

000 001 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030 031 032 033 034 035 036 037 038 039 040 041 042 043 044 045 046 047 048 049 050 051 052 053 LEARNING MULTIMODAL TRAJECTORY REPRESENTATIONS FOR WEB AGENT PLANNING

Anonymous authors

Paper under double-blind review

ABSTRACT

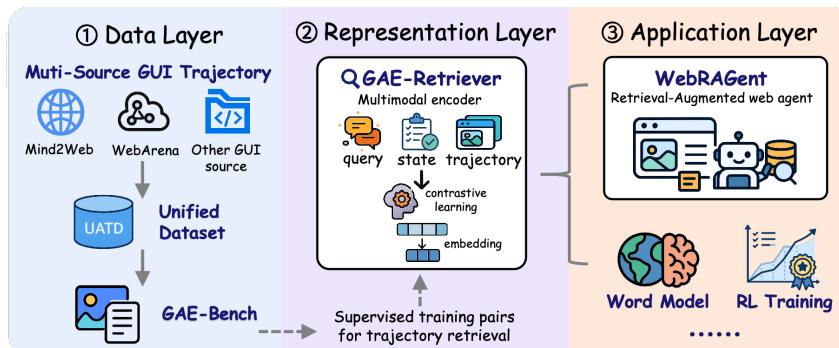
Trajectory data, capturing multimodal human actions and states, are pivotal for building autonomous GUI agents and transferring skills across tasks, encoding knowledge by compressing past experience into structured Markov sequences. Yet current methods for trajectory modeling remain fragmented, often relying on task-specific heuristics or textual signals. Progress on multimodal trajectories has been limited by the difficulty of representing visual information within long-step histories that exceed model context windows. Hence, how to effectively learn from multimodal trajectories remains a major and insufficiently addressed challenge amid ever-growing datasets. In this work, we introduce Multimodal Trajectory Retrieval, bridging the gap between universal retrieval and agent-centric trajectory modeling. We construct the Unified Agent Trajectory Dataset (UATD) from annotated demonstrations and states across diverse real-world scenarios. Building on this, we present GAE-Bench, a benchmark containing a large number of trajectory-based retrieval pairs. Our proposed GAE-Retriever, a multimodal retriever based on vision-language models that uses token selection and GradCache to optimize the contrastive objective. Over multiple web-agent datasets, it surpasses strong baselines on retrieval recall. To demonstrate potential downstream applications, we develop WebRAGent, a retrieval-augmented web agent that integrates GAE-Retriever and supports both DOM- and vision-based observations. WebRAGent proves effective on both textual and visual retrieved knowledge, achieving performance gains of over 15% vs. non-retrieval on the Online-Mind2Web benchmark.

1 INTRODUCTION

Human experience, meticulously recorded across diverse media like text, images, videos, and structured processes, forms a rich repository of knowledge known as *trajectories*. These trajectories encapsulate not only the actions performed but also the environmental states in which they occurred. The vast amount of trajectory data already available in human-generated content, such as instructional videos (Tang et al., 2019) and illustrated guides (Zhou et al., 2022), is continuously expanding through the efforts of AI agent researchers and the deployment of agent products. This wealth of experiential data offers significant value, not only for human reuse and learning but also for enhancing the intelligence in fields such as embodied intelligence (Yue et al., 2024) and computer-use agents (Zhao et al., 2024). Given the ever-increasing volume of trajectory data, a critical question arises: *How can we effectively model these trajectories to boost more advanced intelligence?*

Prior methodologies have explored the representation of states and actions to facilitate the retrieval of optimal actions, diverging from the generation-based approach (Yao et al., 2022). Trajectory-level experience data have shown value within in-context reasoning paradigms (Sridhar et al., 2024; Wang et al., 2024c; Liu et al., 2025a) and reinforcement training (Goyal et al., 2022; Humphreys et al., 2022; Kang et al., 2024), highlighting the rich semantic information embedded within task instructions, states, and trajectories in a latent space that can significantly aid agent inference and learning. However, trajectory-based data inherently involve substantial token consumption, making the efficient retrieval of the most pertinent trajectory data a critical challenge during the trajectory data explosion. Furthermore, state representation modeling holds immense potential for tasks such as world modeling (Gu et al., 2024) and search algorithms (Putta et al., 2024; Koh et al., 2024b), presenting a viable alternative to the conventional strategies of prediction and actual interaction with environments. Despite the promise of trajectory and state representation modeling, existing

054 studies lack a systematic evaluation of model capacities and a comprehensive understanding of how
 055 to scale these capacities with increasing data and task complexity. This work chooses GUI-based
 056 environments as an initial ground for exploration, driven by the practical value of web automation
 057 across diverse applications and the abundance of existing data resources, presented by pioneering
 058 works in this field such as Mind2Web (Deng et al., 2023) and WebArena (Zhou et al., 2024).
 059



060
 061
 062
 063
 064
 065
 066
 067
 068
 069
 070
 071 Figure 1: Overview of the proposed framework. The pipeline spans three layers: (1) Data Layer, unifying
 072 heterogeneous trajectories into **UATD**; (2) Representation Layer, where **GAE-Retriever** learns multimodal
 073 embeddings; and (3) Application Layer, featuring **WebRAGent** for retrieval-augmented planning for web
 074 navigation. Furthermore, the learned trajectory representations demonstrate promising generalization capabilities,
 075 paving the way for advanced paradigms such as world modeling and RL training.
 076

077 Figure 1 illustrates that our study adopts a pipeline encompassing unified data collection, representation
 078 learning, and downstream applications. Building on this framework, we propose a multimodal
 079 trajectory retrieval as a cornerstone for future research, consisting of four main contributions:
 080

081 1. To unify heterogeneous trajectories across web (current), mobile, desktop, and embodied environments (future), we construct the **Unified Agent Trajectory Dataset** (UATD), built from five
 082 open-source GUI agent datasets. UATD includes 7,747 human-annotated demonstrations and 82,793
 083 states, covering diverse real-world use cases such as shopping, travel, business, and social media.
 084 Each trajectory features high-definition visual observations, a standardized action format, and natural
 085 language state descriptions to support customization and the development of downstream tasks.

086 2. We are the first to propose the task of **Multimodal Trajectory Retrieval**, bridging the gap
 087 between universal retrieval and trajectory modeling. This task captures both temporal and semantic
 088 correlations within and across trajectories, targeting fine-grained intra-trajectory components as well
 089 as coarse-grained inter-trajectory semantic relations. To formalize the task, we design 12 extraction
 090 patterns that derive six types of retrieval samples from a single trajectory, where the query and target
 091 can be a state, a trajectory, or a subtrajectory. These include (1) text-to-state, (2) text-to-trajectory, (3)
 092 state-to-state, (4) state-to-trajectory, (5) trajectory-to-state, and (6) trajectory-to-trajectory retrieval.
 093

094 Based on this formulation, we annotate the **GUI Agent Embedding Benchmark** (GAE-Bench) by
 095 converting the UATD using the predefined extraction patterns, yielding a total of 714,628 positive
 096 retrieval pairs. To accommodate the context length limitations of current multimodal language
 097 models and enhance applicable use in downstream applications, we also release GAE-Bench-lite, a
 098 constrained version in which trajectory lengths are capped at 10 steps. It contains 514,956 training
 099 samples, 21,805 in-domain samples, and 27,139 out-of-domain samples, with 574,222 candidates.
 100

101 3. We implement the **GUI Agent Embedding Retriever** (GAE-Retriever), a novel multimodal
 102 trajectory retrieval framework built on VLM2Vec (Jiang et al., 2025), which adopts vision-language
 103 models (VLMs) as its backbone. In contrast to CLIP-based models (Radford et al., 2021; Li et al.,
 104 2022; 2023), VLMs are pretrained on large-scale, instruction-following multimodal data and can
 105 process arbitrary-length combinations of visual and textual inputs, making them well-suited for
 106 modeling multimodal trajectories. To enable effective contrastive learning over multiple high-
 107 resolution trajectory screenshots and a large number of in-batch negatives under limited computing
 108 resources, GAE-Retriever incorporates a token selection mechanism and GradCache optimization.
 109

110 4. We propose **WebRAGent**, a retrieval-augmented multimodal web agent framework that integrates
 111 the GAE-Retriever. WebRAGent supports both DOM and Vision modes and comprises observation,
 112 retrieval, memory, planning, and reward modules. WebRAGent applies the GAE-Retriever to encode
 113

108 the task query and the current state screenshot into embeddings, retrieving step-level guidance for the
 109 next action. We validate this multimodal, model-based retrieval approach on the Online-Mind2Web
 110 benchmark (Xue et al., 2025), where it improves success rate by **15-22%** over non-retrieval baselines
 111 and yields larger gains on hard tasks. Finally, we analyze the respective contributions of single-modal
 112 and multi-modal in-context learning to retrieval-augmented agents.

113 We conduct comprehensive evaluations of GAE-Retriever against multimodal backbone models,
 114 retrieval models, and trajectory planning models, showcasing its strong performance across all
 115 baselines on Recall@1/5/10. Compared to the best-performing baselines, GAE-Retriever achieves
 116 an average improvement of 10.22 points across five datasets, demonstrating the effectiveness of our
 117 training approach. Beyond this, our work establishes a bridge between retrieval and execution by
 118 effectively preserving and leveraging knowledge from similar trajectories. We further demonstrate
 119 the generalization and effectiveness of WebRAGent on the Online-Mind2Web benchmark.

122 RELATED WORK

124 GUI AGENTS

127 The evolution of language models has introduced strong capabilities in tool use, environmental
 128 grounding, and complex reasoning for agentic tasks (Su et al., 2024), such as web browsing (Yao
 129 et al., 2022), travel planning (Xie et al., 2024a; Zhang et al., 2024d), and societal simulation (Park
 130 et al., 2023). GUI navigation (Zhang et al., 2024a), originally derived from web-based tasks, has
 131 become an active area of research across domains including web (Deng et al., 2023; Zhou et al., 2024;
 132 Koh et al., 2024a), mobile (Rawles et al., 2023; 2024), and computer control (Xie et al., 2024b).

133 To better reflect realistic conditions, GUI agents have evolved from reactive systems (Lee et al.,
 134 2023; Li et al., 2024), to proactive agents that take actions based on conversational understanding or
 135 situational reasoning. These two paradigms have led to distinct approaches: user interaction through
 136 dialogue (Deng et al., 2024), environment-based reasoning (Yao et al., 2023), and tree search (Koh
 137 et al., 2024b). However, real-time exploration can be inefficient and may sacrifice user experience. To
 138 reduce this overhead, some approaches retrieve reusable subroutines (Wang et al., 2024c; Zheng et al.,
 139 2025) or demonstration histories from memory (Kagaya et al., 2024; Liu et al., 2025a). Nonetheless,
 140 they perform similarity search using only textual features, neglecting richer multimodal signals.

142 MULTIMODAL RETRIEVAL

144 Recent advances in foundational vision-language models (Liu et al., 2023; Peng et al., 2023) have
 145 shifted research focus from generation to retrieval. Early approaches (Muennighoff et al., 2023; Fu
 146 et al., 2023) primarily addressed single-modality retrieval tasks such as text-to-image and text-to-text
 147 retrieval. More recent efforts (Wei et al., 2024; Jiang et al., 2024; 2025; Zhang et al., 2024b; Lin
 148 et al., 2024b) have expanded to composed and cross-modal retrieval settings, constructing large-scale
 149 benchmarks from vision-language datasets that extend to classification and grounding tasks.

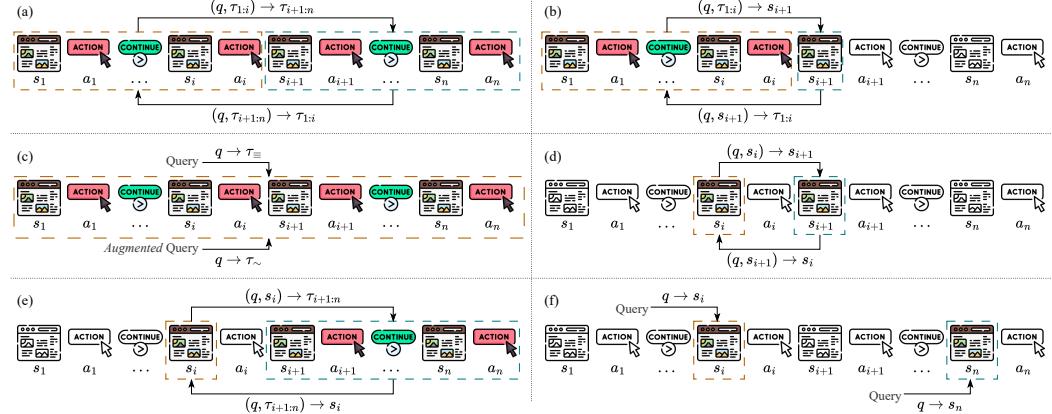
150 Recent studies have targeted more practical domains, including visual documents (Faysse et al., 2024),
 151 videos (Wang et al., 2024b), and screenshots (Liu et al., 2025b). VLM2Vec-V2 (Meng et al., 2025),
 152 for instance, integrates data from diverse modalities to learn universal discriminative embeddings.
 153 Despite substantial progress in multimodal retrieval, few works have explored integrating trajectory
 154 data into retrieval model training. Notably, contemporary findings (Fu et al., 2024; Sridhar et al.,
 155 2024; Yue et al., 2024) highlight the potential of agent trajectories for downstream in-context planning
 156 tasks, underscoring a promising but underexplored direction in this research area.

157 While retrieval is a long-standing topic in multimodal learning, our focus is on autonomous GUI
 158 agents, where most systems remain purely reactive and predict the next action directly from previous
 159 observations without an explicit trajectory memory. In contrast, we advocate retrieval-augmented
 160 planning with explicit long-horizon trajectory representations as a queryable memory, and our multi-
 161 image, high-resolution states ($\sim 1000 \times 1000$) fall outside the standard CLIP-style single-image
 224 \times 224 (or extended composed video) setting, motivating a dedicated trajectory retriever.

