests that the whole task success rate is likely underestimated in the offline evaluation due to the variability in actions and plans. In other words, there may be multiple viable plans for a task, but the reference plan in offline evaluation only captures one of them. Across all three settings, $\text{SEEACT}_{\text{Choice}}$ outperforms both GPT-4 and FLAN-T5-XL by a large margin of over 20% whole task success rate. Using oracle grounding further improves the performance substantially, reaching a remarkable whole task success rate of 51.1%. Although GPT-4 shows much worse performance than FLAN-T5-XL in step success rate under offline evaluation (Table 2), it outperforms FLAN-T5-XL by 3.3% whole task success rate in the online evaluation. These results further confirm the potential of large models for generalist web agents compared with fine-tuned small models.

4.3 ANALYSIS

Online Success Rate by Task Difficulty. We investigate the performance of web agents on tasks across different difficulty levels. We estimate the task difficulty based on the number of actions taken by annotators during action trace annotation. As shown in Appendix E, the whole task success rate is negatively correlated with the number of actions—it decreases as the number of actions increases across all four methods. $\text{SEEACT}_{\text{Oracle}}$ consistently outperforms other methods across all difficulty levels. Interestingly, the gap between $\text{SEEACT}_{\text{Oracle}}$ and $\text{SEEACT}_{\text{Choice}}$ enlarges on longer-horizon tasks. This is understandable because grounding errors cascade to later steps; nonetheless, it further shows the challenge of grounding for GPT-4V and the need for better grounding methods.

Error Analysis in Grounding via Image Annotation. Set-of-mark prompting Yang et al. (2023a) uses a similar method as grounding via image annotation and has been shown effective on object- or scene-centric images Lin et al. (2014); Plummer et al. (2015); Zhou et al. (2017). However, this grounding method is suboptimal on webpage screenshot images that are complex and contain rich semantic and spatial relationships. To analyze the reasons behind the failures, we randomly sample 100 action predictions with correct action generation but wrong grounding results. We observes major types of errors as : (1) Making up bounding box & label; (2) Failure to link bounding boxes with the correct labels. Illustrative examples are included in Appendix G.

Our analysis reveals that 54% of the errors can be attributed to GPT-4V's tendency of visual illusion Guan et al. (2023), where the model misinterprets and fabricates content over the image. Specifically, the target element described in action generation does not have a bounding box or a label on the bottom-left, where the model is supposed to generate "NA". However, the model falsely assumes the presence of a bounding box and makes up a label as the answer. Another 46% of errors are caused by GPT-4V's limitation in recognizing the relative position within an image. Specifically, the model is capable of identifying the target element within the bounding box. However, it struggles to correctly link the bounding box with its corresponding label.

4.4 CASE STUDY

GPT-4V exhibits promising capabilities, ranging from speculative planning, webpage content reasoning, and error correction to surpassing the limitations of superficial textual similarity matching inherent in fine-tuned, text-only models.

World Knowledge. GPT-4V demonstrates substantial advantages in tasks requiring certain knowledge over fine-tuned smaller models. As shown in Appendix J, it is able to identify the IATA code of the airport in Los Cabos as SJD. In contrast, smaller models are typically weaker at knowledge-intensive tasks and are likely to lose knowledge during fine-tuning due to catastrophic forgetting.

World Model (for Websites). GPT-4V exhibits the potential of a "world model" for websites. As shown in Appendix H, GPT-4V can predict the state transitions on a website (e.g., what would happen if I clicked this button). Based on its awareness of website state transitions, it can conduct speculative planning involving a sequence of subsequent actions in the future to complete the task.

Error Correction Awareness. GPT-4V also exhibits the awareness of error correction in the previous actions. In the example in Appendix K, it realizes that the mobile phone number is invalid due to the wrong format and generates the description of the action to correct this error. This highlights the model's potential for adaptation in online settings, where actions may not always follow pre-defined, ideal paths as in offline evaluations. This capability paves the way for adding robustness and reasonable dynamic planning.

5 RELATED WORK

Web Agent. Many works have focused on improving web agents relying on the HTML document Deng et al. (2023); Gur et al. (2023; 2022; 2023); Kim et al. (2023); Sridhar et al. (2023). However, a raw HTML document is often massive making it infeasible or cost-prohibitively to feed into LLMs. MindAct Deng et al. (2023) instead employs a small language model to rank each HTML element and selectively consider top elements as the context. WebAgent Gur et al. (2023) proposes an enhanced planning strategy by summarizing the HTML documents and decomposing the instruction into multiple sub-instructions. Another stream considers visual information for web agents Shaw et al. (2023); Furuta et al. (2023); Hong et al. (2023). Pix2Act Shaw et al. (2023) leverages Pix2Struct Lee et al. (2022) to parse screenshot images into simplified HTML to complete GUI-based tasks Shaw et al. (2023); Liu et al. (2018); Shi et al. (2017); Mazumder & Riva (2020); Yao et al. (2022). WebGUM Furuta et al. (2023) and CogAgent Hong et al. (2023) pre-train an LMM with massive screenshot-HTML data to enhance its decision-making on real-world web. However, generalizing to various web environments remains a challenge for existing models. Thus, SEEACT explores more powerful LMMs, like GPT-4V and Gemini, to demonstrate their potential as generalist web agents with comprehensive online and offline evaluation and analysis. In a concurrent work Yan et al. (2023), GPT-4V exhibits strong performance on mobile UI understanding, which is less complex than the desktop websites we study.

Large Multimodal Models. GPT-4V OpenAI (2023) and Gemini Anil et al. (2023) represent significant progress in LMMs. Several studies Akter et al. (2023); OpenAI (2023); Yang et al. (2023c); Zhang et al. (2023); Yang et al. (2023a); Yan et al. (2023) have highlighted their remarkable multimodal capabilities. Their performance on a series of benchmarks Kazemzadeh et al. (2014); Goyal et al. (2016); Hendrycks et al. (2020); Saikh et al. (2022); Lu et al. (2022); Zhong et al. (2023); Yue et al. (2023) also showcases remarkable capabilities on vision-and-language understanding and reasoning. Although open-sourced models still exhibit a performance gap with GPT-4V, they have the advantages of controllability and ease of fine-tuning for various applications, like webpage understanding Hong et al. (2023) and visual referring and grounding You et al. (2023).

Visual Grounding. Despite LMMs having achieved remarkable vision-language understanding capabilities, they still face challenges in fine-grained visual grounding. Various visual prompting Shtedritski et al. (2023); Yang et al. (2023b;c;a); Yan et al. (2023) methods have been proposed to augment GPT-4V's image detail grounding ability by overlaying visual marks onto the image, including numbers, alphabets, masks, and bounding boxes. It's also effective to fine-tune LMMs on images annotated with different location representations, including textual location tokens Peng et al. (2023), masks Zhao et al. (2023), spatial coordinates tokens Chen et al. (2023); You et al. (2023).

6 CONCLUSION

In this work, we developed SEEACT, a generalist web agent that harnesses the power of large multimodal models (LMMs) like GPT-4V to integrate visual understanding and acting on the web. We showed that LMMs present a great promise for generalist web agents, with a success rate of 50% on live websites given an oracle grounding method. GPT-4V also exhibits impressive capabilities, such as error correction and speculative planning. However, fine-grained visual grounding is still a major challenge. The most effective grounding strategies we explored in this paper still exhibit a 20-25% performance gap compared to oracle grounding. Future work should better leverage the unique properties of the Web, *e.g.*, the known correspondence between HTML and visual elements, for improving grounding and reducing hallucinations from LMMs. Furthermore, we show a significant discrepancy between online and offline evaluations, emphasizing the importance of online evaluation for an accurate assessment of a model's capabilities. This discrepancy is largely due to the variability in potential plans for completing the same task, pointing to the dynamic nature of web interactions.

REFERENCES

Syeda Nahida Akter, Zichun Yu, Aashiq Muhamed, Tianyue Ou, Alex Bauerle, Ángel Alexander Cabrera, Krish Dholakia, Chenyan Xiong, and Graham Neubig. An in-depth look at gemini's language abilities. 2023. URL https://api.semanticscholar.org/CorpusID:266359502.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *ArXiv*, abs/2204.14198, 2022. URL https://api.semanticscholar.org/CorpusID:248476411.

Gemini Team Google Rohan Anil, Sebastian Borgeaud, and Yonghui Wu et al. Gemini: A family of highly capable multimodal models. 2023. URL https://api.semanticscholar.org/CorpusID:266361876.