162 3 DATASET

164 To build datasets for the multimodal trajectory retrieval task, a unified agent trajectory format is
 165 required for generating valid positive and negative samples. Current datasets, whether sourced
 166 from digital environments (Zhou et al., 2024; Chai et al., 2024) or embodied platforms (Shridhar
 167 et al., 2021), encode trajectories using heterogeneous structures, making consistent retrieval data
 168 extraction nontrivial. To address this challenge, we design a pipeline that converts these datasets into
 169 a standardized format, resulting in the **Unified Agent Trajectory Dataset** (UATD). From UATD, we
 170 define 12 extraction schemes that convert each individual trajectory into labeled samples covering six
 171 core retrieval tasks, forming the **GUI Agent Embedding Benchmark** (GAE-Bench) series.

172 3.1 UATD



188 Figure 2: Illustration of positive pair extraction from UATD. Subfigures (a), (b), (d), and (e) depict temporal
 189 retrieval; (c) and (f) show semantic retrieval. We use $\tau_{i:j}$ to denote a subsequence of trajectory τ from state s_i to
 190 action a_j , i.e., $\tau_{i:j} = (s_i, a_i, \dots, s_j, a_j)$. q serves as the retrieval query, composed of a task-specific instruction
 191 (shown in Table 1) along with a corresponding trajectory description.

192 Table 1: The overview of the GUI Agent Embedding Benchmark (GAE-Bench) series. This table summarizes
 193 the six fundamental retrieval tasks and their twelve corresponding subtasks, developed from the predefined
 194 extraction schemes in Figure 2. For every subtask, we present a representative instruction (see Appendix E.1 for
 195 more instruction templates). The table also reports the number of subtask examples in GAE-Bench and the split
 196 statistics in GAE-Bench-lite: training, in-domain (IND), out-of-domain (OOD), and total instances.

Task	Subtask	Instruction (shown 1 out of 10)	GAE-Bench				GAE-Bench-lite				
			Train	IND	OOD	Total	Train	IND	OOD	Total	
1. $(q, \tau) \rightarrow \tau'$	$(q, \tau_{1:i}) \rightarrow \tau_{i+1:n}$	Apply the request to the previous web navigation steps to derive the next trajectory.	75,046	40,748	2,523	2,665	45,936				
	$(q, \tau_{1:i:n}) \rightarrow \tau_{1:i}$	Find the previous web browsing trajectory based on the user input and the current trajectory.	75,046	40,748	723	2,665	44,136				
2. $(q, \tau) \rightarrow s$	$(q, \tau_{1:i}) \rightarrow s_{i+1}$	Locate the following state from the given former web navigation trajectory and the task input.	75,046	51,661	3,197	2,665	57,523				
	$(q, \tau_{1:i:n}) \rightarrow s_i$	Locate the former observation using the instruction and the upcoming web interaction trajectory.	75,046	51,661	3,197	2,665	57,523				
3. $q \rightarrow \tau$	$q \rightarrow \tau_{\equiv}$	Identify the unique navigation trajectory for web agents according to the provided instruction.	7,747	3,717	424	470	4,611				
		Retrieve the analogous interaction history for GUI agents based on the provided command.	38,735	20,081	624	2,350	23,055				
4. $(q, s) \rightarrow s'$	$(q, s_i) \rightarrow s_{i+1}$	Using the provided instruction and the former state, what is the next GUI navigation state?	75,046	68,849	3,532	2,665	75,046				
	$(q, s_{i+1}) \rightarrow s_i$	Taking the description and the current state into account, search the previous web agent state.	75,046	68,849	3,532	2,665	75,046				
5. $(q, s) \rightarrow \tau$	$(q, s_i) \rightarrow \tau_{i+1:n}$	Locate the next GUI navigation trajectory by applying the instruction to the previous state.	75,046	54,124	734	2,665	57,523				
	$(q, s_{i+1}) \rightarrow \tau_{1:i}$	Identify the web navigation trajectory preceding the current state according to the task.	75,046	54,124	734	2,665	57,523				
6. $q \rightarrow s$	$q \rightarrow s_i$	From the description, identify the specific navigation observation for GUI navigation.	60,031	53,892	3,610	2,529	60,031				
	$q \rightarrow s_n$	Search the terminal observation in the web navigation for the task instruction.	7,747	6,502	775	470	7,747				

206 To identify the essential element in trajectory representations, we model a typical agent interacting
 207 with an environment as a deterministic Markov Decision Process (MDP) $\mathcal{E} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T})$, where
 208 \mathcal{S} denotes the state space, \mathcal{A} the action space, \mathcal{O} the observation space, and $\mathcal{T} : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ is the
 209 environment transition function. A trajectory is thus represented as a sequence of state s_i and action
 210 a_i with n steps, $\tau = (s_1, a_1, s_2, a_2, \dots, s_n, a_n)$, where each transition satisfies $s_{i+1} = \mathcal{T}(s_i, a_i)$.

211 Inspired by AGUVIS (Xu et al., 2024), we take each observation $o_i \in \mathcal{O}$ to be the raw visual content
 212 of the interface (i.e., a screenshot). This choice eliminates reliance on platform-specific textual
 213 representations and promotes broader generalization across visual contexts. For each action $a_i \in \mathcal{A}$,
 214 we stipulate three components: (1) operation: the name or type of the action; (2) target: the object
 215 or region within the environment on which the action is executed; (3) value: additional arguments
 required to perform the action. To address cross-platform discrepancies in action spaces that hinder

scalability, we follow the action README from ShowUI (Lin et al., 2024a), allowing each trajectory to be associated with its own customizable textual action definitions.

To create the dataset under a uniform trajectory representation, we collect real-world human trajectories with their action definitions from five existing GUI data sources. The details of dataset construction and statistics are provided in Appendix B.1.

3.2 GAE-BENCH

After acquiring the unified trajectory dataset, we introduce 12 extraction patterns to generate ground truth retrieval pairs from each trajectory based on a query q , as depicted in Figure 2. These patterns encompass two categories of multimodal trajectory retrieval: **temporal retrieval** (Task 1, 2, 4, and 5 in Table 1), which captures sequential relationships within a trajectory, and **semantic retrieval** (Task 3 and 6 in Table 1), which targets underlying intent or operation over different trajectories and states.

Figure 2 subfigures (a) and (d) detail scenarios to retrieve the next state s_{i+1} or the remaining trajectory $\tau_{i+1:n}$ given a current state s_i or a prefix trajectory $\tau_{1:i}$, and vice versa. Subfigures (b) and (e) demonstrate situations involving different granularities, such as identifying the correspondence between the partial trajectory $\tau_{1:i}$ and the next state s_{i+1} , or between the present state s_i and a remaining trajectory $\tau_{i+1:n}$. These positive pairs capture sequential relations within a trajectory.

Figure 2 subfigures (c) and (f) expound two types of semantic retrieval tasks. In particular, subfigure (c) defines retrieval between a query and its reference-aligned trajectory (known as the gold trajectory), denoted as $q \rightarrow \tau_{\equiv}$, which is directly from the UATD ground truth. Conversely, subfigure (f) constructs retrieval pairs between a query and a semantically similar variant (known as the silver trajectory), denoted as $q \rightarrow \tau_{\sim}$. This represents a trajectory that preserves the same high-level intent and functional objective but varies in contextual details. For instance, the silver trajectory corresponding to “Buy a t-shirt for children on Amazon” could be “Order a laser printer on eBay”.

Because generating silver trajectories is more challenging than augmenting queries, we design a three-step procedure to produce silver instructions for trajectories: (1) Identify named entities in the original query using named entity recognition; (2) Generate alternative expressions for the identified entities while preserving their types; (3) Rewrite the original query after substituting the identified entities with the corresponding alternatives. Appendix E.1 outlines the prompts for silver generation.

To implement $q \rightarrow s_i$ and $q \rightarrow s_n$ in subfigure (f), state descriptions from UATD are utilized to formulate retrieval of a specific intermediate state s_i and the final state s_n .

Table 2: Distribution of GAE-Bench by retrieval tasks and data sources (left; 714,628 positive pairs total) and its candidate sets (right). For each of the data sources, we define three types of candidate sets (i.e., *state*, *trajectory*, and *interval*), corresponding to retrieval targets of a single state s_i , a full trajectory τ , and a trajectory subsequence $\tau_{i:j}$. Note that $\tau_{i:j}$ can range in length from a single state-action pair to the entire trajectory.

Data Source	Retrieval Tasks						Total	Candidate Sets			
	$(q, \tau) \rightarrow \tau'$	$(q, \tau) \rightarrow s$	$q \rightarrow \tau$	$(q, s) \rightarrow s'$	$(q, s) \rightarrow \tau$	$q \rightarrow s$		State	Trajectory	Interval	Total
Mind2Web	16,306	16,306	8,808	16,306	16,306	10,943	84,975	9,475	1,468	48,238	59,181
AutoWebGLM	960	960	840	960	960	760	5,440	620	140	2,980	3,740
WebArena	2,040	2,040	1,206	2,040	2,040	1,305	10,671	1,104	201	6,640	7,945
WebLINX	12,794	12,794	2,910	12,794	12,794	6,337	60,423	5,852	485	76,095	82,432
GUIAct	117,992	117,992	32,718	117,992	117,992	48,433	553,119	42,980	5,453	529,528	577,961

Table 1 outlines the extracted positive pairs organized into six retrieval tasks and twelve subtasks that comprise GAE-Bench. To accommodate the limited context window of current multimodal models and maintain the feasibility of training, we also release GAE-Bench-lite, a constrained version of GAE-Bench in which each trajectory sequence contains fewer than 10 steps. To construct this version, we first randomly sample trajectories to form the out-of-domain evaluation subset. From the remaining positive pairs, we then partition the in-domain evaluation set and the training set, with approximately 90% of the examples assigned to the training set. To support efficient and applicable encoding during evaluation, we additionally introduce a mini partition of the GAE-Bench-lite candidate sets, referred to as GAE-Bench-lite (mini), which contains only candidates relevant to the evaluation subset. Detailed distributions of GAE-Bench is displayed in Table 2. See Appendix B.2 for details of GAE-Bench-lite and GAE-Bench-lite (mini).

270 4 FRAMEWORK
271272 This section presents the problem definition for the multimodal trajectory retrieval, along with our
273 model, GAE-Retriever, which extends VLM2Vec for trajectory learning.
274275 4.1 PROBLEM DEFINITION
276277 In real-use cases, trajectories may be gathered either as complete
278 sequences τ or as partial segments, such as individual observations
279 s_i or trajectory subsequences $\tau_{i:j}$. This gives rise to diverse types
280 of retrieval queries and candidate sets, as outlined in Table 1. The
281 retrieval key (i.e., the input) \mathbf{k} may consist of a textual query, a state,
282 a full trajectory, or a trajectory subsequence. Similarly, the retrieval
283 value (i.e., the target) \mathbf{v} can take the form of a state, a complete
284 trajectory sequence, or a subsequence.285 To handle different retrieval intentions, we formulate the structural
286 definition of retrieval keys and values for model input using the
287 following recursive grammar in Figure 3.288 In this grammar, text in teletype font (like `Observation`) denotes terminal symbols, and square
289 brackets `[]` indicate token placeholders. An n -step trajectory is denoted by $\tau_{1:n}$, where $\tau_{i:i}$ is a
290 state-action pair at step i , and $u_{i:j}$ refers to $\tau_{i:j}$ with action definitions. \tilde{q} denotes q with a task-specific
291 instruction (refers to Table 1) and contextual trajectory description.292 Formally, we can build a multimodal trajectory retrieval model f that accepts any type of valid key
293 and returns any type of value as specified by instruction \tilde{q} :

294
$$\mathbf{v}^* = \underset{\mathbf{v} \in \mathcal{V}}{\operatorname{argmax}} [f(\mathbf{k})^\top \cdot f(\mathbf{v})]$$

295

296 where \mathcal{V} corresponds to the heterogeneous set of candidates, $f(\cdot)$ is the retrieval function optimized
297 via maximum dot-product similarity, and \mathbf{v}^* is the predicted result.298 4.2 GAE-RETRIEVER
299300 We implement GAE-Retriever, a contrastive retrieval model implemented on top of VLM2Vec (Jiang
301 et al., 2025) for multimodal trajectory tasks, as portrayed in Figure 4. The main technical challenge
302 in modeling GAE-Retriever lies in memory constraints. Encoding these regions introduces excessive
303 visual tokens, leading to higher inefficient computational costs and limited scalability, particularly
304 as trajectory lengths increase. To address this, we follow prior work (Raposo et al., 2024; Lin
305 et al., 2024a) and apply token selection by constructing a UI-connected graph in RGB space to
306 guide attention toward salient elements while skipping redundant tokens. Specifically, adjacent
307 image patches with differences below a threshold are merged into a single token, reducing redundant
308 attention computation and improving inference efficiency.309 In our multimodal trajectory retrieval task, obtaining hard negatives is difficult, thus making large
310 batch sizes essential for learning high-quality embeddings through contrastive learning. However,
311 GPU memory limitations constrain the batch size and the number of in-batch negatives, especially
312 since each training instance may include multiple high-resolution images, increasing memory con-
313 sumption. To address this, we adopt GradCache (Gao et al., 2021), as in VLM2Vec, a gradient
314 caching method that decouples backpropagation between the encoder and the contrastive loss to
315 support larger batch training. Appendix C.1 presents the action JSON schema and examples.316 Given a pretrained VLM, we feed \mathbf{k} and \mathbf{v} to obtain embeddings, $f(\mathbf{k})$ and $f(\mathbf{v})$, by taking the final
317 layer representation of the last token. In order to train the model, we minimize the InfoNCE loss:

318
$$\mathcal{L} = -\log \frac{\phi(f(\mathbf{k})^\top f(\mathbf{v}^+))}{\sum_{\mathbf{v} \in \mathcal{B}} \phi(f(\mathbf{k})^\top f(\mathbf{v}))}$$

319 320

321 where \mathcal{B} is the set of in-batch candidates, \mathbf{v}^+ is the positive sample associated with the retrieval key
322 \mathbf{k} , t is the temperature parameter, and $\phi(\cdot)$ is the temperature-scaled exponential function defined as
323 $\phi(x) = \exp(\frac{x}{t})$. Here, the in-batch set \mathcal{B} comprises the positive \mathbf{v}^+ and all other value embeddings
from the current mini-batch; for a given key \mathbf{k} , elements in $\mathcal{B} \setminus \{\mathbf{v}^+\}$ serve as implicit negatives.

$$\begin{aligned}
 s_i &\rightarrow \text{Observation: [image}_i\text{]} \\
 a_i &\rightarrow [\text{action}_i] \\
 u_{i:j} &\rightarrow [\text{action space}] \tau_{i:j} \\
 \tau_{i:j} &\rightarrow \begin{cases} s_i a_i, & \text{if } i = j, \\ s_i a_i \tau_{i+1:j}, & \text{if } i < j \end{cases} \\
 \mathbf{k} &\rightarrow \tilde{q} \mathbf{v} \\
 \mathbf{v} &\rightarrow s_i \mid u_{i:j}
 \end{aligned}$$

Figure 3: Recursive grammar definition for model input.

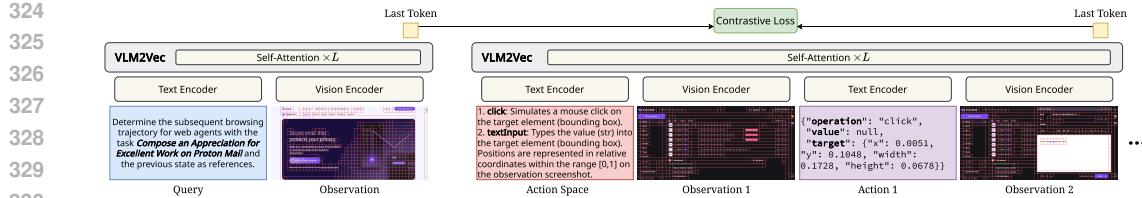


Figure 4: Illustration of GAE-Retriever on the subtask $(q, s_i) \rightarrow \tau_{i+1:n}$. Given a retrieval query and either a visual observation (left, used as the query) or a sequence of observation-action pairs (right, used as the key), GAE-Retriever employs a VLM backbone to process multimodal inputs, filtering out redundant visual tokens through token selection (red blocks in observation indicate similar token clusters). Training is performed via contrastive loss between query \mathbf{q} and target \mathbf{v} sides, conditioned on task-specific instructions.

4.3 WEBRAGENT: RETRIEVAL-AUGMENTED GENERATION FOR WEB NAVIGATION

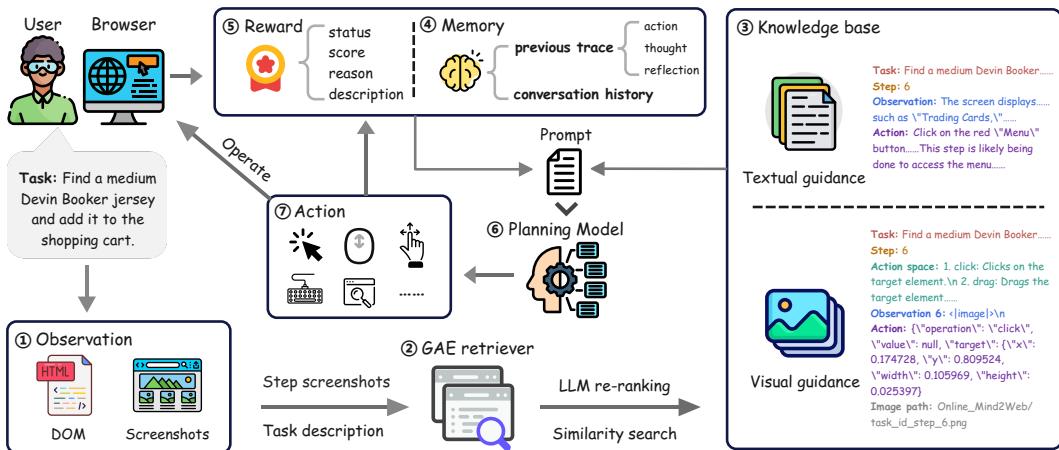


Figure 5: WebRAGent workflow. The system supports two observation modes, DOM or page screenshots ①. Before each decision, WebRAGent uses GAE-Retriever ② to encode the task instruction and step screenshot into a query. Next, it retrieves from the knowledge base ③ step-relevant textual or visual guidance. The retrieved knowledge, with memory ④ and prior reward feedback ⑤, is composed into prompts for the planning model ⑥, resulting in actions ⑦. This design applies external experience for decision-making and execution.

We introduce WebRAGent, a framework that integrates multimodal retrieval augmentation with a web agent (see Figure 5). Given a web navigation task q , the agent operates in an environment $\mathcal{E} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T})$. At step i , the agent in state $s_i \in \mathcal{S}$ receives an observation $o_i \in \mathcal{O}$ and selects an action $a_i \in \mathcal{A}$; the environment transition function \mathcal{T} then yields the next state $s_{i+1} = \mathcal{T}(s_i, a_i)$. Each component in the framework is elaborated below. Further details are shown in Appendix C.3.

Observation. (1) DOM mode: interactive DOM tree elements; (2) Vision mode: screenshot.

Step-Level Knowledge Retrieval. The GAE-Retriever model retrieves the next state s_{i+1} from the current state s_i and user query q , i.e., $(q, s_i) \rightarrow s_{i+1}$, shown in Figure 2(d). To further improve matching precision, we apply an LLM re-ranking stage. We align each step in the UATD trajectories by pairing every observation-action tuple (\tilde{o}, \tilde{a}) . Based on these aligned pairs, we construct two knowledge bases: (1) a *textual knowledge base* $\mathcal{D}_{\text{text}}$, where each element $t \in \mathcal{D}_{\text{text}}$, $t = (\tilde{o}_{\text{text}}, \tilde{a}_{\text{text}}, \mu)$, with \tilde{o}_{text} and \tilde{a}_{text} denoting descriptions of the observation and action, and μ denoting associated metadata; (2) a *visual knowledge base* $\mathcal{D}_{\text{vision}}$, where each element $v \in \mathcal{D}_{\text{vision}}$, $v = (\tilde{o}_{\text{vision}}, \tilde{a}_{\text{str}}, \mu)$, with $\tilde{o}_{\text{vision}}$ denoting an exemplar screenshot and \tilde{a}_{str} a structured representation of the action.

Memory and Reward. We formalize the step- i cycle of Thought–Action–Reflection–Reward as the historical memory M_i . The reward module adopts an *LLM-as-judge* scheme to evaluate the preceding trajectory, yielding a score (1–10), the grading rationale, and a follow-up plan, denoted by R_i .

Planning and Action. We use modality-specific planners π for different observation modes. More details for planning models are shown in Appendix C.3. Given task instruction q , observation o_i , memory M_i , reward R_i , and retrieved guidance v , the model outputs the action $a_i = \pi(q, o_i, M_i, R_i, v)$.

378

5 EXPERIMENTS

379

380 Table 3: Overall evaluation of all methods on each data source, reported in Recall@1/5/10.

381 Method	382 Mind2Web				383 AutoWebGLM				384 WebArena				385 WebLINX				386 GUIAct			
	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10		
Multimodal Backbone Models																				
Qwen2-VL-2B (Wang et al., 2024a)	0.7	14.5	18.2	1.2	6.3	10.7	1.4	8.8	12.2	3.1	14.2	18.0	3.1	8.1	9.4					
Qwen2.5-VL-3B (Bai et al., 2025)	1.0	7.8	9.7	0.9	3.8	6.3	0.7	6.5	9.9	3.4	13.0	15.9	3.0	7.9	9.5					
Multimodal Retrieval Models																				
LamRA-Ret (Liu et al., 2024)	1.1	15.1	19.2	4.9	15.2	22.8	2.0	10.3	14.5	3.4	16.9	21.5	4.2	10.4	12.4					
ColQwen2-v1.0 (Faysse et al., 2024)	3.2	22.0	29.9	3.9	17.7	26.3	2.9	13.7	20.0	4.2	19.6	25.1	6.2	15.5	19.2					
GME-Qwen2VL-2B (Zhang et al., 2024c)	3.7	24.2	33.4	8.7	27.9	37.4	4.2	17.7	24.7	5.2	22.4	29.7	6.0	16.7	20.7					
UniSE-MLLM (Liu et al., 2025b)	0.8	12.6	16.0	0.3	4.9	9.5	1.2	9.0	12.8	2.9	14.2	17.6	3.1	7.9	9.7					
VLM2Vec-Qwen2VL-2B (Jiang et al., 2025)	6.7	37.5	51.5	12.7	41.8	54.1	5.5	22.5	31.3	6.7	27.6	38.3	7.0	20.2	25.6					
VLM2Vec-V2.2 (Meng et al., 2025)	10.2	44.0	60.1	15.7	51.2	67.1	9.1	29.1	37.8	10.7	38.4	50.5	12.2	33.1	40.6					
Multimodal Trajectory Planning Models																				
UGround-V1-2B (Gou et al., 2024)	0.8	12.6	16.0	0.3	4.9	9.5	1.2	9.0	12.8	2.9	14.2	17.6	3.1	7.9	9.7					
ShowUI-2B (Lin et al., 2024a)	1.0	13.3	17.0	0.8	6.0	8.2	1.6	8.5	11.7	3.3	13.7	17.3	3.1	7.9	9.2					
UI-TARS-2B-SFT (Qin et al., 2025)	0.7	12.5	15.6	0.6	4.8	8.4	1.1	7.4	11.2	3.0	13.7	17.3	3.1	8.0	9.4					
TongUI-3B (Zhang et al., 2025)	1.3	9.3	11.4	0.5	3.9	7.3	1.5	7.0	10.5	3.4	13.8	17.4	3.0	8.1	9.7					
Multimodal Trajectory Retrieval Models																				
GAE-Retriever (Ours)	15.0	50.7	67.6	22.1	63.6	76.3	10.3	31.7	44.1	13.7	41.7	54.1	25.7	59.2	67.9					

397 This section showcases our experimental setup and analyzes both aggregate and task-specific level
398 results. Subtask-level performance details are available in Appendix D.1.

400

5.1 EXPERIMENTAL SETUPS

401 **Baselines** We evaluate three groups of zero-shot baselines: (1) **Multimodal Backbone**: Qwen2-VL (Wang et al., 2024a) and Qwen2.5-VL (Bai et al., 2025); (2) **Multimodal Retrieval**: VLM2Vec-Qwen2VL-2B (Jiang et al., 2025), UniSE-MLLM (Liu et al., 2025b), etc.; (3) **Multimodal Trajectory Planning**: UGround-V1-2B (Gou et al., 2024), UI-TARS-2B-SFT (Qin et al., 2025), ShowUI-2B (Lin et al., 2024a), etc. Model descriptions and evaluation setup are in Appendix D.2.

407 **Implementation Details** Data are annotated with `gpt-4o-mini-2024-07-18`. We train a
408 LoRA-tuned Qwen2-VL-Instruct retriever on GAE-Bench-lite for 256 steps using GradCache and
409 long-context inputs on 16×800 GPUs; UI-graph token selection is enabled only during training for
410 positional consistency. For more implementation details, please refers to Appendix C.2.

412

5.2 OVERALL EVALUATION

413

414 Table 3 summarizes Recall@1/5/10 of all baseline models across five datasets. Our proposed
415 method, **GAE-Retriever**, surpasses all baselines by a substantial margin on all evaluation metrics. Its
416 consistently outstanding performance over all datasets reflects the robustness and reliability of the
417 proposed benchmark for evaluating trajectory retrieval. A task-wise breakdown (Recall@5 by task
418 under in-domain and out-of-domain settings) is provided in Appendix D.4 and visualized in Figure 8.

419 **Multimodal Backbone Models** The two backbone models, Qwen2-VL-2B and Qwen2.5-VL-3B,
420 display relatively low retrieval performance compared to their understanding and reasoning skills.
421 Their Recall@1 scores remain below 4.0 across all datasets, indicating the need for task-specific
422 adaptation to develop effective retrieval abilities. Interestingly, Qwen2.5-VL-3B performs slightly
423 worse than the smaller Qwen2-VL-2B on most Recall@K metrics, underscoring a substantial gap
424 between current multimodal LLM pretraining and the demands of trajectory-based retrieval.

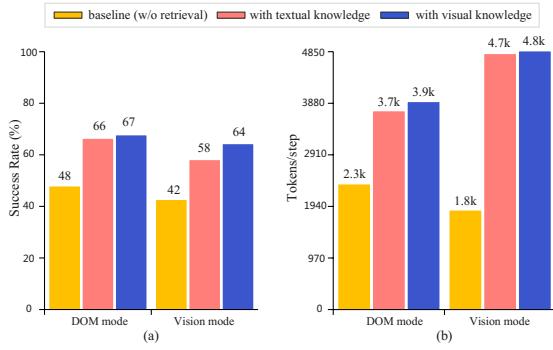
425 **Multimodal Retrieval Models** Across retrieval-focused baselines, **VLM2Vec-v2.2** attains the best
426 and most stable results, consistently surpassing **VLM2Vec-Qwen2VL-2B**. Overall, the VLM2Vec
427 family outperforms other methods, suggesting that diverse fused-modality training with interleaved
428 batches is key to retrieval effectiveness. In contrast, **LamRA-Ret** (text-image only) trails models
429 exposed to structured visual content, such as **ColQwen2-v1.0** and **GME-Qwen2VL-2B**. **GME-Qwen2VL-2B**
430 further outperforms **ColQwen2-v1.0**, while **UniSE-MLLM** (screenshot-focused) is
431 the weakest, sometimes below backbone baselines. Further analysis can be found in Appendix D.3.