Khyathi Raghavi Chandu, Yonatan Bisk, and Alan W Black. Grounding 'grounding'in nlp. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 4283–4305, 2021.

Ke Chen, Zhao Zhang, Weili Zeng, Richong Zhang, Feng Zhu, and Rui Zhao. Shikra: Unleashing multimodal llm's referential dialogue magic. *ArXiv*, abs/2306.15195, 2023. URL https://api.semanticscholar.org/CorpusID:259262082.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Yantao Li, Jianbing Zhang, and Zhiyong Wu. Seeclick: Harnessing gui grounding for advanced visual gui agents. 2024. URL https://api.semanticscholar.org/CorpusID:267069082.

Hyung Won Chung, Le Hou, S. Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Wei Yu, Vincent Zhao, Yanping Huang, Andrew M. Dai, Hongkun Yu, Slav Petrov, Ed Huai hsin Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. Scaling instruction-finetuned language models. *ArXiv*, abs/2210.11416, 2022. URL https://api.semanticscholar.org/CorpusID:253018554.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *arXiv preprint arXiv:2306.06070*, 2023.

Hiroki Furuta, Ofir Nachum, Kuang-Huei Lee, Yutaka Matsuo, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models. *ArXiv*, abs/2305.11854, 2023. URL https://api.semanticscholar.org/CorpusID:258823350.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. *International Journal of Computer Vision*, 127:398 – 414, 2016. URL https://api.semanticscholar.org/CorpusID:8081284.

Yu Gu, Xiang Deng, and Yu Su. Don't generate, discriminate: A proposal for grounding language models to real-world environments. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4928–4949, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.270. URL https://aclanthology.org/2023.acl-long.270.

Tianrui Guan, Fuxiao Liu, Xiyang Wu, Ruiqi Xian, Zongxia Li, Xiaoyu Liu, Xijun Wang, Lichang Chen, Furong Huang, Yaser Yacoob, Dinesh Manocha, and Tianyi Zhou. Hallusionbench: An advanced diagnostic suite for entangled language hallucination&visual illusion in large vision-language models. 2023. URL https://api.semanticscholar.org/CorpusID:265499116.

Izzeddin Gur, Ofir Nachum, Yingjie Miao, Mustafa Safdari, Austin Huang, Aakanksha Chowdhery, Sharan Narang, Noah Fiedel, and Aleksandra Faust. Understanding html with large language models. In *Conference on Empirical Methods in Natural Language Processing*, 2022. URL https://api.semanticscholar.org/CorpusID:252780086.

Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. A real-world webagent with planning, long context understanding, and program synthesis. *ArXiv*, abs/2307.12856, 2023. URL https://api.semanticscholar.org/CorpusID:260126067.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Xiaodong Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *ArXiv*, abs/2009.03300, 2020. URL https://api.semanticscholar.org/CorpusID:221516475.

Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents. 2023. URL https://api.semanticscholar.org/CorpusID:266210390.

Sahar Kazemzadeh, Vicente Ordonez, Marc andre Matten, and Tamara L. Berg. Referitgame: Referring to objects in photographs of natural scenes. In *Conference on Empirical Methods in Natural Language Processing*, 2014. URL https://api.semanticscholar.org/CorpusID:6308361.

Geunwoo Kim, Pierre Baldi, and Stephen Marcus McAleer. Language models can solve computer tasks. *ArXiv*, abs/2303.17491, 2023. URL https://api.semanticscholar.org/CorpusID:257834038.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. 2024. URL https://api.semanticscholar.org/CorpusID:267199749.

Kenton Lee, Mandar Joshi, Iulia Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. *ArXiv*, abs/2210.03347, 2022. URL https://api.semanticscholar.org/CorpusID:252762394.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *ArXiv*, abs/2301.12597, 2023. URL https://api.semanticscholar.org/CorpusID:256390509.

Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014. URL https://api.semanticscholar.org/CorpusID:14113767.

Evan Zheran Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. In *International Conference on Learning Representations (ICLR)*, 2018. URL https://arxiv.org/abs/1802.08802.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning, 2023a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *ArXiv*, abs/2304.08485, 2023b. URL https://api.semanticscholar.org/CorpusID:258179774.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023c.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Yuxian Gu, Hangliang Ding, Kai Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Shengqi Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. Agentbench: Evaluating llms as agents. *ArXiv*, abs/2308.03688, 2023d. URL https://api.semanticscholar.org/CorpusID:260682249.

Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and A. Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *ArXiv*, abs/2209.09513, 2022. URL `https://api.semanticscholar.org/CorpusID:252383606`.

S. Mazumder and Oriana Riva. Flin: A flexible natural language interface for web navigation. *ArXiv*, abs/2010.12844, 2020. URL `https://api.semanticscholar.org/CorpusID:225067907`.

OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023. URL `https://api.semanticscholar.org/CorpusID:257532815`.

Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu Wei. Kosmos-2: Grounding multimodal large language models to the world. *ArXiv*, abs/2306.14824, 2023. URL `https://api.semanticscholar.org/CorpusID:259262263`.

Bryan A. Plummer, Liwei Wang, Christopher M. Cervantes, Juan C. Caicedo, J. Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. *International Journal of Computer Vision*, 123:74 – 93, 2015. URL `https://api.semanticscholar.org/CorpusID:6941275`.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Tanik Saikh, Tirthankar Ghosal, Amish Mittal, Asif Ekbal, and Pushpak Bhattacharyya. Scienceqa: a novel resource for question answering on scholarly articles. *International Journal on Digital Libraries*, 23:289 – 301, 2022. URL `https://api.semanticscholar.org/CorpusID:250729995`.

Peter Shaw, Mandar Joshi, James Cohan, Jonathan Berant, Panupong Pasupat, Hexiang Hu, Urvashi Khandelwal, Kenton Lee, and Kristina Toutanova. From pixels to ui actions: Learning to follow instructions via graphical user interfaces. *ArXiv*, abs/2306.00245, 2023. URL `https://api.semanticscholar.org/CorpusID:258999511`.

Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How much can clip benefit vision-and-language tasks? *arXiv preprint arXiv:2107.06383*, 2021.

Tianlin Shi, Andrej Karpathy, Linxi (Jim) Fan, Josefa Z. Hernández, and Percy Liang. World of bits: An open-domain platform for web-based agents. In *International Conference on Machine Learning*, 2017. URL `https://api.semanticscholar.org/CorpusID:34953552`.

Aleksandar Shtedritski, C. Rupprecht, and Andrea Vedaldi. What does clip know about a red circle? visual prompt engineering for vlms. *ArXiv*, abs/2304.06712, 2023. URL `https://api.semanticscholar.org/CorpusID:258108138`.

Abishek Sridhar, Robert Lo, Frank F. Xu, Hao Zhu, and Shuyan Zhou. Hierarchical prompting assists large language model on web navigation. *ArXiv*, abs/2305.14257, 2023. URL `https://api.semanticscholar.org/CorpusID:258841249`.

An Yan, Zhengyuan Yang, Wanrong Zhu, Kevin Lin, Linjie Li, Jianfeng Wang, Jianwei Yang, Yiwu Zhong, Julian McAuley, Jianfeng Gao, Zicheng Liu, and Lijuan Wang. Gpt-4v in wonderland: Large multimodal models for zero-shot smartphone gui navigation. 2023. URL `https://api.semanticscholar.org/CorpusID:265149992`.

Jianwei Yang, Hao Zhang, Feng Li, Xueyan Zou, Chun yue Li, and Jianfeng Gao. Set-of-mark prompting unleashes extraordinary visual grounding in gpt-4v. *ArXiv*, abs/2310.11441, 2023a. URL `https://api.semanticscholar.org/CorpusID:264172201`.

Lingfeng Yang, Yue Wang, Xiang Li, Xinlong Wang, and Jian Yang. Fine-grained visual prompting. *ArXiv*, abs/2306.04356, 2023b. URL `https://api.semanticscholar.org/CorpusID:259096008`.

Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v(ision). *ArXiv*, abs/2309.17421, 2023c. URL `https://api.semanticscholar.org/CorpusID:263310951`.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *ArXiv*, abs/2207.01206, 2022. URL `https://api.semanticscholar.org/CorpusID:250264533`.

Haoxuan You, Haotian Zhang, Zhe Gan, Xianzhi Du, Bowen Zhang, Zirui Wang, Liangliang Cao, Shih-Fu Chang, and Yinfei Yang. Ferret: Refer and ground anything anywhere at any granularity. *ArXiv*, abs/2310.07704, 2023. URL `https://api.semanticscholar.org/CorpusID:263834718`.