432 **Multimodal Trajectory Planning Models** Methods in this group (e.g., ShowUI-2B, TongUI-3B, UGround-V1-2B) show no significant performance improvement over the backbone models. Their modest retrieval results imply that capabilities in planning or grounding generation do not directly translate to multimodal retrieval proficiency. Additionally, the difference in model size, i.e., 2B vs. 3B, appears to have minimal impact in our experiment.

433 **Trajectory Retrieval Models** GAE-Retriever achieves the best results across all datasets and 434 evaluation metrics. Notably, it reaches a Recall@10 of 76.3 on AutoWebGLM and 67.9 on GUIAct, 435 with Recall@1 consistently above 10.0 across all benchmarks. Compared to the strongest baseline, 436 VLM2Vec-V2.2, GAE-Retriever improves Recall@1 by up to 7.1 points on GUIAct and 6.4 points 437 on AutoWebGLM. These gains validate the design of our training strategy.

438 5.3 ONLINE WEB TASK EVALUATION

439 Our evaluation is conducted on Online-Mind2Web (Xue et al., 2025), which is markedly more 440 challenging for task execution and evaluation than offline benchmarks such as WebArena (Zhou et al., 441 2024). In total, we select 100 tasks spanning 51 websites for comparative evaluation; the full task 442 list and website details are provided in Appendix B.3. Experiments are performed under both DOM 443 and Vision observation modes across three representative methods. Figure 6 presents the overall task 444 success rates and the average number of tokens consumed per step, while Table 4 reports task success 445 rates and average execution steps broken down by task difficulty levels. Furthermore, Appendix D.7 446 analyzes the inference efficiency of incorporating GAE-Retriever: each step adds $\sim 10\%$ latency 447 (+0.38sec), the reranking stage takes 3.5sec, and enabling caching yields a $9.24\times$ speedup.



448 Figure 6: Online-Mind2Web evaluation: baseline vs. models augmented with textual/visual knowledge.

449 Table 4: Results in DOM and Vision modes.

Method	Success Rate (%)			#Average	
	Easy	Medium	Hard	Step	Reward
DOM mode					
baseline	70.45	35.89	21.42	8.72	7.60
+ $\mathcal{D}_{\text{text}}$	79.54	61.53	35.71	7.47	8.19
+ $\mathcal{D}_{\text{vision}}$	77.27	58.97	64.28	10.63	7.96
Vision mode					
baseline	59.09	28.20	28.57	14.88	7.35
+ $\mathcal{D}_{\text{text}}$	77.27	43.58	35.71	16.37	8.08
+ $\mathcal{D}_{\text{vision}}$	79.50	56.41	35.71	19.85	8.29
SeeAct					
SeeAct	71.42	47.36	6.25	18.45	-
+ $\mathcal{D}_{\text{text}}$	78.57	55.26	18.75	22.87	-
+ $\mathcal{D}_{\text{vision}}$	76.19	60.52	25.00	21.92	-

470 **Why Retrieval Matters: Multimodal Retrieval vs. None-Retrieval Baseline** Compared with 471 the non-retrieval setting, the multimodal retrieval-augmented approach achieves substantial gains 472 under both observation modes (Figure 6), with improvements of up to 22%. As shown in Table 4, the 473 gains are particularly pronounced on medium and hard tasks, effectively alleviating the performance 474 collapse caused by longer action sequences and more complex requirements. Appendix F provides 475 detailed execution cases along with side-by-side comparisons between retrieval and non-retrieval 476 settings. We further compare our method with a purely text-based trajectory retrieval approach 477 ($q \rightarrow \tau$) (Kim et al., 2024) in Appendix D.5, and observe that it is less effective than multimodal 478 retrieval. In contrast, our model-based retrieval method (Gu et al., 2024) more accurately aligns with 479 the decision needs of $(q, s) \rightarrow s'$, thereby reducing the noise introduced by irrelevant exemplars. To 480 verify the effectiveness of trajectory retrieval across different state-of-the-art agent frameworks, we 481 further find that introducing trajectory knowledge into the SeeAct framework (Zheng et al., 2024) 482 improves the overall success rate by 8-10%, with particularly large gains on medium and hard tasks 483 (shown in Appendix D.6). This demonstrates that our method is highly scalable.

484 **Which Knowledge Format Helps More: Textual or Visual?** Table 4 shows that visual exemplars 485 slightly outperform textual ones across both observation modes, with a clearer margin on hard tasks 486 , highlighting the importance of visual anchors (e.g., elements, button shapes, relative positions)

486 for cross-site transfer. For efficiency, textual guidance lowers steps and token usage by providing
 487 explicit rules and clearer goals, whereas visual guidance prompts more exploratory alignment, raising
 488 token use but improving robustness. [Appendix F](#) provides case studies and qualitative analyses, with
 489 [Appendix F.5](#) offering an in-depth discussion of how retrieved knowledge helps address hard tasks.
 490

491 6 CONCLUSION

492
 493 In this work, we introduce the task of Multimodal Trajectory Retrieval and present UATD, a unified-
 494 format dataset of real-world, GUI-based agent trajectories. Building on this, we construct two
 495 standardized benchmarks, GAE-Bench and GAE-Bench-lite, to support evaluation. To address the
 496 challenges of long multimodal sequences, we propose GAE-Retriever, a VLM2Vec-based retrieval
 497 framework enhanced with token selection and GradCache for efficient contrastive learning. Experi-
 498 mental results show that GAE-Retriever achieves the best performance across all five environments.
 499 Compared to the strongest retrieval baseline (VLM2Vec-V2.2), it improves Recall@1 by up to 12.9
 500 points. We introduce WebRAGent, a retrieval-augmented web agent, achieving 15-22% gains over
 501 non-retrieval baselines on the Online-Mind2Web benchmark. More importantly, as an ongoing effort,
 502 our recipe for constructing a multimodal trajectory retrieval system lays the groundwork for future
 503 investigations into retrieval-based context learning, reinforcement learning, and world modeling.
 504

505 **Limitations and Future Work** Our research also has several limitations. First, although we
 506 position the learned trajectory representation as a foundation for broader agentic capabilities (e.g.,
 507 world modeling, RL training), in this paper we primarily validate it through retrieval-augmented
 508 planning, and its effectiveness in other downstream tasks remains to be established. Second, our
 509 experiments are currently concentrated on web-agent data; while the methodology is in principle
 510 generalizable, we have not yet scaled training and evaluation to desktop, mobile, or embodied
 511 environments, which limits our ability to assess cross-platform generalization. Finally, our intention
 512 in positioning this paper is to highlight future exploration directions for the research community,
 513 encouraging further investigation into this valuable yet often overlooked area.

514 REPRODUCIBILITY STATEMENT

515 We have made every effort to ensure that the results reported in this paper are reproducible. All
 516 code and datasets are publicly available in an anonymous repository¹ to facilitate replication and
 517 verification. We provide detailed descriptions of the experimental setup, including training procedures,
 518 model configurations, and hardware specifications. In addition, the appendix contains numerous
 519 task execution examples and side-by-side comparisons. We believe these measures will enable other
 520 researchers to reproduce our work and further advance the field.
 521

523 REFERENCES

524
 525 Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang,
 526 Shijie Wang, Jun Tang, et al. Qwen2. 5-vl technical report. *ArXiv preprint*, abs/2502.13923, 2025.
 527 URL <https://arxiv.org/abs/2502.13923>.

528 Yuxiang Chai, Siyuan Huang, Yazhe Niu, Han Xiao, Liang Liu, Dingyu Zhang, Peng Gao, Shuai Ren,
 529 and Hongsheng Li. Amex: Android multi-annotation expo dataset for mobile gui agents. *ArXiv
 530 preprint*, abs/2407.17490, 2024. URL <https://arxiv.org/abs/2407.17490>.

531 Wentong Chen, Junbo Cui, Jinyi Hu, Yujia Qin, Junjie Fang, Yue Zhao, Chongyi Wang, Jun Liu,
 532 Guirong Chen, Yupeng Huo, et al. Guicourse: From general vision language models to versatile
 533 gui agents. *ArXiv preprint*, abs/2406.11317, 2024. URL <https://arxiv.org/abs/2406.11317>.

534
 535 Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samual Stevens, Boshi Wang, Huan Sun, and Yu Su.
 536 Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing
 537 Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023,
 538 New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

539
 1<https://anonymous.4open.science/r/7sy7CFD5>

540 Yang Deng, Xuan Zhang, Wenzuan Zhang, Yifei Yuan, See-Kiong Ng, and Tat-Seng Chua. On the
 541 multi-turn instruction following for conversational web agents. *ArXiv preprint*, abs/2402.15057,
 542 2024. URL <https://arxiv.org/abs/2402.15057>.

543 Manuel Faysse, Hugues Sibille, Tony Wu, Bilel Omrani, Gautier Viaud, Céline Hudelot, and Pierre
 544 Colombo. Colpali: Efficient document retrieval with vision language models. In *The Thirteenth*
 545 *International Conference on Learning Representations*, 2024.

546 Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip
 547 Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. In
 548 *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information*
 549 *Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

550 Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae,
 551 and Honglak Lee. Autoguide: Automated generation and selection of context-aware guidelines
 552 for large language model agents. In *Advances in Neural Information Processing Systems 38:*
 553 *Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver,*
 554 *BC, Canada, December 10 - 15, 2024*, 2024.

555 Luyu Gao, Yunyi Zhang, Jiawei Han, and Jamie Callan. Scaling deep contrastive learning batch size
 556 under memory limited setup. In *Proceedings of the 6th Workshop on Representation Learning*
 557 *for NLP (Rep4NLP-2021)*, pp. 316–321, 2021. doi: 10.18653/v1/2021.repl4nlp-1.31. URL
 558 <https://aclanthology.org/2021.repl4nlp-1.31>.

559 Boyu Gou, Ruohan Wang, Boyuan Zheng, Yanan Xie, Cheng Chang, Yiheng Shu, Huan Sun, and
 560 Yu Su. Navigating the digital world as humans do: Universal visual grounding for gui agents.
 561 *ArXiv preprint*, abs/2410.05243, 2024. URL <https://arxiv.org/abs/2410.05243>.

562 Anirudh Goyal, Abram L. Friesen, Andrea Banino, Theophane Weber, Nan Rosemary Ke, Adrià Puig-
 563 domènec Badia, Arthur Guez, Mehdi Mirza, Peter C. Humphreys, Ksenia Konyushkova, Michal
 564 Valko, Simon Osindero, Timothy P. Lillicrap, Nicolas Heess, and Charles Blundell. Retrieval-
 565 augmented reinforcement learning. In *International Conference on Machine Learning, ICML*
 566 *2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning*
 567 *Research*, pp. 7740–7765, 2022. URL <https://proceedings.mlr.press/v162/goyal22a.html>.

568 Yu Gu, Kai Zhang, Yuting Ning, Boyuan Zheng, Boyu Gou, Tianci Xue, Cheng Chang, Sanjari
 569 Srivastava, Yanan Xie, Peng Qi, et al. Is your llm secretly a world model of the internet?
 570 model-based planning for web agents. *ArXiv preprint*, abs/2411.06559, 2024. URL <https://arxiv.org/abs/2411.06559>.

571 Peter Conway Humphreys, Arthur Guez, Olivier Tielemans, Laurent Sifre, Theophane Weber, and
 572 Timothy P. Lillicrap. Large-scale retrieval for reinforcement learning. In *Advances in Neural*
 573 *Information Processing Systems 35: Annual Conference on Neural Information Processing Systems*
 574 *2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

575 Ting Jiang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, Qi Zhang, Deqing
 576 Wang, and Fuzhen Zhuang. E5-v: Universal embeddings with multimodal large language models.
 577 *ArXiv preprint*, abs/2407.12580, 2024. URL <https://arxiv.org/abs/2407.12580>.

578 Ziyan Jiang, Rui Meng, Xinyi Yang, Semih Yavuz, Yingbo Zhou, and Wenhui Chen. VLM2vec:
 579 Training vision-language models for massive multimodal embedding tasks. In *The Thirteenth*
 580 *International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=TEOKOzWYAF>.

581 Tomoyuki Kagaya, Thong Jing Yuan, Yuxuan Lou, Jayashree Karlekar, Sugiri Pranata, Akira Kinose,
 582 Koki Oguri, Felix Wick, and Yang You. Rap: Retrieval-augmented planning with contextual
 583 memory for multimodal llm agents. *ArXiv preprint*, abs/2402.03610, 2024. URL <https://arxiv.org/abs/2402.03610>.

584 Jikun Kang, Romain Laroche, Xingdi Yuan, Adam Trischler, Xue Liu, and Jie Fu. Think before
 585 you act: Decision transformers with working memory. In *Forty-first International Conference*
 586 *on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*, 2024. URL <https://openreview.net/forum?id=PSQ5Z920M8>.

594 Minsoo Kim, Victor Burszty, Eunyee Koh, Shunan Guo, and Seung-won Hwang. RaDA: Retrieval-
 595 augmented web agent planning with LLMs. In *Findings of the Association for Computational*
 596 *Linguistics: ACL 2024*, pp. 13511–13525, 2024. doi: 10.18653/v1/2024.findings-acl.802.

597

598 Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Chong Lim, Po-Yu Huang, Graham
 599 Neubig, Shuyan Zhou, Ruslan Salakhutdinov, and Daniel Fried. Visualwebarena: Evaluating
 600 multimodal agents on realistic visual web tasks. *ArXiv preprint*, abs/2401.13649, 2024a. URL
 601 <https://arxiv.org/abs/2401.13649>.

602

603 Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language
 604 model agents. *ArXiv preprint*, abs/2407.01476, 2024b. URL <https://arxiv.org/abs/2407.01476>.

605

606 Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen
 607 Zhang, Xiaohan Zhang, Yuxiao Dong, and Jie Tang. Autowebglm: A large language model-based
 608 web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Dis-
 609 covery and Data Mining, KDD 2024, Barcelona, Spain, August 25-29, 2024*, pp. 5295–5306, 2024.
 610 doi: 10.1145/3637528.3671620. URL <https://doi.org/10.1145/3637528.3671620>.