Xiang Yue, Yuansheng Ni, Kai Zhang, Tianyu Zheng, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. Mmmu: A massive multi-discipline multimodal understanding and reasoning benchmark for expert agi. *ArXiv*, abs/2311.16502, 2023. URL `https://api.semanticscholar.org/CorpusID:265466525`.

Xinlu Zhang, Yujie Lu, Weizhi Wang, An Yan, Jun Yan, Lianke Qin, Heng Wang, Xifeng Yan, William Yang Wang, and Linda Ruth Petzold. Gpt-4v(ision) as a generalist evaluator for vision-language tasks. *ArXiv*, abs/2311.01361, 2023. URL `https://api.semanticscholar.org/CorpusID:264935635`.

Yang Zhao, Zhijie Lin, Daquan Zhou, Zilong Huang, Jiashi Feng, and Bingyi Kang. Bubogpt: Enabling visual grounding in multi-modal llms. *ArXiv*, abs/2307.08581, 2023. URL `https://api.semanticscholar.org/CorpusID:259937702`.

Wanjun Zhong, Ruixiang Cui, Yiduo Guo, Yaobo Liang, Shuai Lu, Yanlin Wang, Amin Saied Sanosi Saied, Weizhu Chen, and Nan Duan. Agieval: A human-centric benchmark for evaluating foundation models. *ArXiv*, abs/2304.06364, 2023. URL `https://api.semanticscholar.org/CorpusID:258108259`.

Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5122–5130, 2017. URL `https://api.semanticscholar.org/CorpusID:5636055`.

Appendix Table of Content

A IMPACT STATEMENTS

Generalist web agents hold the potential to automate routine web tasks, enhance user experiences, and promote web accessibility, safety concerns related to their real-world deployment are also critical. These concerns span privacy issues, such as access to users' personal profiles, and sensitive operations, such as financial transactions or application form submissions. During the online evaluation, we noticed the possibility for these web agents to generate harmful actions on the web, and we manually validated the safety of all the actions before execution. It is critical for further research to thoroughly assess and mitigate the safety risks associated with web agents, ensuring they are safeguarded against producing and executing harmful actions. The code will also be released solely for research purposes, with the goal of making the web more accessible via language technologies under an OPEN-RAIL License. We are strongly against any potentially harmful use of the data or technology by any party.

B OFFLINE EXPERIMENTS METHOD DETAILS

FLAN-T5. We fine-tune FLAN-T5 using a left-to-right language modeling objective with the target sequence of ground-truth actions in the Mind2Web training data. The fine-tuned FLAN-T5 then serves as the backbone for inference, enabling action generation in the target format for parsing.

BLIP-2-T5. The BLIP-2 model combines a vision encoder and an LLM with a bridging component for modality connection. We jointly fine-tune the LLM and the bridge module on MIND2WEB training data while keeping the vision encoder frozen. For the vision encoder, we leverage the ViT-L/14 pre-trained from CLIP Radford et al. (2021) with an image resolution of 2048. To ensure a fair comparison with the FLAN-T5-based text-only model, we choose FLAN-T5 as the language model and initialize it with the parameters fine-tuned on MIND2WEB.

GPT-3.5 and GPT-4. We also conduct experiments with text-only LLMs, specifically GPT-3.5-turbo-0613 and GPT-4-turbo-1106-preview, using in-context learning in 3-shot settings. We use the same multiple-choice formulation and include three demonstration examples for in-context learning as specified in MindAct.

SeeAct We experiment with GPT-4-vision-preview, Gemini Pro Vision, LLaVA-1.5. Gemini Pro Vision supports only single-turn conversations; therefore, we merge the two turns used in other models for compatibility.

CogAgent We utilize the cogagent-chat-hf checkpoint that hasn't been fine-tuned on Mind2Web for experiments.

C MARKUP TYPE ABLATION STUDY

The markup types might influence model performance as shown in Yang et al. (2023a); Yan et al. (2023); Koh et al. (2024). We first tested grounding via image annotation through different types of text labels of numerical value, single-digit characters, and two-digit characters, at two different positions to choose a relatively better markup type. The results are shown in Table 5.

Table 5: Grounding via Image Annotation with different markup types and locations. Method with * mark means the annotated image is used in action generation.

Label	Location	Ele. Acc	Op. F1	Step SR
Number	Left	**27.0**	73.7	**24.3**
	Center	23.0	76.4	21.8
Letter-1	Left	19.4	**81.0**	17.2
	Center	19.7	78.8	19.7
Letter-2	Left	19.8	68.3	18.3
	Center	22.4	74.6	22.4
NUMBER*	Left*	26.6	73.9	22.3

D Online Experiment Details

We develop an online evaluation tool using Playwright to load webpages, acquire textual representation of interactive elements, and perform operations generated by web agents. We manually monitor each step of the model and assess whether it finishes the tasks. Attempts to log in, make final submissions, or perform other potentially harmful actions are prohibited to avoid negative consequences.

MindAct. We adhere strictly to the original settings in MindAct-FLAN-T5 and MindAct-GPT-4, as in offline experiment. For consistency with the MindAct framework and models, we use the scripts from Mind2Web for processing webpages, elements, and generating options, employing the same action space: `Click`, `Type`, and `Select`.

SeeAct$_{\text{Oracle}}$. In the oracle setting, we manually implement the model's intended actions. The action history is automatically generated by the model, with an added requirement to summarize actions in the "Element", "Operation", "Value" format in another turn conversation. To avoid overly long screenshots, we use screenshots of current views, and hence allow intentions of scrolling.

SeeAct$_{\text{Choice}}$. For a relatively fair comparison, we still adopt the ranker and top-50 candidate setting, batching them into three option groups as described in offline experiments. We enable `PRESS ENTER` and `TERMINATE` for the model to make confirmation or stop the process.

During our tests, pop-up ads on webpages were manually closed. The MindAct model, not trained on handling pop-up ads, lacks the feature to automatically manage them, potentially causing stalls. In contrast, SeeAct models can proactively suggest closing ads through visual analysis and reasoning.

E Online Success Rate by Task Difficulty

We categorize tasks based on the number of actions to complete, i.e., Easy: 1-4, Medium: 5-9, and Hard: 10-18, with 37, 35, and 18 tasks in each group, respectively. The whole task success rate is shown in Figure 3.

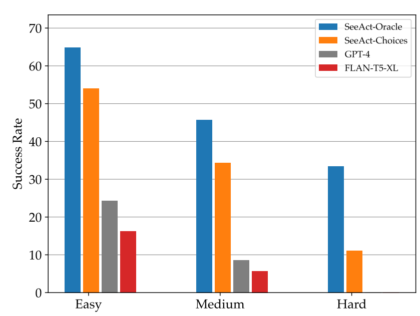


Figure 3: Whole task success rate across task difficulty levels.

F Offline Experiment Prompts

The prompt for action generation is shown in Table 6. For grounding via textual choices, image annotation, and element attributes, the prompts are shown in tables 7 to 9, along with specific tasks and examples in figs. 4 to 8.

Table 6: Prompt for SEEACT Action Generation with LMMs.

System Role	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 an element with the mouse, select an option, or type text with the keyboard. (For your understanding, they are like the click(), select_option() and type() functions in playwright respectively) One next step means one operation within the three.
Action Generation	You are asked to complete the following task: {TASK} 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: (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. (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. You should closely examine the screenshot details to see what steps have been completed by previous actions even though you are given the textual previous actions. Because the textual history may not clearly and sufficiently record some effects of previous actions, you should closely evaluate the status of every part of the webpage to understand what you have done. (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. 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.

G ERROR EXAMPLES FOR GROUNDING VIA IMAGE ANNOTATION

In the method of grounding via image annotation, we observe significant hallucination errors that can be classified into the following categories:

Making up bounding box & label. In our grounding method, if the correct element is absent from the set of candidate elements, the model is anticipated to generate "NA" as the answer. However, as depicted in Figure 10 and Figure 11, the model erroneously claims the element is included within a red bounding box and makes up a wrong index label as the answer.

Failure to link bounding boxes with the correct labels. Another challenge arises in accurately linking bounding boxes to their corresponding index labels. This challenge can be attributed to both LMMs' limitations in understanding relative spatial positions and the complex, dense layout of webpage elements. The model often mistakenly associates the labels of adjacent elements (as illustrated in Figure 12 and Figure 13), rather than accurately predicting the intended index label for the targeted element.

Table 7: Prompt for SEEACT grounding via element attributes. We make a slight modification to enhance action generation and only show the modified part here to save space, as well as the prompts in Table 8 and Table 9.