611

612 Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos,
 613 Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot
 614 parsing as pretraining for visual language understanding. In *International Conference on Machine
 615 Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of
 616 Machine Learning Research*, pp. 18893–18912, 2023. URL <https://proceedings.mlr.press/v202/lee23g.html>.

617

618 Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. BLIP: bootstrapping language-
 619 image pre-training for unified vision-language understanding and generation. In *International
 620 Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*,
 621 volume 162 of *Proceedings of Machine Learning Research*, pp. 12888–12900, 2022. URL
 622 <https://proceedings.mlr.press/v162/li22n.html>.

623

624 Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. BLIP-2: bootstrapping language-
 625 image pre-training with frozen image encoders and large language models. In *International
 626 Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume
 627 202 of *Proceedings of Machine Learning Research*, pp. 19730–19742, 2023. URL <https://proceedings.mlr.press/v202/li23q.html>.

628

629 Zhangheng Li, Keen You, Haotian Zhang, Di Feng, Harsh Agrawal, Xiujun Li, Mohana Prasad Sathya
 630 Moorthy, Jeff Nichols, Yinfei Yang, and Zhe Gan. Ferret-ui 2: Mastering universal user interface
 631 understanding across platforms. *ArXiv preprint*, abs/2410.18967, 2024. URL <https://arxiv.org/abs/2410.18967>.

632

633 Kevin Qinghong Lin, Linjie Li, Difei Gao, Zhengyuan Yang, Shiwei Wu, Zechen Bai, Weixian
 634 Lei, Lijuan Wang, and Mike Zheng Shou. Showui: One vision-language-action model for gui
 635 visual agent. *ArXiv preprint*, abs/2411.17465, 2024a. URL <https://arxiv.org/abs/2411.17465>.

636

637 Sheng-Chieh Lin, Chankyu Lee, Mohammad Shoeybi, Jimmy Lin, Bryan Catanzaro, and Wei Ping.
 638 Mm-embed: Universal multimodal retrieval with multimodal llms. *ArXiv preprint*, abs/2411.02571,
 639 2024b. URL <https://arxiv.org/abs/2411.02571>.

640

641 Guangyi Liu, Pengxiang Zhao, Liang Liu, Zhiming Chen, Yuxiang Chai, Shuai Ren, Hao Wang,
 642 Shibo He, and Wencho Meng. Learnact: Few-shot mobile gui agent with a unified demonstration
 643 benchmark. *ArXiv preprint*, abs/2504.13805, 2025a. URL <https://arxiv.org/abs/2504.13805>.

644

645 Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in
 646 Neural Information Processing Systems 36: Annual Conference on Neural Information Processing
 647 Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

648 Yikun Liu, Pingan Chen, Jiayin Cai, Xiaolong Jiang, Yao Hu, Jiangchao Yao, Yanfeng Wang, and
 649 Weidi Xie. Lamra: Large multimodal model as your advanced retrieval assistant. *ArXiv preprint*,
 650 abs/2412.01720, 2024. URL <https://arxiv.org/abs/2412.01720>.

651

652 Ze Liu, Zhengyang Liang, Junjie Zhou, Zheng Liu, and Defu Lian. Any information is just worth
 653 one single screenshot: Unifying search with visualized information retrieval. *ArXiv preprint*,
 654 abs/2502.11431, 2025b. URL <https://arxiv.org/abs/2502.11431>.

655

656 Xing Han Lu, Zdenek Kasner, and Siva Reddy. Weblinx: Real-world website navigation with
 657 multi-turn dialogue. In *Forty-first International Conference on Machine Learning, ICML 2024*,
 658 Vienna, Austria, July 21-27, 2024, 2024. URL <https://openreview.net/forum?id=mUSPhG4uDW>.

659

660 Rui Meng, Ziyan Jiang, Ye Liu, Mingyi Su, Xinyi Yang, Yuepeng Fu, Can Qin, Zeyuan Chen, Ran
 661 Xu, Caiming Xiong, Yingbo Zhou, Wenhui Chen, and Semih Yavuz. Vlm2vec-v2: Advancing
 662 multimodal embedding for videos, images, and visual documents. *ArXiv preprint*, abs/2507.04590,
 663 2025. URL <https://arxiv.org/abs/2507.04590>.

664

665 Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. MTEB: Massive text embed-
 666 ding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association*
 667 *for Computational Linguistics*, pp. 2014–2037, 2023. doi: 10.18653/v1/2023.eacl-main.148. URL
 668 <https://aclanthology.org/2023.eacl-main.148>.

669

670 Joon Sung Park, Joseph O’Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S
 671 Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th*
 672 *annual ACM symposium on user interface software and technology*, pp. 1–22, 2023.

673

674 Zhiliang Peng, Wenhui Wang, Li Dong, Yaru Hao, Shaohan Huang, Shuming Ma, and Furu
 675 Wei. Kosmos-2: Grounding multimodal large language models to the world. *ArXiv preprint*,
 676 abs/2306.14824, 2023. URL <https://arxiv.org/abs/2306.14824>.

677

678 Pranav Putta, Edmund Mills, Naman Garg, Sumeet Motwani, Chelsea Finn, Divyansh Garg, and
 679 Rafael Rafailov. Agent q: Advanced reasoning and learning for autonomous ai agents. *ArXiv*
 680 preprint, abs/2408.07199, 2024. URL <https://arxiv.org/abs/2408.07199>.

681

682 Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao
 683 Li, Yunxin Li, Shijue Huang, et al. Ui-tars: Pioneering automated gui interaction with native agents.
 684 *ArXiv preprint*, abs/2501.12326, 2025. URL <https://arxiv.org/abs/2501.12326>.

685

686 Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal,
 687 Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever.
 688 Learning transferable visual models from natural language supervision. In *Proceedings of the*
 689 *38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual*
 690 *Event*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763, 2021. URL
 691 <http://proceedings.mlr.press/v139/radford21a.html>.

692

693 David Raposo, Sam Ritter, Blake Richards, Timothy Lillicrap, Peter Conway Humphreys, and Adam
 694 Santoro. Mixture-of-depths: Dynamically allocating compute in transformer-based language
 695 models. *ArXiv preprint*, abs/2404.02258, 2024. URL <https://arxiv.org/abs/2404.02258>.

696

697 Christopher Rawles, Alice Li, Daniel Rodriguez, Oriana Riva, and Timothy P. Lillicrap. An-
 698 droidinthewild: A large-scale dataset for android device control. In *Advances in Neural Infor-
 699 mation Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023*,
 700 *NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

701

702 Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, Jonathan Waltz, Gabrielle Lau, Marybeth
 703 Fair, Alice Li, William Bishop, Wei Li, Folawiyo Campbell-Ajala, et al. Androidworld: A dynamic
 704 benchmarking environment for autonomous agents. *ArXiv preprint*, abs/2405.14573, 2024. URL
 705 <https://arxiv.org/abs/2405.14573>.

702 Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew J.
 703 Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. In *9th*
 704 *International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May*
 705 *3-7, 2021, 2021*. URL <https://openreview.net/forum?id=0IOX0YcCdTn>.

706 Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, and Insup Lee. Regent: A retrieval-
 707 augmented generalist agent that can act in-context in new environments. *ArXiv preprint*,
 708 [abs/2412.04759](https://arxiv.org/abs/2412.04759), 2024. URL <https://arxiv.org/abs/2412.04759>.

709 Yu Su, Diyi Yang, Shunyu Yao, and Tao Yu. Language agents: Foundations, prospects, and risks.
 710 In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pp. 17–24, 2024. doi: 10.18653/v1/2024.emnlp-tutorials.3.

711 Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and
 712 Jie Zhou. COIN: A large-scale dataset for comprehensive instructional video analysis. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 1207–1216, 2019. doi: 10.1109/CVPR.2019.00130.

713 Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu,
 714 Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model’s perception of the
 715 world at any resolution. *ArXiv preprint*, [abs/2409.12191](https://arxiv.org/abs/2409.12191), 2024a. URL <https://arxiv.org/abs/2409.12191>.

716 Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Guo Chen, Baoqi Pei, Rongkun Zheng,
 717 Zun Wang, Yansong Shi, et al. Internvideo2: Scaling foundation models for multimodal video
 718 understanding. In *European Conference on Computer Vision*, pp. 396–416. Springer, 2024b.

719 Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *ArXiv preprint*, [abs/2409.07429](https://arxiv.org/abs/2409.07429), 2024c. URL <https://arxiv.org/abs/2409.07429>.

720 Cong Wei, Yang Chen, Haonan Chen, Hexiang Hu, Ge Zhang, Jie Fu, Alan Ritter, and Wenhui Chen.
 721 Uniir: Training and benchmarking universal multimodal information retrievers. In *European Conference on Computer Vision*, pp. 387–404. Springer, 2024.

722 Jian Xie, Kai Zhang, Jiangjie Chen, Tinghui Zhu, Renze Lou, Yuandong Tian, Yanghua Xiao, and
 723 Yu Su. Travelplanner: A benchmark for real-world planning with language agents. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*,
 724 2024a. URL <https://openreview.net/forum?id=15XQzNkAOe>.

725 Tianbao Xie, Danyang Zhang, Jixuan Chen, Xiaochuan Li, Siheng Zhao, Ruisheng Cao, Toh Jing
 726 Hua, Zhoujun Cheng, Dongchan Shin, Fangyu Lei, Yitao Liu, Yiheng Xu, Shuyan Zhou, Silvio
 727 Savarese, Caiming Xiong, Victor Zhong, and Tao Yu. Osworld: Benchmarking multimodal agents
 728 for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024b.

729 Yiheng Xu, Zekun Wang, Junli Wang, Dunjie Lu, Tianbao Xie, Amrita Saha, Doyen Sahoo, Tao Yu,
 730 and Caiming Xiong. Aguvis: Unified pure vision agents for autonomous gui interaction. *ArXiv preprint*, [abs/2412.04454](https://arxiv.org/abs/2412.04454), 2024. URL <https://arxiv.org/abs/2412.04454>.

731 Tianci Xue, Weijian Qi, Tianneng Shi, Chan Hee Song, Boyu Gou, Dawn Song, Huan Sun, and
 732 Yu Su. An illusion of progress? assessing the current state of web agents, 2025. URL <https://arxiv.org/abs/2504.01382>.

733 Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable
 734 real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

735 Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R. Narasimhan, and Yuan
 736 Cao. React: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*, 2023. URL
 737 https://openreview.net/pdf?id=WE_vluYUL-X.

756 Junpeng Yue, Xinru Xu, Börje F Karlsson, and Zongqing Lu. Mllm as retriever: Interactively
 757 learning multimodal retrieval for embodied agents. *ArXiv preprint*, abs/2410.03450, 2024. URL
 758 <https://arxiv.org/abs/2410.03450>.

759 Bofei Zhang, Zirui Shang, Zhi Gao, Wang Zhang, Rui Xie, Xiaojian Ma, Tao Yuan, Xinxiao Wu, Song-
 760 Chun Zhu, and Qing Li. Tongui: Building generalized gui agents by learning from multimodal web
 761 tutorials. *ArXiv preprint*, abs/2504.12679, 2025. URL <https://arxiv.org/abs/2504.12679>.

762 Chaoyun Zhang, Shilin He, Jiaxu Qian, Bowen Li, Liqun Li, Si Qin, Yu Kang, Minghua Ma, Guyue
 763 Liu, Qingwei Lin, et al. Large language model-brained gui agents: A survey. *ArXiv preprint*,
 764 abs/2411.18279, 2024a. URL <https://arxiv.org/abs/2411.18279>.

765 Kai Zhang, Yi Luan, Hexiang Hu, Kenton Lee, Siyuan Qiao, Wenhui Chen, Yu Su, and Ming-Wei
 766 Chang. Magiclens: Self-supervised image retrieval with open-ended instructions. In *Forty-first*
 767 *International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*,
 768 2024b. URL <https://openreview.net/forum?id=Zc22RDtsvP>.

769 Xin Zhang, Yanzhao Zhang, Wen Xie, Mingxin Li, Ziqi Dai, Dingkun Long, Pengjun Xie, Meishan
 770 Zhang, Wenjie Li, and Min Zhang. Gme: Improving universal multimodal retrieval by multi-
 771 modal llms. *ArXiv preprint*, abs/2412.16855, 2024c. URL <https://arxiv.org/abs/2412.16855>.

772 Xuan Zhang, Yang Deng, Zifeng Ren, See-Kiong Ng, and Tat-Seng Chua. Ask-before-plan: Proactive
 773 language agents for real-world planning. *ArXiv preprint*, abs/2406.12639, 2024d. URL <https://arxiv.org/abs/2406.12639>.

774 Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. Expel: LLM
 775 agents are experiential learners. In *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI*
 776 *2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024*,
 777 *Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February*
 778 *20-27, 2024, Vancouver, Canada*, pp. 19632–19642, 2024. doi: 10.1609/AAAI.V38I17.29936.
 779 URL <https://doi.org/10.1609/aaai.v38i17.29936>.

780 Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v(ision) is a generalist web
 781 agent, if grounded, 2024. URL <https://arxiv.org/abs/2401.01614>.

782 Boyuan Zheng, Michael Y Fatemi, Xiaolong Jin, Zora Zhiruo Wang, Apurva Gandhi, Yueqi Song,
 783 Yu Gu, Jayanth Srinivasa, Gaowen Liu, Graham Neubig, et al. Skillweaver: Web agents can
 784 self-improve by discovering and honing skills. *ArXiv preprint*, abs/2504.07079, 2025. URL
 785 <https://arxiv.org/abs/2504.07079>.

786 Shuyan Zhou, Li Zhang, Yue Yang, Qing Lyu, Pengcheng Yin, Chris Callison-Burch, and Graham
 787 Neubig. Show me more details: Discovering hierarchies of procedures from semi-structured web
 788 data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*
 789 *(Volume 1: Long Papers)*, pp. 2998–3012, 2022. doi: 10.18653/v1/2022.acl-long.214. URL
 790 <https://aclanthology.org/2022.acl-long.214>.