System Role	Same as Table 6
Action Generation	Slightly modified from Table 6 ... (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. Please also closely examine the screenshot to adequately describe its position relative to nearby elements and its textual or visual content (if it has). If you find multiple elements similar to your target element, use a more precise description to ensure people can distinguish your target element from them through your answer. ...
Format Answer	(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, element type, element text, action and value should be in five separate lines. Format: ELEMENT: Please describe which element you need to operate with. Describe it as detailed as possible, including what it is and where it is. ELEMENT TYPE: Please specify its type from these options: BUTTON, TEXTBOX, SELECTBOX, or LINK. ELEMENT TEXT: Please provide the exact text displayed on the element. Do not invent or modify the text; reproduce it as-is from the screenshot. ACTION: Choose an action from {CLICK, TYPE, SELECT}. VALUE: Provide additional input based on ACTION. The VALUE means: If ACTION == TYPE, specify the text to be typed. If ACTION == SELECT, specify the option to be chosen. If ACTION == CLICK, write "None".

H STRONG CAPABILITY OF PLANNING

GPT-4V shows remarkable understanding and planning capabilities during our experiments. As depicted in Figure 14, the model is capable of understanding the website and generating a full plan for the given task involving multiple low-level tasks. Specifically, GPT-4V could understand reasonably well about the process and the remaining work of the task by its careful examination of the webpage, as shown in Figure 15.

I CHALLENGES IN GROUNDING VIA TEXTUAL CHOICES

Although textual choices achieved the best results among the three grounding approaches, it still suffers from challenges of similar or identical elements which are common in webpages. The model tends to choose the first text choice that seemingly corresponds to its intention. Moreover, this is inevitable, as web pages indeed contain many elements that may even have exact identical HTML information, as the "Schedule" button shown in Figure 16.

Table 8: Prompt for SEEACT grounding via textual choices.

System Role	Same as Table 6
Action Generation	Same as Table 6
Referring Description	(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. From the screenshot, find out where and what each one is on the webpage. 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 [None of the other options match the correct element]. A. [CHOICE A] B. [CHOICE B] ...
Format Answer	(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. ACTION: Choose an action from {CLICK, TYPE, SELECT}. VALUE: Provide additional input based on ACTION. The VALUE means: If ACTION == TYPE, specify the text to be typed. If ACTION == SELECT, specify the option to be chosen. If ACTION == CLICK, write "None".

J KNOWLEDGE AND REASONING REQUIREMENTS

Some tasks require a certain degree of reasoning and knowledge, which may be challenging for fine-tuned models like MindAct. For instance, the task in Figure 17 necessitates the model to know the specific district of Dublin in Virginia. In the task of Figure 18, the model correctly provided the IATA airport code of airports in Indira Gandhi and Los Cabos.

K PATH VARIATION AND AWARENESS OF ERROR CORRECTION

On webpages, multiple paths often exist to accomplish a given task. For instance, varying the execution order of actions within an interchangeable sequence can result in diverse routes to task completion. Additionally, the agent can navigate to different webpages but still accomplish the give tasks. Figure 19 presents a straightforward example where the model chose a more direct route that differs from the ground truth annotated in the dataset.

When running on live website, the agent's previous action histories is likely to be filled with redundant, unnecessary, erroneous, failed operations generated, or merely exploratory attempts by the model, resulting in a final path that deviates significantly from the ground truth. Despite these circumstances, the model can still accomplish the task amidst numerous incorrect explorations. The process of

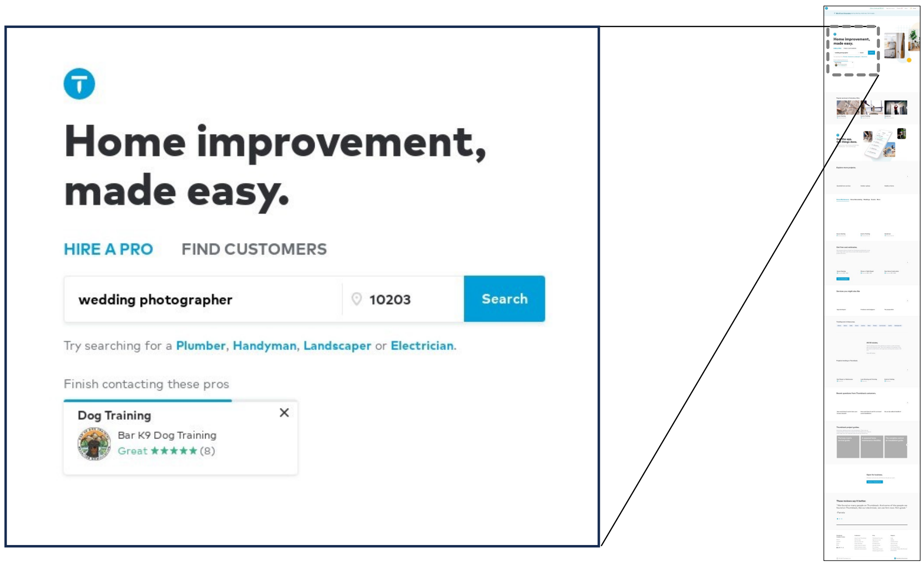
Table 9: Prompt for SEEACT grounding via image annotation.

System Role	Same as Table 6
Action Generation	Same as Table 6
Referring Description	(Reiteration) First, reiterate your next target element, its detailed location, and the corresponding operation. (Verification with the Screenshot) Then, please closely re-examine the screenshot to find whether your target element is marked by a red bounding box and has a white number on a black background at the bottom left corner of the bounding box, which is positioned closely next to the bounding box. If yes, use that number for your final answer. If not, please do not make them up. If it is not marked, please output "NA" as your target element in the following final answer part.
Format Answer	(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 number of your choice. ACTION: Choose an action from {CLICK, TYPE, SELECT}. VALUE: Provide additional input based on ACTION. The VALUE means: If ACTION == TYPE, specify the text to be typed. If ACTION == SELECT, specify the option to be chosen. If ACTION == CLICK, write "None".

exploration and correction requires the model to possess a sense of self-correction. As shown in Figure 20, GPT-4V demonstrates this awareness of correcting errors caused by previous steps.

Example of Grounding via Element Attributes – 1st Turn

Input – 1st turn: Action Generation Prompts
Task: View the profile of a Wedding Photographer near 10203 for a 4 hour wedding on april 13
Previous Actions:
[textbox] Search on Thumbtack -> TYPE: wedding photographer
[textbox] Zip code -> TYPE: 10203



GPT-4V: (Current Webpage Identification)
The current webpage appears to be the homepage of Thumbtack, which is a local services marketplace where users can find professionals for various services, including wedding photography.

(Previous Action Analysis)
The previous actions indicate that the user has typed "wedding photographer" into the search box and entered the zip code "10203" to find a wedding photographer near that location. The intention behind these actions is to search for a wedding photographer in the specified area.

(Screenshot Details Analysis)
The screenshot shows the Thumbtack homepage with the search bar at the top, containing the previously entered search term "wedding photographer" and the zip code "10203." Below the search bar, there are suggestions for different services and popular articles, but these are not relevant to the task at hand. The search button next to the zip code field is visible, indicating that the search has not yet been initiated.

(Next Action Based on Webpage and Analysis)
Based on the analysis, the next logical step is to initiate the search for wedding photographers in the specified zip code area. To do this, the user should click on the "Search" button.

The "Search" button is located to the right of the zip code field at the top of the webpage. It is a rectangular button with rounded corners, and the text "Search" is visible on it. This button is directly adjacent to the zip code field where "10203" has been entered.

The first next action to take is to click the "Search" button to proceed with finding a wedding photographer for the specified requirements.

Figure 4: An example of action generation in grounding via element attributes.

Example of Grounding via Element Attributes – 2nd Turn

Input – 2nd turn: Referring Prompts

GPT-4V: ELEMENT: The "Search" button located to the right of the zip code field at the top of the Thumbtack homepage.

ELEMENT TYPE: BUTTON

ELEMENT TEXT: Search

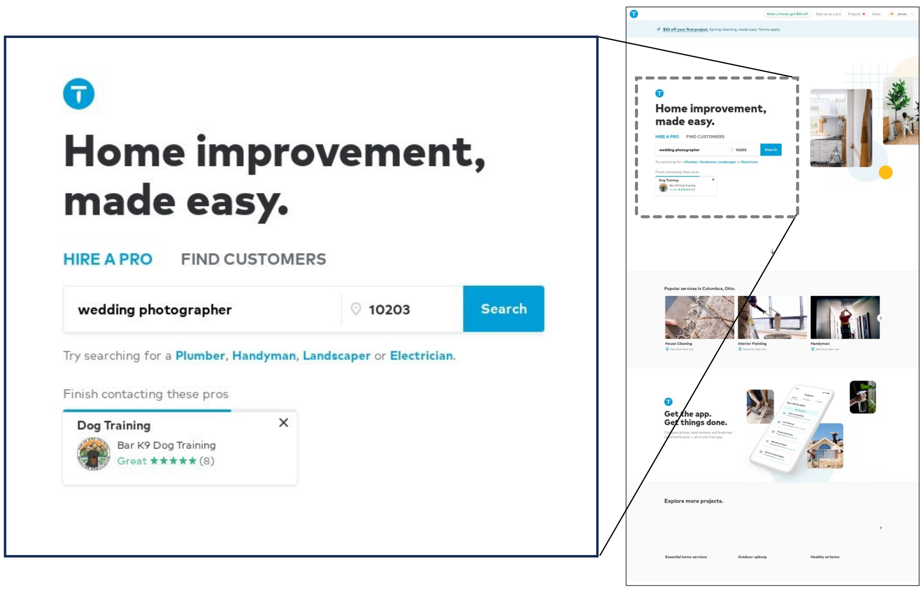
ACTION: CLICK

VALUE: None

Figure 5: An example of grounding via element attributes after action generation.

Example of Grounding via Textual Choices – 1st Turn

Input – 1st turn: Action Generation Prompts
Task: View the profile of a Wedding Photographer near 10203 for a 4 hour wedding on april 13
Previous Actions:
[textbox] Search on Thumbtack -> TYPE: wedding photographer
[textbox] Zip code -> TYPE: 10203



GPT-4V: (Current Webpage Identification)
The current webpage appears to be the homepage of Thumbtack, a platform for hiring local professionals for various services.

(Previous Action Analysis)
The previous actions indicate that the user is looking for a wedding photographer. The user has typed "wedding photographer" into a search box and entered the zip code "10203" into another box, presumably to localize the search to that area.

(Screenshot Details Analysis)
The screenshot shows the Thumbtack homepage with the search fields already populated with "wedding photographer" and "10203." The user has not yet initiated the search, as the search results are not displayed on the screen. The "Search" button is visible next to the zip code field, indicating that the next logical step is to execute the search.

(Next Action Based on Webpage and Analysis)
Based on the analysis, the next action should be to initiate the search for wedding photographers in the specified zip code area. The first next target element is the "Search" button located to the right of the zip code field.

The corresponding operation is to click the "Search" button to proceed with finding a list of wedding photographers near the zip code 10203.

Figure 6: An example of action generation in grounding via textual choices.

Example of Grounding via Textual Choices – 2nd Turn

Input – 2nd turn: Referring Prompts
Choices:
A. <button id=0 button header dropdown button> <div> JS
B.   Refer a friend, get $50 off
C. <div> Profile </div>
D. <button id=3 button> HIRE A PRO </button>
E. <button id=4 button> FIND CUSTOMERS </button>
F. <input id=5 search on thumbtack wedding photographer whats on your
G. <input id=6 zip code zip_code 10203 />
H. <button id=7 submit search> Search </button>
I. ,
J. .
K. <button id=10 button navigation arrow down />
L. <a id=11> <source image/webp /> <div furniture moving and heavy
M. <input id=12 search on thumbtack whats on your to-do list?
N. <input id=13 zip code zip_code 43235 />
O. <button id=14 button> Cancel </button>
P. <button id=15 submit> Search </button>
Q. None of the other options match the correct element

GPT-4V: (Reiteration)
The next target element is the "Search" button located to the right of the zip code field.

(Multichoice Question)
H. <button id=7 submit search> Search </button>

(Final Answer)

ELEMENT: H

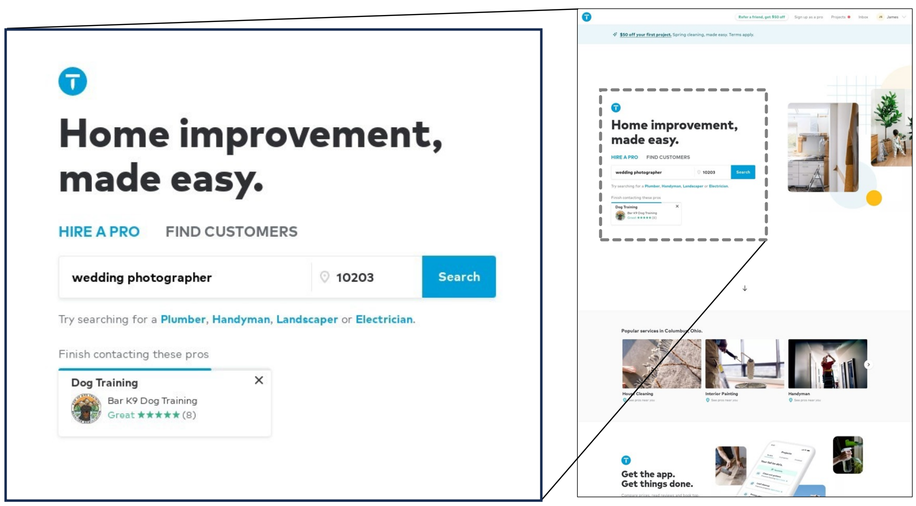
ACTION: CLICK

VALUE: None

Figure 7: An example of grounding via textual choices after action generation.

Example of Grounding via Image Annotation – 1st Turn

Input – 1st turn: Action Generation Prompts
Task: View the profile of a Wedding Photographer near 10203 for a 4 hour wedding on april 13
Previous Actions:
[textbox] Search on Thumbtack -> TYPE: wedding photographer
[textbox] Zip code -> TYPE: 10203



GPT-4V: (Current Webpage Identification)
The current webpage appears to be the homepage of Thumbtack, a local services marketplace where users can find professionals for various services, including home improvement, events, and more.

(Previous Action Analysis)
The first action was typing "wedding photographer" into a search box, indicating the user's intent to find a photographer for a wedding. The second action involved typing "10203" into a zip code field, specifying the location where the service is needed.

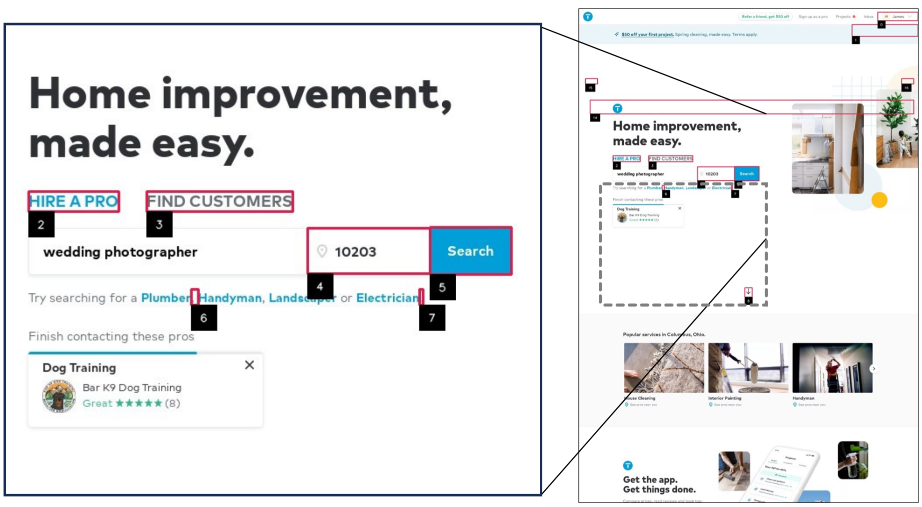
(Screenshot Details Analysis)
The screenshot shows the Thumbtack homepage with the search fields at the top. The "wedding photographer" text is visible in the search box, and "10203" is in the zip code field. There is a "Search" button next to these fields, which has not been clicked yet, as indicated by the presence of the search terms in the fields and the lack of search results on the page.

The first next target element is the "Search" button located to the right of the zip code field at the top of the webpage. The corresponding operation is to click this button to execute the search for wedding photographers near the zip code 10203.

Figure 8: An example of grounding via image annotation.