791 Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng,
 792 Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic
 793 web environment for building autonomous agents. In *The Twelfth International Conference on*
 794 *Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*, 2024. URL <https://openreview.net/forum?id=oKn9c6ytLx>.

795

803

804 **A THE USE OF LLMs**

805

806 We used a large language model (LLM) to assist with linguistic polishing of the manuscript, including
 807 refining expressions, enhancing textual coherence, improving readability, and performing sentence
 808 rewrites and grammar checking. In this work, the LLM did not participate in research ideation,
 809 method design, or experimental analysis; all scientific content was completed independently by the
 authors. The LLM’s role was strictly limited to language-level improvements. The authors take full

810 responsibility for the entire content and have ensured that the use of the LLM adheres to academic
 811 ethics, precluding plagiarism and other inappropriate practices.
 812

813 B DATASET DETAILS

814 B.1 DETAILS OF THE UNIFIED AGENT TRAJECTORY DATASET

815 This subsection introduces the methodology for constructing the Unified Agent Trajectory Dataset
 816 and presents its statistical properties. The real-world human trajectories and their action definitions
 817 in this dataset are derived from the following five GUI datasets: Mind2Web (Deng et al., 2023),
 818 AutoWebGLM (Lai et al., 2024), WebArena (Zhou et al., 2024; Koh et al., 2024a), WebLINX (Lu
 819 et al., 2024), and GUIAct (Chen et al., 2024). The raw data is manually cleaned by removing invalid
 820 trajectories and corrupted states. For sources without native visual observations (e.g., AutoWebGLM),
 821 we first complete the HTML using `gpt-4o-mini`, and then render the content via Playwright to
 822 produce screenshot-based observations. Additionally, textual descriptions are generated for each state,
 823 conditioned on the corresponding screenshot, to facilitate subsequent retrieval tasks. The prompts
 824 used for this annotation process and definition of action spaces are listed in Appendix E.1.
 825

826 Consequently, we obtain the Unified Agent Trajectory Dataset, with statistics shown in Table 5. In
 827 this dataset, all screenshots are preserved at their original resolution, and target elements are annotated
 828 using bounding boxes.
 829

830 Table 5: Statistics of the Unified Agent Trajectory Dataset (UATD). For each data source, we include the total
 831 number of tasks along with the average, minimum, maximum, and total number of states.
 832

833 Source	834 Task	835 State			
		836 Min	837 Max	838 Avg	839 Total
Mind2Web	1,468	2	31	6.65	9,621
AutoWebGLM	140	1	34	4.43	620
WebArena	201	1	26	6.07	1,221
WebLINX	485	1	68	14.19	6,882
GUIAct	5,453	1	85	11.82	64,449

840 B.2 GAE-BENCH-LITE / (MINI) DETAILS

841 GAE-Bench-lite contains **563,900** positive pairs; Table 6 summarize its overall composition and
 842 the per-task, per-source distribution (Mind2Web, AutoWebGLM, WebArena, WebLINX, GUIAct).
 843 We further introduce GAE-Bench-lite (mini), which contains only the candidates relevant to the
 844 evaluation subset. Table 7 lists the candidate sets of GAE-Bench-lite.
 845

846 Table 6: Distribution of GAE-Bench-lite by retrieval tasks and data sources. This compact variant includes
 847 563,900 positive pairs.
 848

849 Retrieval Task	850 Mind2Web	851 AutoWebGLM	852 WebArena	853 WebLINX	854 GUIAct
$(q, \tau) \rightarrow \tau'$	13,630	718	1,580	4,376	67,968
$(q, \tau) \rightarrow s$	14,912	822	1,796	7,746	89,770
$q \rightarrow \tau$	7,566	792	1,026	1,242	17,040
$(q, s) \rightarrow s'$	16,306	960	2,040	12,794	117,992
$(q, s) \rightarrow \tau$	14,912	822	1,796	7,746	89,770
$q \rightarrow s$	10,943	760	1,305	6,337	48,433
Total	78,269	4,874	9,543	40,241	430,973

855 B.3 ONLINE MIND2WEB TASK LIST

856 This setting is more realistic and challenging but also much less stable. In preliminary runs over all 300
 857 tasks, we frequently encountered network failures, SSL errors, region-based access restrictions, login
 858 verification, and various anti-crawling or Turing-test mechanisms; despite engineering mitigations,
 859 robustness remained limited, and running multiple trials over all tasks would incur prohibitive token
 860 costs. For both cost and reproducibility, we therefore construct an online evaluation set of 100 tasks
 861 via a two-stage selection procedure: first, we run each of the 300 tasks at least three times on six
 862

864
865
866
867
868
869
870
871
872
873

Table 7: Overview of candidate sets in the GAE-Bench-lite

	Mind2Web	WebLINX	WebArena	GUIAct	AutoWebGLM	Total
GAE-Bench-lite						
State	9,475	5,852	1,104	42,980	620	60,031
Trajectory	1,261	207	171	2,840	132	4,611
Interval	44,323	48,663	5,882	408,311	2,401	509,580
GAE-Bench-lite (mini)						
State	2,842	2,127	588	10,530	349	16,436
Trajectory	239	67	58	484	48	896
Interval	15,237	26,191	3,093	46,515	1,720	92,756

different servers and retain 156 tasks that can be executed reliably; second, we perform stratified sampling over these 156 tasks to preserve website and task-category diversity and to match the easy/medium/hard difficulty distribution of the original dataset.

Our final evaluation set contains **100** tasks from **51** distinct websites. A detailed description of each task is provided below, and Figure 7 summarizes the distribution of websites in this set.

1. Add Elevate at Chicago, IL, to favorites and show a virtual tour.
2. Book 4 tickets in the upper for any Kevin Hart show in New York in the next three months and view ticket prices with estimated fees.
3. Browse iPhone X for sale that is in good condition, has a max price of 400, and searches in titles only.
4. Browse recipes for gluten-free chocolate chip cookies that can be made without nuts.
5. Check the hourly forecast for Boston.
6. Check the specifications of the best-selling HP FHD laptop with 16 GB RAM and core i7 running on Windows 11.
7. Compare the breeds Afghan Hound, Akita and Azawakh.
8. Display the figure comparing unemployment trends among women in Illinois and Michigan.
9. Estimate the federal income tax I would owe on \$158,500 of taxable income in ZIP code 97007, filing as single.
10. Find a condo for rent in Houston, TX, with a monthly rent of no more than 30% of an income of \$8000. The condo should have a minimum area of 600 square feet, and the move-in date is the 1st of next month.
11. Find a day-use park that offers horseback riding near Nashville.
12. Find a dog groomer for nail trimming within 100 miles of zip code 10005 and check the detailed service prices of the first one.
13. Find a list of houses for sale in zip code 85747 with a private pool.
14. Find a personal trainer service at 10040 for a 25-year-old client aiming to build muscle.
15. Find a recipe that includes eggplant and mushrooms.
16. Find a Single-Family House for Rent in Houston, TX with 1 bed.
17. Find a walkthrough for the game "The Legend of Zelda: Breath of the Wild" on ign.
18. Find an editor's choice review with a score of 10 in the boardgame category on ign.
19. Find and open the earliest press release.
20. Find discussions of the community and open one with the most replies on Flightaware.
21. find electricians near 10203.
22. Find Farms land in Wilkes County, NC with the lowest price.
23. Find healthy savory vegan snack recipes which can be cooked within 5 minutes and contain a high level of protein.
24. Find Linux platform software developers in 10080 who master the Python language and Java language with web interface project type.
25. Find obedience trials in state of New York during the month of May.
26. Find out the cold and flu forecast and today's air quality in Champaign, IL.
27. Find the closing stock price for Tesla on March 17, 2023.
28. Find the lowest-priced Student housing near Liverpool International College which has been priced between 100 to 300 pounds and has a private bathroom.
29. Find the most cited publication at the 2022 CVPR main conference.
30. Find the most frequent word that rhymes with "thought" and has three syllables.
31. Find the race time for who wins the first place in the last race of the 2023 Formula 1 (F1).
32. Find UA or AA flights from London to New York that arrive between 8:00 PM and 11:00 PM on FlightAware.
33. Get the frozen vegan cheese pizza between 5 to 10 USD on Target.
34. Get the report from the final environmental impact statement for the Jamaica Bus Depot expansion on new.mta.info.
35. Identify a pill with a pink color and oval shape with 894 5 number on it.
36. Look for the largest hunting land for auction in Kansas high plain region with mineral rights posted in the last seven days.

918 37. Open the page with an overview of the submission of releases on Discogs.
 919 38. Search for a beginner's course in computer science that includes advertisement skills.
 920 39. Search for a job in Miami, Florida, in Human Resources on target.
 921 40. Search for NordicTrack with the lowest price.
 922 41. Search for rentals in Corning, CA with a maximum price of \$1500.
 923 42. See the monthly forecast for Atlanta, GA.
 924 43. Show crazy credits for the movie "Prometheus" on IMDb.
 925 44. Show daily weather for New York City.
 926 45. Show houses for sale in Maryland with a maximum price of \$60,000.
 927 46. Show me the monthly weather forecast for Florida City.
 928 47. Show me the page with average wait times for U.S. citizens arriving at Raleigh-Durham International Airport on 2025-03-12.
 929 48. Show me the wind flow map for Belo Horizonte.
 930 49. Submit a request for vehicle registration renewal with title number X123456 and last 4 digits of VIN is 1234.
 931 50. View the cheapest apartment available for students at the University of Leeds with bills that include WIFI and cleaning services.
 932 51. What are the Nearby Attractions from the most popular attraction in Hong Kong?
 933 52. What is the ownership cost of the first car in the list "top buys 2025"?
 934 53. Check the current wind speed in Calgary, Alberta.
 935 54. Open the most helpful 5-star reviews of Alpine Ridge.
 936 55. Browse the first top news of Microsoft stock on Google Finance.
 937 56. Find the list of neighborhood maps for Brooklyn on new.mta.info.
 938 57. Find the park that offers the cheapest paddling permits.
 939 58. Browse Marriott Bonvoy credit cards on Marriott.
 940 59. Find the recommended dosage for Vivitrol.
 941 60. Show me the comparison of the first two personal credit cards that do not charge foreign transaction fees.
 942 61. Search for papers related to reinforcement learning under the topics of computer science and mathematics on arxiv, with recent
 943 submission dates between September 2024 and January 2025.
 944 62. Open the reviews of a recipe with beef sirloin.
 945 63. Can you show me the page with the filing fee for a self-petitioned I-140 application?
 946 64. Compare available plans for the AeroAPI on Flightaware.
 947 65. Find the interactions between Eulexin and hepatic dysfunction.
 948 66. Find the stock price for WWE over the last month.
 949 67. Find the next available dates for Alley Creek Camp.
 950 68. Find the latest climate news.
 951 69. Show me historical data for EUR/USD.
 952 70. Browse the list of Civil Division forms.
 953 71. Find the side effects of taking Montelukast.
 954 72. View the list of the Most Popular TV on rotten tomatoes.
 955 73. Find the tech specs of the MacBook Pro 16-inch introduced in November 2023.
 956 74. Show Teen Driver Safety program information.
 957 75. Add a \$100 Best Buy gift card for a birthday to my cart.
 958 76. Add a \$50 Uber gift card to the cart.
 959 77. Check the interaction between Novolin N and Novolin R.
 960 78. Find support services jobs in Bentonville, in the state of Arkansas.
 961 79. Find out what to do when I lose an item on a bus on us.megabus.
 962 80. Check the status of bus S92 for any disruptions on new.mta.info.
 963 81. Look up the current temperature for zip code 10019.
 964 82. Show me the rules and cancellation for Alley Spring.
 965 83. Open the XRP yearly chart.
 966 84. Find the weather for Vancouver, British Columbia for the next seven days.
 967 85. Find the next available date for Albion Basin.
 968 86. Browse the final skin in the list for the champion Ahri.
 969 87. Find a DMV center in Richmond.
 970 88. Check drug interaction for melatonin and Folate Forte.
 971 89. Identify the ongoing competition that offers the highest prize and find the code that received the most votes in that competition.
 90. Find the 5-day price chart for Bitcoin.

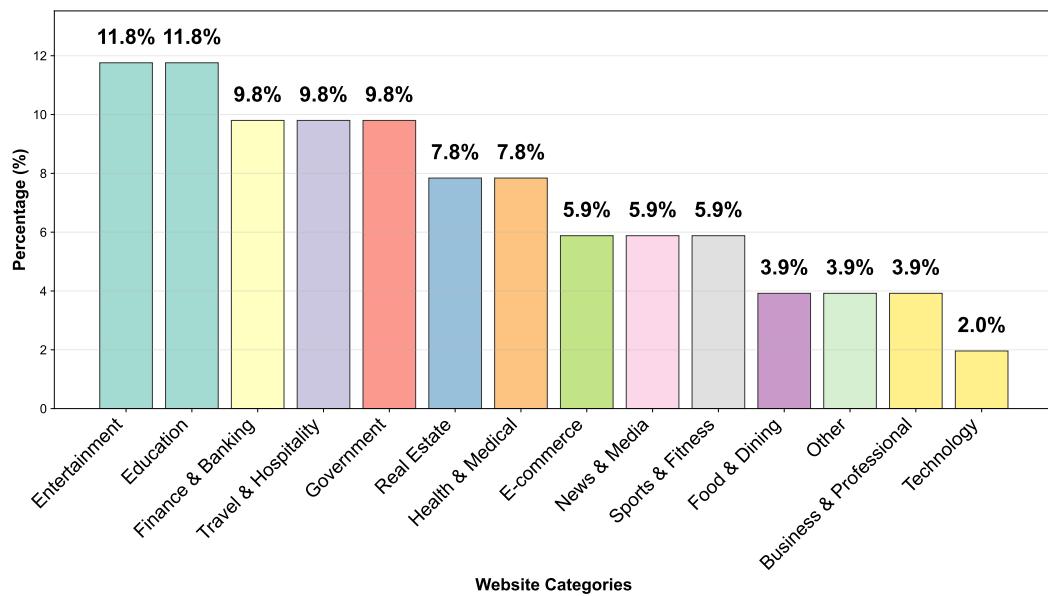


Figure 7: Distribution of website categories across all tasks.