Example of Grounding via Image Annotation – 2nd Turn

Input – 2nd turn: Referring Prompts



GPT-4V: ELEMENT: 4

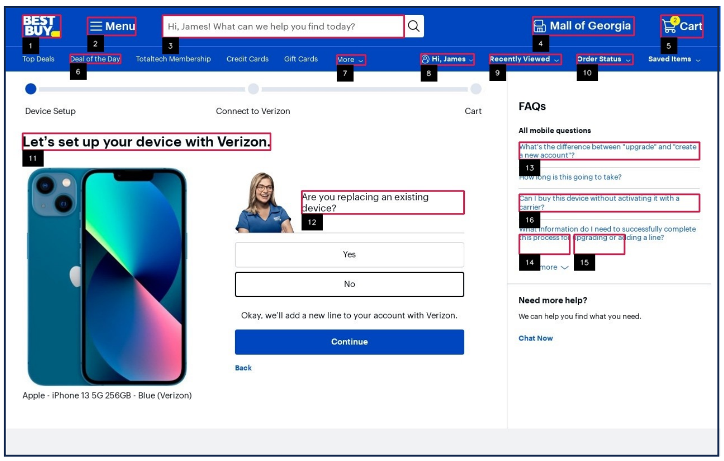
ACTION: CLICK

VALUE: None

Figure 9: An example of grounding via image annotation.

Makeup Image Annotation – Example 1

Task: As a Verizon user, finance a new blue iPhone 13 with 256gb along with monthly apple care.
Previous Actions:
...
[checkbox] AppleCare+ Theft and Loss for iPhone 13 - Monthly ... -> CLICK
[button] Add to Cart -> CLICK
[button] Let's Get Started -> CLICK
[label] Yes -> CLICK



Action Generation: ...The first next target element is the "Continue" button located in the center of the page, just below the message about adding a new line to the Verizon account. The corresponding operation is to click this button.

Action Grounding: ELEMENT: **12**

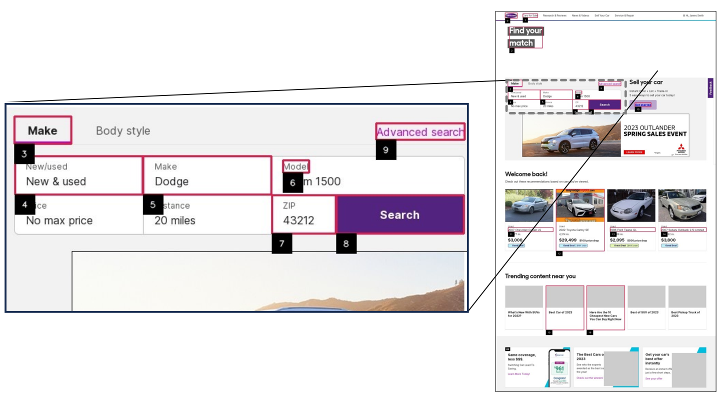
ACTION: CLICK

VALUE: None

Figure 10: In this example, the model describes a correct element in action generation. However, the identified element is absent from the set of candidate elements. Despite this, the model erroneously assigns it the index number "12".

Makeup Image Annotation - Example 2

Task: Find me the cheapest Dodge Ram 1500 within 50 miles of 21122
Previous Actions:
[combobox] Make -> SELECT: Dodge
[combobox] Model -> SELECT: Ram 1500



Action Generation: ... Next Action:
- Target Element: 'Distance' dropdown menu
- Location: Under the 'Model' dropdown, to the right of where '20 miles' is currently displayed
- Operation: Click on the 'Distance' dropdown menu to expand it and then select '50 miles'.

Grounding: ELEMENT: 5

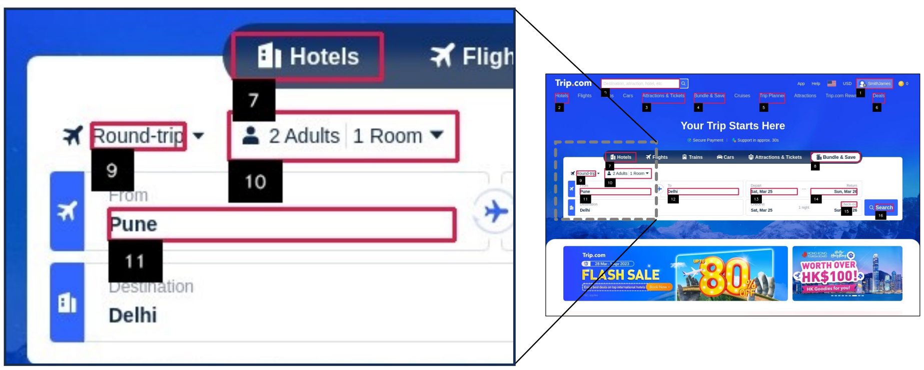
ACTION: CLICK

VALUE: None

Figure 11: In this example, the model predicts the appropriate element. Nevertheless, the identified element is not present in the provided image options. Despite this, the model erroneously assigns it with the label '5' nearby.

Failure of Linking - Example 1

Task: Book a cheapest bundle and save option for 2 adults from Ahmedabad to Dubai on April 5 with free cancellation options, hotel should be a 3 star near Burj Khalifa with guest rating above 4, one night
Previous Actions:
[span] Bundle & Save -> CLICK



Action Generation: The first next target element is the "From" field, located in the search form at the top of the page, which currently has "Pune" as the input. The corresponding operation is to clear the current input and type "Ahmedabad."

Next action: Click on the "From" field and type "Ahmedabad."

Action Grounding:
ELEMENT: **10**

ACTION: TYPE

VALUE: Ahmedabad

Figure 12: In this case, while the model predicts the appropriate element, it incorrectly associates the element with the nearby label ’10’ instead of the correct label ’11’.

Failure of Linking - Example 2

Task: Find a psychiatrist who offers virtual appointmentsthat has experience with treating neurodevelopment disorders and accepts new patients.
Previous Actions:
[searchbox] Search doctors, conditions, or procedures -> TYPE: psychiatrist
[em] Psychiatrist -> CLICK

Action Generation: ...Next action:
- **Click on the "Telehealth" filter located at the top of the page**, in the row of filters, to apply this criterion to the search results.

Action Grounding:
ELEMENT: **7**

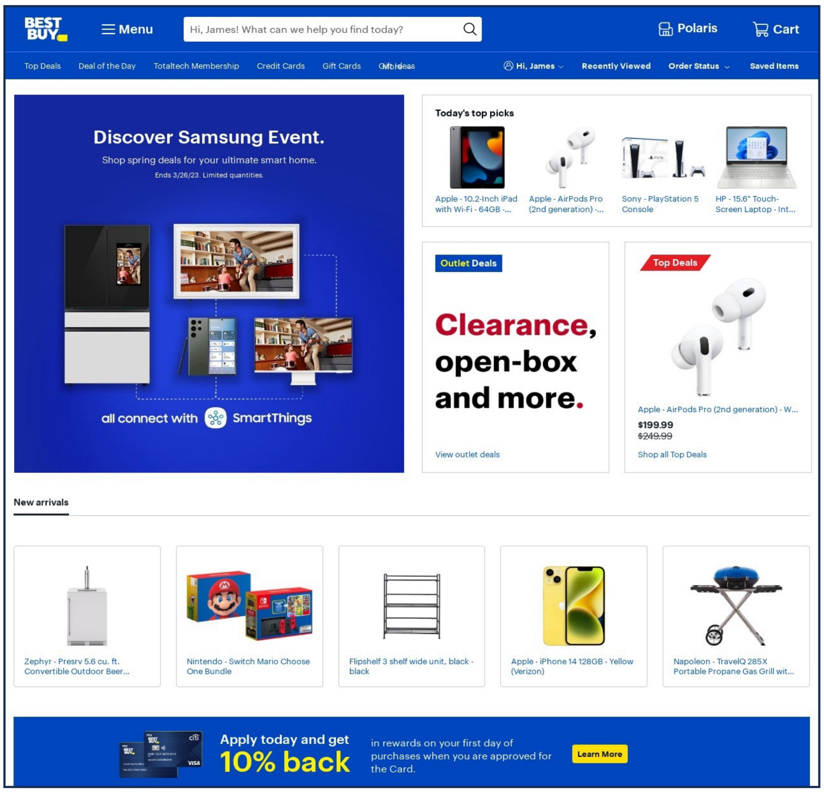
ACTION: CLICK

VALUE: None

Figure 13: In this case, while the model predicts the appropriate element, it incorrectly associates the element with the nearby label ’7’ instead of the correct label ’6’.

Strong Capability of Planning

Task: View the speakers that are bluetooth and wireless and filter the results to only show models that are on sale and cost less than $50.
Previous Actions:
None

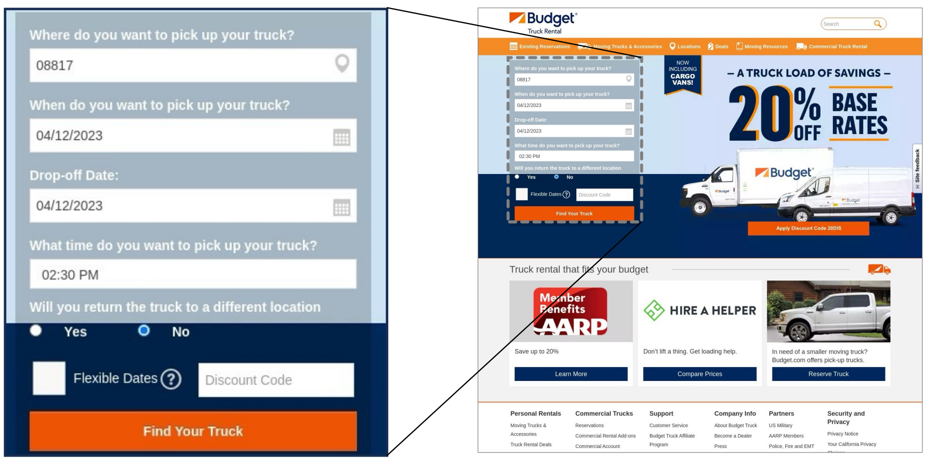


GPT-4V: … To proceed with the task, the user should: **Locate the menu bar to find the category for audio equipment or speakers. This is typically found in a dropdown under a section such as "Products", "Categories", or directly as "Audio". Once the correct category is found, the user should click on it to navigate to the section of the website that lists speakers. After reaching the speakers section, the user will need to use the website's filtering options to select 'Bluetooth' and 'Wireless' features, and then set the price filter to show only the models that are on sale and cost less than $50.** …

Figure 14: In this example, the model generates a comprehensive plan for the task, including subsequent actions on following pages that are not currently visible.

Strong Capability of Understanding

Task: Rent a truck with the lowest rent with two dozen furniture pads for 100 miles at zip 08817 on April 12 at 2:30 pm rented truck is to be returned to the exact location and date, and the pickup and drop off will be at the nearest location.
Previous Actions:
…
[textbox] US City,State or Zip Code -> TYPE: 08817
[textbox] mm/dd/yyyy -> CLICK
[link] 12 -> CLICK
[select] Select Pick-up Time -> SELECT: 02:30 PM
[input] -> CLICK

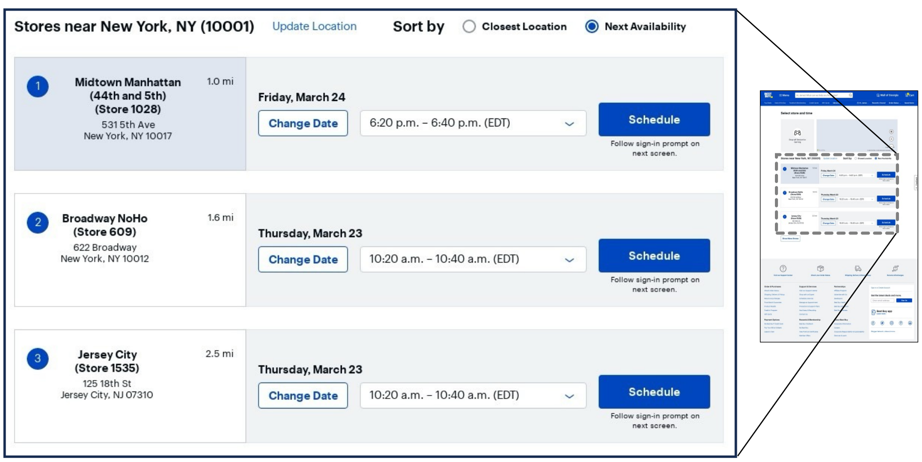


GPT-4V: … The current webpage appears to be the truck rental reservation page of the Budget Truck Rental website. This page is designed for customers to input their rental details such as pickup location, dates, and times.
(Screenshot Details and Previous Action Analysis)
The screenshot shows that the user has already entered the pickup location (zip code 08817), selected the pickup date (April 12, 2023), and chosen the pickup time (2:30 PM). The drop-off date is also set to the same day, which aligns with the task requirement of returning the truck to the same location on the same date. The option "Will you return the truck to a different location" is set to "No," which is correct for the task at hand. The last action taken was a click, but it's not clear on what element the click was performed. However, based on the task, it's likely that the user is ready to proceed to the next step of finding a truck.
(Intended Action Based on Current Page and Previous Actions)
The next logical step for the user to take would be to click on the "Find Your Truck" button. This action will likely lead to a page where the user can select the type of truck they wish to rent and add any additional accessories, such as the two dozen furniture pads mentioned in the task. Since the task specifies renting the truck with the lowest rent, the user will need to compare the available options and select the most cost-effective one.
The "Find Your Truck" button is located in the central portion of the webpage, just below the input fields for the rental details. The corresponding operation would be to move the cursor over the "Find Your Truck" button and perform a click action.…

Figure 15: In this case, two critical pieces of information are inadequately captured by the textual history. Firstly, the website automatically set the drop-off date to the same day. Secondly, secondly, the 'No' button was selected (However it was missed in previous actions due to the button's lack of text). Nevertheless, the model discerns these details through a meticulous analysis of the screenshot, enabling it to make the correct decision for the next step.

Identical Elements

Task: Schedule a repair service for my gaming console closest to the zip code 10001 on March 24, anytime after 6 pm.
Previous Actions:
...
[[link] Schedule a Service -> CLICK
[p] Video Games & Entertainment -> CLICK
[heading] Schedule a repair or tune-up -> CLICK
[button] Update Location -> CLICK
[textbox] Enter Zip Code or City and State -> TYPE: 10001
[button] Find Stores -> CLICK
[button] Change Date -> CLICK
[button] March 24, 2023, is available -> CLICK
[combobox] Select a Time Slot -> SELECT: 6:20 p.m. – 6:40 p.m. (EDT)



GPT-4V: ... The user is currently in the process of selecting a store location and time for the service.(Details and Previous Action Analysis)
The user initiated this process by navigating to the support & services section of the Best Buy menu, then selecting the option to schedule a service. They narrowed down their service type by selecting the "Video Games & Entertainment" category and chose to schedule a repair or tune-up. The user has also updated their location using the zip code 10001 to find stores near that area. They then selected the date of March 24, 2023, and chose the available time slot of 6:20 p.m. – 6:40 p.m. (EDT) for their service. This time slot fits the requirement of scheduling the service anytime after 6 pm on March 24th.From the screenshot, three store options near the zip code 10001 are listed. The closest one is the Midtown Manhattan location, which is 1.0 mile away from the provided zip code. This store already has the selected date and time of Friday, March 24, from 6:20 p.m. to 6:40 p.m. The last action from the previous action history indicates that the user has selected the time slot of 6:20 p.m. – 6:40 p.m. (EDT). This suggests that the user is now ready to finalize and confirm this scheduled service.(Intended Action Based on Current Page and Previous Actions)
Considering the details and the objective to schedule a repair service, the next logical step for the user would be to confirm and schedule the service at the selected store, date, and time. On the current page, the element that seems to be relevant to this action is the "Schedule" button associated with the selected store, date, and time. **Therefore, the next action step is:**
[button] Schedule (associated with the Midtown Manhattan location for the selected date and time) -> CLICK.

Figure 16: In this example, there are three identical 'Schedule' buttons, making it impossible for SEEACT$_{\text{Choice}}$ to distinguish among them. We empirically find that the model tends to choose the first one among the choices.