91. Show theatre events for Las Vegas and select one.
 92. Browse Humira dosage information.
 93. Find the app for iOS.
 94. Show me recipes for pancakes with wheat and without beetroot.
 95. Check the current standings for MLS on Fox Sports.
 96. Find technical specs for the latest Macbook Air on Apple.
 97. Find the Drug Interaction Report for Viagra and alcohol.
 98. Browse the upcoming SuperBike events taking place in Italy.
 99. Find possible causes for the symptoms of chest pain which is sharp which is accompanied by anxiety.
 100. Take a weight management quiz to find a motivating article for a non-exercising, mostly eating out and can't control portions and cravings, and who has a strong support system, enjoys traveling, loves family time and cooking.

C IMPLEMENTATION DETAILS

C.1 INPUT SEQUENCE FORMATION

As described in Subsections 3.1 and 4.1, we represent a state s_i using a screenshot, and each action a_i is expressed as a JSON object with three keys: *operation*, *value*, and *target*. To support action modeling across diverse platforms, we standardize this JSON schema across environments; the *target* is specified by relative bounding-box coordinates in $[0, 1]$ (e.g., $(x_{\min}, y_{\min}, x_{\max}, y_{\max})$), and the action space definition is prepended as context.

A trajectory subsequence $\tau_{i:j}$ or a full trajectory $\tau_{1:n}$ (or simply τ) is composed of state-action pairs. To standardize the representation across different environments, we prepend the action space definition to $\tau_{i:j}$ and $\tau_{1:n}$, resulting in $u_{i:j}$ and $u_{1:n}$. For the target side of retrieval, \mathbf{v} can be either a single state s_i or a trajectory sequence with action definitions, denoted as $u_{i:j}$. For the query side \mathbf{k} , we prepend the retrieval query \tilde{q} , including a task-specific instruction and trajectory description to s_i or $u_{i:j}$. Table 8 provides concrete examples, where `[image]` denotes image tokens.

C.2 IMPLEMENTATION DETAILS OF THE RETRIEVER

We adopt the OpenAI model `gpt-4o-mini-2024-07-18` for data annotation. The retrieval model is based on Qwen2-VL-Instruct and trained with LoRA of rank 8 on 16 NVIDIA H800 GPUs. We employ GradCache during training with a sub-batch size of 1 and a total accumulated batch size of 2,048. The schedule consists of 256 steps on GAE-Bench-lite, summing to 1,044 GPU hours. Unless otherwise noted, we use a learning rate of 5×10^{-5} , a warm-up ratio of 5%, and an interleaved

1026 Table 8: Examples of model input representations for a state (s_i), an action (a_i), a trajectory sequence or
 1027 subsequence with action definitions ($u_{i:j}$), and an augmented retrieval query (\tilde{q}). The retrieval key \mathbf{k} consists of
 1028 \tilde{q} combined with either s_i or $u_{i:j}$, while the retrieval value \mathbf{v} is composed of either s_i or $u_{i:j}$.

1029 Variable	1030 Representation Instance
1030 s_i	1031 Observation: [image]
1031 a_i	1032 Action i : {"operation": "type", "value": "Elon Musk", "target": {"x": 0.5704, "y": 0.2142, "width": 0.3678, "height": 0.0663}}
1032 $u_{i:j}$	1033 Action Space: 1034 1. click: Simulates a mouse click on the target bounding box. 1035 2. type: Types the value (str) into the target bounding box. 1036 Positions are represented in relative coordinates within the range [0,1] on the observation screenshot. 1037 Observation 1: [image] 1038 Action 1: {"operation": "click", "value": null, "target": {"x": 0.0021, "y": 0.1424, "width": 0.0243, "height": 0.0519}} 1039 Observation 2: [image] 1040 Action 2: {"operation": "type", "value": "", "target": {"x": 0.8343, "y": 0.5659, "width": 0.1034, "height": 0.2496}}
1041 \tilde{q}	1042 Apply the request "Creating a template on Trello in a new tab." to the previous web navigation steps to derive the next trajectory.

1042 batch size of 0.2; training runs in Bfloat16 precision with a loss temperature of 0.02. The maximum
 1043 token length is 65,536 for both training and evaluation. We enable UI-graph-based token selection
 1044 only during training, as it adds no learnable parameters and preserves the positional structure of the
 1045 full token sequence; following Lin et al. (2024a), we apply a 0.5 mask ratio across all transformer
 1046 layers. We compute final sequence representations using EOS pooling followed by normalization.
 1047 For image preprocessing, we rely on *qwen-vl-utils*, which resizes inputs to tensors with spatial bins
 1048 of 28×28 and channel depth ranging from 4 to 1280, depending on content. Evaluation is conducted
 1049 on GAE-Bench-lite with the mini candidate sets using a batch size of 6 on 8 NVIDIA H800 GPUs,
 1050 requiring 22.5 GPU hours.

1051 C.3 IMPLEMENTATION DETAILS OF THE WEBRAGENT FRAMEWORK

1052 **Observation** The framework supports two independent observation modes: DOM (textualized
 1053 structure) and Vision (page screenshots). In DOM mode, we apply strict filtering to remove redundant
 1054 web-structure noise, selecting only interactive elements and exposing their tags and content. In Vision
 1055 mode, full-page screenshots capture cues missing from the DOM view.

1056 **Planning and Action** We use different planning models for the two observation modes. In
 1057 DOM mode, we adopt GPT-4.1, which excels at textual/structural understanding and supports
 1058 controllable reasoning traces. In Vision mode, we use OpenAI's computer-use-preview,
 1059 which performs screen understanding and pointer-style action decisions. The action space
 1060 covers: click, double_click, type, scroll, keypress, drag, search, wait,
 1061 get_final_answer. The detailed prompt for the planning model is shown in snippet 4.

1062 **Reward** Our framework's reward module evaluates tasks based on the historical trajectory and the
 1063 current web state. It uses the agent's thoughts, actions, and reflections to dynamically assess progress
 1064 and produce a quantitative score. The evaluator considers the full trace, accessibility-tree features,
 1065 and (when available) screenshot cues to rate the trajectory's validity and effectiveness on a discrete
 1066 scale 1, 3, 7, 9, 10: **1** indicates a loop requiring course correction, while **10** denotes a perfectly
 1067 completed task. Results are emitted as structured JSON containing the status, score, rationale, and
 1068 description, enabling clear feedback on current progress, next steps, and improvement directions to
 1069 efficiently guide subsequent actions.

1070 **Retrieve from the Step-Level Knowledge Base** To reuse experience more accurately and effi-
 1071 ciently, we employ the GAE-Retriever to encode the current-step state (task instruction + screenshot)
 1072 into a retrieval query for similarity matching, and then apply LLM re-ranking to improve match
 1073 precision, directly locating the most similar textual or visual guidance in the step-level knowledge
 1074 base. The knowledge base is step-aligned: each record represents an "observation → action" example.
 1075 It offers two guidance types. Visual guidance includes the step image plus a brief structured summary
 1076 (task description, action space, and action metadata). Textual guidance is generated with GPT-4.0 on
 1077 the Unified Agent Trajectory Dataset (UATD), providing detailed natural-language descriptions of

the observation and action. Detailed data examples are shown in Snippets 7–8. The knowledge base can be incrementally updated to rapidly adapt to UI changes.

D EXPERIMENTAL DETAILS

D.1 SUBTASK PERFORMANCE

In this section, we present the evaluation of all baseline methods described in Subsection 5.1, along with our proposed GAE-Retriever, on 12 retrieval subtasks spanning all data sources. Results are given in terms of Recall@1/5/10. Table 9 provides the outcomes under the in-domain setting, while Table 10 covers the out-of-domain evaluation.

Table 9: Recall@1/5/10 for all methods on each subtask in the in-domain setting.