Knowledge Requirement – Example 1

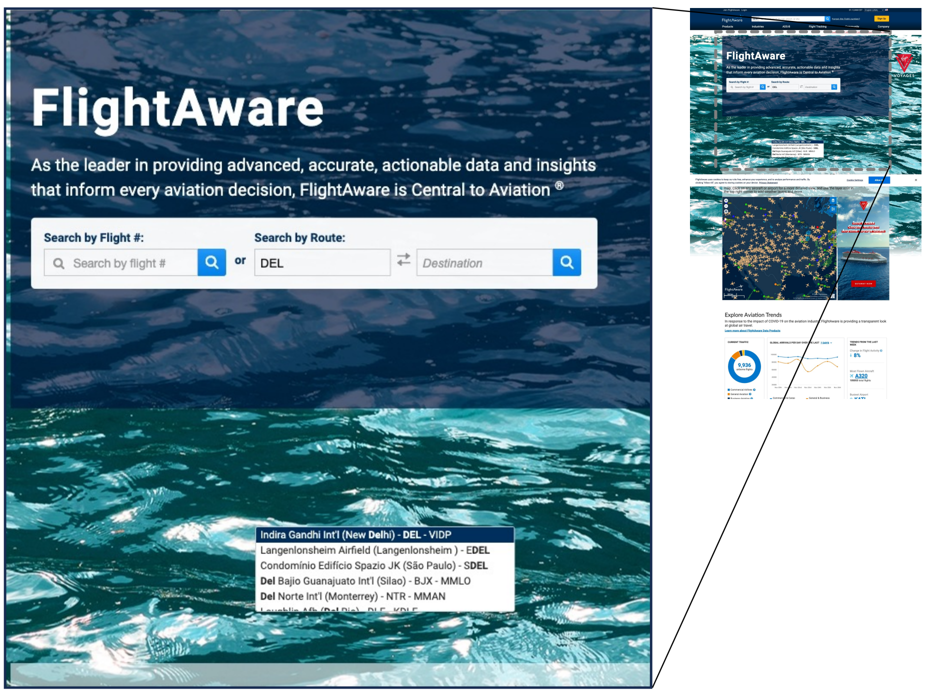
Task: Find a driver training school in Dublin
Previous Actions:
…
[link] All -> CLICK
[link] Attending a Driver Training School -> CLICK
[link] Driver Training Schools -> CLICK
[link] Find a Driver Training School Near You -> CLICK

GPT-4V: … The current page has a list of districts, which implies that the user may need to select a district closest to Dublin, Virginia, to continue their search. **However, since there is no district explicitly named "Dublin," the user must know which district Dublin falls under or look for an option that allows for a more specific search.**
(Intended Action Based on Current Page and Previous Actions)
The first next target element that the user would likely need to operate with is the district name that includes or is closest to Dublin, Virginia. However, the screenshot does not provide enough detail to determine which district Dublin is in.…

Figure 17: In this example, the task necessitates knowledge about which district Dublin is located in.

Knowledge Requirement – Example 2

Task: Find flights going from Indira Gandhi to Los Cabos.
Previous Actions:
[textbox] Origin -> TYPE: DEL



GPT-4V: ... Then, the user needs to identify the IATA code for Los Cabos International Airport, which is "SJD," and enter it into the destination field. Thus, the next target element is the destination field, which is located to the right of the origin field in the "Search by Route" section of the search interface.
Here is the action to be taken:
[textbox] Destination -> TYPE: SJD (after ensuring the origin field correctly contains "DEL").

Figure 18: In this example, the task requires knowledge of the IATA code for Los Cabos International Airport. GPT-4V accurately provides the correct code.

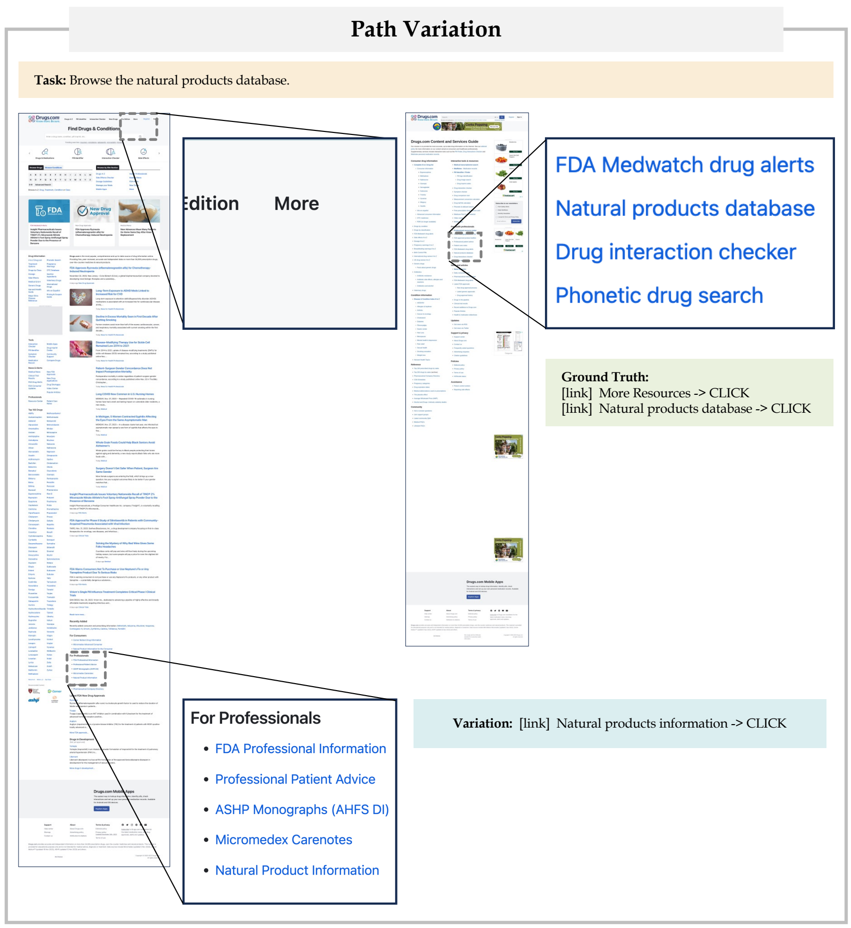
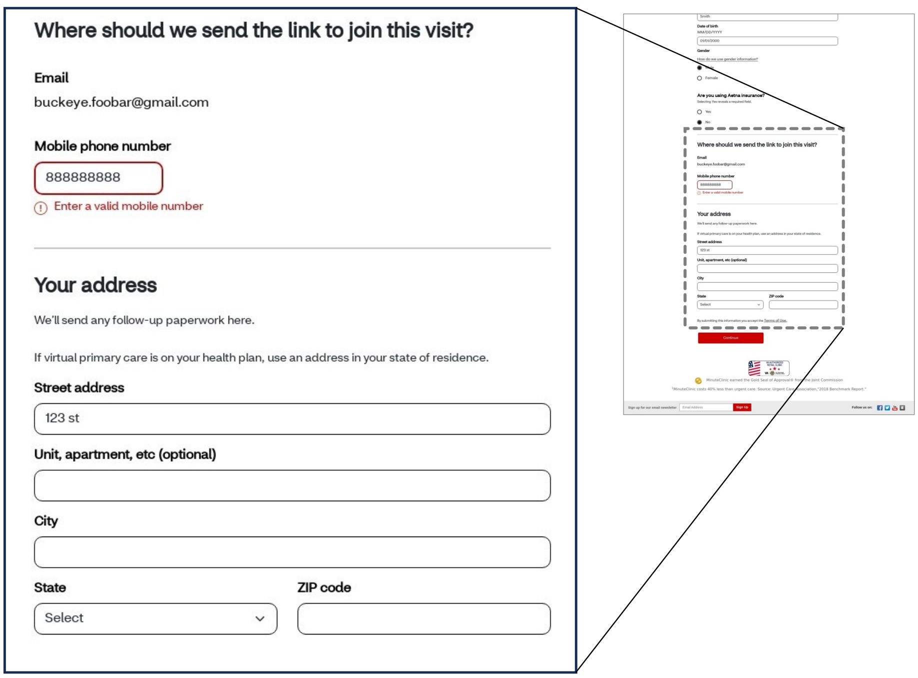


Figure 19: In this example, the ground truth in Mind2Web is to firstly click "More", then click "Natural products database" on the second page. In contrast, the model identifies a more direct approach, achieving the target page through 'Natural products information' on the first page.

Awareness of Error Correction

Task: Signup for virtual healthcare visit.
Previous Actions:
...
[span] James Smith -> CLICK
[button] Select patient and continue -> CLICK
[textbox] Date of birth -> CLICK
[textbox] Date of birth -> TYPE: 01/01/2000
[radio] Male -> CLICK
[radio] No -> CLICK
[textbox] Mobile phone number -> TYPE: 888888888
[textbox] Street address -> TYPE: 123 st



GPT-4V: ... Since the mobile phone number field is highlighted, it suggests there might be a problem with the entry—such as an incorrect format or not enough digits. **Therefore, the next action would be to correct the mobile phone number by clicking on the mobile phone number text box and entering a valid number.** Once that's corrected, the user should continue filling out the rest of the form with the city, state, and ZIP code before proceeding to click the "Continue" button to move forward with the signup process...

Figure 20: In this example, the webpage displays an error message indicating an invalid phone number, a consequence of prior actions. The model identifies this error and prioritizes its immediate rectification, foregoing the subsequent planned steps.