Source	$(q, \tau_{1:n}) \rightarrow \tau_{i+1:n}$	$(q, \tau_{i+1:n}) \rightarrow \tau_{1:i}$	$(q, \tau_{1:n}) \rightarrow s_{i+1}$	$(q, \tau_{i+1:n}) \rightarrow s_i$	$q \rightarrow \tau_{\sim}$	$q \rightarrow \tau_{\sim}$	$(q, s_i) \rightarrow s_{i+1}$	$(q, s_{i+1}) \rightarrow s_i$	$(q, s_i) \rightarrow \tau_{i+1:n}$	$(q, s_{i+1}) \rightarrow \tau_{1:i}$	$q \rightarrow s_i$	$q \rightarrow s_n$
Qwen2-VL-2B												
Mind2Web	0.0/0.0/1.5	0.0/0.0/2.0	0.1/0.3/0.6	0.0/0.1/0.7	0.0/0.0/1.8	0.0/0.1/5.2	3.2/54.9/70.8	2.6/55.0/71.2	0.0/0.2/0.5	0.0/0.0/0.0	0.0/0.3/0.6	0.0/0.7/0.7
AutoWebGLM	0.0/0.0/0.0	2.0/2.0/4.0	0.0/0.0/0.0	7.1/17.0/35.7	8.9/16.1/39.3	0.0/0.0/2.0	0.0/0.0/2.0	2.0/6.0/8.0	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.0/0.0	3.3/3.3/3.3
WebArena	0.0/0.0/0.0	0.0/0.0/0.0	0.2/0.4/3.6	0.0/0.3/6.6	0.0/0.10/5.8	0.0/0.20/2.6	9.0/9.3/20.0	10.4/47.9/52.4	0.0/0.2/4.2	0.0/0.0/0.0	0.0/0.0/1.9	0.0/0.0/0.0
WebLINX	0.0/0.5/0.5	0.0/0.5/1.0	0.3/0.8/1.3	0.0/0.0/0.0	0.0/0.3/10.9	1.1/5.4/15.5	1.9/35.8/48.1	17.5/48.7/58.3	0.0/0.0/5.5	0.0/0.1/1.0	0.0/0.2/0.5	0.0/0.0/0.0
GUIAct	0.0/0.0/0.0	0.0/0.0/0.0	0.1/0.1/0.3	0.0/0.2/0.4	1.0/4.0/4.5	0.0/0.0/0.0	0.6/18.9/23.8	23.4/55.9/39.1	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.1/0.3	0.0/0.2/0.2
Qwen2.5-VL-3B												
Mind2Web	0.0/0.0/0.5	0.0/0.0/0.0	0.0/0.0/7.0	0.0/0.1/0.3	0.9/2.7/12.4	0.5/9.0/16.0	3.7/30.7/39.0	4.1/20.3/37.7	0.0/0.0/0.0	0.0/0.0/0.0	0.2/0.5/0.6	0.0/0.7/0.7
AutoWebGLM	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.2/0.4	0.0/0.2/0.4	3.6/14.3/21.4	1.8/14.3/21.6	0.0/0.4/0.8	0.0/0.0/2.0	0.0/0.0/0.0	0.0/0.2/0.2	0.0/1.8/2.6	0.0/3.3/6.7
WebArena	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.0/2.4	0.0/0.0/1.2	2.7/8.1/24.3	2.7/8.0/20.4	2.1/24.4/42.7	4.2/37.5/44.8	0.0/1.2/1.2	0.0/0.0/0.0	0.0/0.1/1.0	0.0/2.2/4.4
WebLINX	0.0/1.0/1.0	0.5/0.5/1.0	0.0/0.3/0.3	0.0/0.3/0.3	2.2/8.7/23.9	2.2/1.8/21.5	5.1/34.7/41.7	16.6/44.7/53.2	0.0/0.0/0.0	0.0/0.0/0.5	0.0/0.2/0.3	0.0/0.2/0.2
GUIAct	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.1/0.1	0.0/0.1/0.2	1.0/1.0/1.0	0.0/2.5/0.5	1.9/18.7/23.4	20.6/33.0/36.9	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.0/0.0
LanMRQA-Ref												
Mind2Web	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.3/0.6	0.0/0.7/1.2	6.2/13.3/19.5	0.5/9.0/16.0	2.6/54.5/69.3	3.3/54.6/70.1	0.0/0.0/0.5	0.0/0.0/0.0	1.0/5.0/7.7	0.7/2.2/4.5
AutoWebGLM	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.0/16.0	4.0/0.0/16.0	25.0/39.3/35.6	16.1/35.7/50.0	0.0/24.0/36.0	2.0/6.0/16.0	0.0/0.0/0.0	0.0/0.0/0.0	0.0/1.7/1.9	0.0/6.7/16.7
WebArena	0.0/0.0/0.4	0.0/0.0/0.0	0.2/0.4/4.8	0.0/0.2/4.8	13.5/52.4/48.6	6.7/23.7/23.3	0.0/34.4/49.0	9.4/40.6/51.0	0.0/0.0/0.0	0.0/0.1/2.1	0.0/0.0/1.9	0.0/0.0/4.4
WebLINX	0.0/0.0/1.0	0.0/0.0/1.0	0.3/0.5/1.1	0.0/0.0/0.3	4.3/21.7/32.6	2.5/2.4/23.7	5.0/38.9/48.9	17.8/40.5/58.3	0.0/0.0/0.0	0.0/0.0/0.0	1.7/7.0/12.0	2.0/6.0/14.0
GUIAct	0.0/0.0/0.0	0.0/0.0/0.0	0.1/0.1/0.3	0.1/0.4/0.7	21.0/46.5/56.5	9.0/24.0/32.0	0.7/18.7/23.6	23.1/51.3/38.2	0.5/0.5/0.5	0.0/0.0/0.5	0.3/0.8/1.5	0.2/0.6/0.8
CoQOwn2-v1.0												
Mind2Web	0.0/0.0/0.0	0.0/0.0/0.0	1.6/8.9/14.2	1.9/7.7/13.5	9.7/23.0/36.3	6.0/17.0/28.0	2.4/56.2/73.6	3.2/55.2/74.1	0.0/1.5/3.0	1.0/5.0/8.5	3.2/8.0/13.8	1.5/9.0/11.2
AutoWebGLM	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.2/20.8	0.0/0.0/22.0	21.4/35.7/35.6	21.4/30.4/21.4	0.0/24.0/42.0	0.0/0.0/50.0	0.0/0.0/0.0	0.0/0.6/12.0	5.4/8.9/16.1	16.7/26.7/26.7
WebArena	0.0/0.0/0.4	0.0/0.0/0.0	2.4/5.0/16.7	1.2/1.9/16.7	8.1/27.0/54.4	3.1/42.7/51.2	11.5/43.8/52.1	0.0/0.0/0.0	1.2/4.8/7.1	1.9/3.9/9.7	0.0/8.9/13.3	0.0/0.0/4.4
WebLINX	0.0/0.0/1.0	0.0/0.0/1.5	1.6/6.9/11.5	1.3/1.0/15.7	21.7/58.7/67.4	7.3/74.4/54.2	1.1/39.5/41.6	17.2/48.6/59.9	0.5/0.5/2.5	1.5/3.0/3.5	0.3/4.4/7.5	4.0/10.0/12.0
GUIAct	0.0/0.0/0.0	0.0/0.0/0.0	0.4/1.6/2.6	1.1/4.7/7.7	57.0/81.0/89.0	20.5/48.5/63.0	0.7/24.1/10.1	23.3/38.9/42.9	0.5/0.4/8.5	1.0/1.5/2.0	0.8/1.9/3.2	0.4/1.7/2.3
GME-Qwen2VL-2B												
Mind2Web	0.0/0.0/1.0	0.0/0.0/0.0	2.0/8.7/16.7	1.5/8.4/15.8	13.3/39.0/36.3	10.5/24.0/36.5	2.0/55.7/57.4	2.8/55.8/74.1	0.5/0.6/0.5	0.5/0.4/0.5	2.5/7.2/12.8	1.5/6.7/9.0
AutoWebGLM	0.0/0.0/0.0	0.0/0.0/0.0	6.0/16.0/20.0	6.0/18.0/20.0	71.4/82.1/92.9	30.4/60.7/50.0	0.0/40.0/60.0	0.0/0.0/60.0	0.0/0.0/0.0	0.0/0.6/12.0	5.4/8.9/16.1	13.3/33.5/25.0
WebArena	0.0/0.0/0.0	0.0/0.0/0.0	1.2/0.9/16.7	3.6/16.7/22.6	16.2/45.5/70.0	17.0/40.6/52.1	11.5/47.9/53.1	1.2/2.4/7.1	1.2/4.4/4.8	4.9/16.5/18.4	0.0/0.0/17.8	0.0/0.0/17.8
WebLINX	0.0/0.0/1.0	0.0/0.0/1.5	2.1/4.1/24.3	4.3/13.3/25.6	34.8/63.0/73.9	11.8/40.9/57.0	1.1/40.4/51.9	17.0/53.6/20.3	0.0/0.4/6.5	0.0/0.4/7.0	0.5/4.0/27.2	2.0/6.0/6.0
GUIAct	0.0/0.0/0.0	0.0/0.0/0.0	0.9/4.4/6.9	1.7/8.0/11.6	50.0/80.5/93.0	12.0/48.5/48.0	1.1/24.0/47.0	22.1/39.4/43.7	0.0/0.4/5.0	1.5/4.0/5.5	0.9/2.8/4.2	0.4/1.0/2.7
UniS-EMLM												
Mind2Web	0.0/0.0/0.5	0.0/0.3/1.6	0.7/3.9/7.6	5.3/9.7/13.3	1.5/6.0/11.5	5.8/57.1/73.4	4.5/55.3/74.8	2.0/4.0/5.5	0.0/0.5/1.0	1.7/6.0/11.5	0.7/7.5/9.0	0.0/0.0/0.0
AutoWebGLM	0.0/0.0/0.0	0.0/0.0/0.2	4.0/16.0/20.0	0.0/0.4/12.0	25.0/32.1/32.0	8.9/16.9/37.5	0.0/24.0/34.0	0.0/0.0/38.0	0.0/0.0/0.0	0.0/0.5/2.0	0.0/3.5/6.4	0.0/3.3/6.7
WebArena	0.0/0.0/0.0	0.0/0.0/0.4	1.2/4.8/7.1	0.0/0.1/15.5	8.1/13.5/37.8	5.3/14.7/32.0	5.2/24.7/31.1	9.4/44.8/54.2	1.2/4.8/7.1	0.0/0.0/0.0	1.9/5.8/9.7	0.0/6.7/8.9
WebLINX	0.0/0.0/1.0	0.0/0.0/1.5	0.5/3.7/7.1	0.0/0.8/1.6	2.8/2.7/21.7	1.7/1.5/22.6	3.5/36.9/48.1	17.5/48.5/59.4	0.0/0.5/2.0	0.3/0.9/1.4	0.0/2.0/2.0	0.0/0.2/0.2
GUIAct	0.0/0.0/0.0	0.0/0.0/0.0	0.4/0.4/0.9	0.4/1.3/1.6	3.5/10.5/23.5	1.0/2.5/25.3	0.7/20.2/26.6	23.2/37.4/41.0	0.0/0.0/0.0	0.1/0.3/0.0	0.1/0.3/0.0	0.0/0.2/0.4
VLM2Vec-Qwen2VL-2B												
Mind2Web	0.0/0.0/0.0	0.0/0.5/0.5	4.5/31.8/57.8	24.8/66.0/27.1	10.5/33.0/49.0	4.2/66.4/77.9	2.0/59.6/80.2	0.5/5.5/7.5	0.5/4.5/5.5	0.5/4.5/5.5	1.5/4.5/8.0	10.4/33.6/45.5
AutoWebGLM	0.0/0.0/0.0	0.0/0.0/0.4	8.0/26.0/56.0	8.0/38.0/56.0	67.9/82.1/89.3	32.1/58.9/71.4	0.0/44.0/64.0	0.0/0.0/56.0	0.0/0.0/0.0	0.0/0.2/0.8	2.0/28.0/4.0	16.1/44.6/57.1
WebArena	0.0/0.0/0.0	0.0/0.0/0.0	4.4/31.1/26.4	4.8/20.2/27.4	24.3/59.5/57.8	14.7/40.0/57.0	3.2/06.1/47.2	4.2/43.8/53.1	0.0/0.4/8.9	0.0/0.2/4.3	5.8/23.3/47.6	13.3/24.4/28.9
WebLINX	0.0/0.0/0.5	0.0/0.0/0.5	5.6/23.7/39.7	6.4/23.2/42.1	37.0/56.5/57.8	14.0/39.9/51.6	1.3/43.0/45.5	16.9/51.6/56.4	1.5/5.5/8.5	0.5/2.5/6.5	4.9/20.1/30.4	2.0/4.0/6.0
GUIAct	0.0/0.0/0.0	0.0/0.0/0.0	0.9/5.2/8.0	3.9/14.0/19.3	67.0/91.0/95.0	15.0/35.6/35.8	0.3/28.0/35.8	24.0/42.3/47.2	7.5/16.0/23.5	0.0/1.5/2.0	3.6/11.1/16.0	0.4/1.2/2.1
VLM2Vec-Ground-V1-2B												
Mind2Web	0.0/0.0/0.0	0.0/0.0/0.0	7.6/46.0/87.0	4.1/40.8/74.7	66.4/49.0/39.5	28.0/60.0/76.5	5.1/59.7/87.2	3.8/55.3/79.2	1.5/6.0/10.0	1.5/4.5/8.0	11.8/37.4/55.6	17.9/43.3/61.2
AutoWebGLM	0.0/0.0/0.2	0.0/0.0/10.0	6.0/44.0/80.0	4.0/32.0/72.0	85.7/100.0/100.0	48.2/73.2/83.9	0.0/58.0/76.0	0.0/0.0/70.0	0.0/0.0/16.0	2.0/2.0/4.0	16.1/62.5/85.7	16.7/40.0/66.7
WebArena	0.0/0.0/0.0	0.0/0.0/0.0	3.6/32.1/26.2	1.2/2.7/44.0	45.9/78.4/86.5	32.0/61.7/50.8	4.2/44.0/57.0	10.0/45.0/59.4	0.0/0.4/8.9	0.0/0.1/24.8	7.8/31.1/47.4	8.9/35.6/51.1
WebLINX	0.0/0.0/0.5	0.0/0.0/0.5	8.5/44.8/86.4	6.4/37.3/62.4	78.3/91.3/39.5	3.0/45.5/49.2	18.8/53.6/56.7	0.0/0.0/10.5	0.0/0.0/6.5	14.3/43.5/61.1	0.0/0.0/14.0	0.0/0.0/14.0
GUIAct	0.0/0.0/0.0	0.0/0.0/0.0	5.0/21.9/31.6	9.4/34.0/45.1	96.0/100.0/100.0	1.6/33.2/24.1	23.5/46.2/25.2	4.5/11.5/15.0	0.0/0.0/0.0	0.0/0.2/5.4	19.9/41.5/58.0	1.6/6.0/10.3
ShowUI-2B												
Mind2Web	0.0/0.0/0.0	0.0/0.0/0.4	1.0/0.4/6.0	0.0/0.4/6.0	1.8/2.7/5.3	1.0/3.0/11.5	4.2/52.9/69.9	4.9/5.1/36.9	0.0/0.0/5.0	0.0/0.0/0.0	0.1/0.1/0.5	0.0/0.0/0.0
AutoWebGLM	0.0/0.0/0.0	2.0/4.0/4.0	0.0/0.0/2.0	0.0/0.0/17.9	0.0/0.0/23.2	0.0/0.2/16.0	0.0/0.6/22.0	0.0/0.0/22.2	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.0/0.0	0.0/0.0/0.0
WebArena	0.0/0.0/											

Table 10: Recall@1/5/10 for all methods on each subtask in the out-of-domain setting.

Source	$(q, \tau_{1:n}) \rightarrow \tau_{i+1:n}$	$(q, \tau_{i+1:n}) \rightarrow \tau_{1:i}$	$(q, \tau_{1:n}) \rightarrow s_{i+1}$	$(q, \tau_{i+1:n}) \rightarrow s_i$	$q \rightarrow \tau_{\leq i}$	$q \rightarrow \tau_{\sim i}$	$(q, s_i) \rightarrow s_{i+1}$	$(q, s_{i+1}) \rightarrow s_i$	$(q, s_i) \rightarrow \tau_{1+1:n}$	$(q, s_{i+1}) \rightarrow \tau_{1:i}$	$q \rightarrow s_i$	$q \rightarrow s_n$
Qwen2-VL-2B												
Mind2Web	0.0/0/3/1.0	0.2/1/6/2.8	0.2/0/7/1.2	0.2/0/20/3	0.8/0/0/6.3	0.8/4/6/7.8	2.3/62.5/70.7	2.4/0/8/71.2	0.2/0/3/0.7	0.0/0/0/0.3	0.1/0/7/1.6	0.0/0/0/0.0
AutoWebGLM	0.0/0/0/0.0	0.0/0/0/0.0	0.0/7/5.7	0.0/0/0/12.5	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/22.2	0.0/0/0/35.5	2.0/0/0/0.0	0.0/0/0/0.5	1.2/0/0/2.5	5.0/0/0/0.0
WebArena	0.0/1/7/3.4	0.8/1/7/5.2	0.0/0/0/0.0	5.0/0/0/15.0	5.0/0/0/15.0	1.7/0/0/72.4	3.4/3/8/70.7	0.0/1/71.7	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0.0
WebLNX	0.0/0/0/0.9	0.9/0/9/0.9	0.0/0/0/0.0	5.0/0/0/0.0	6.0/1/0/0.24.0	0.9/0/73/50.4	10/4/0/6/53.0	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0.0
GUIAct	0.0/0/0/0.0	0.0/0/0/0.1	0.0/0/1/0.1	0.0/0/1/0.2	0.4/3/5/5.6	0.5/2/4/4.4	5.0/2/7/34.5	23/7/40/44.6	0.1/0/1.0/1	0.0/0/0/0.0	0.0/0/0/0.1	0.0/0/0/0.0
Qwen2-VL-3B												
Mind2Web	0.0/0/3/0.3	0.0/0/3/0.4	0.0/0/2/0.2	0.0/0/2/0.2	0.0/0/0/0.0	2.3/0/1/4.7	5.0/9/4/28.7	0.0/0/2/0.2	0.0/0/0/0.2	0.0/0/0/0.1	0.0/0/0/0.0	0.0/0/0/0.0
AutoWebGLM	0.0/0/0/0.0	0.0/0/0/0.0	2.5/2/3/2.2	0.0/0/3/2.2	5.0/0/3/0.5	4.0/1/0/21.0	0.0/0/0/5.0	0.0/0/0/5.0	0.0/0/0/2.5	0.0/0/0/0.0	1.7/1/0/2.1	0.0/0/0/0.0
WebArena	0.0/1/7/1.7	0.0/0/0/0.0	0.0/0/0/0.6	0.0/0/0/6.9	0.0/0/0/15.0	0.0/6/0/0.0	3.4/1/2/11.8	1.7/1/3/82.0	0.0/0/0.0	0.0/0/0.0	0.0/0/0/4.1	0.0/0/0/0.0
WebLNX	0.0/0/0/0.9	0.0/0/0/0.9	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0.0	0.0/3/0/8.0	5.2/3/8/34.5	12/2/4/5/47.8	0.0/0/0.0	0.0/0/0.0	0.0/0/0/8.8	0.0/0/0/0.5
GUIAct	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0.2	0.1/0/10.2	0.0/1/4/4.2	0.7/1/63.3	2.3/29/0.36	21/1/41/0.47.7	0.1/0/5.0/7	0.0/0/0.0	0.1/0/20.2	0.0/0/0.0
Lamra-Ret												
Mind2Web	0.0/0/3/0.9	0.0/0/3/1.0	0.2/1/0/2.1	0.3/0/7/1.7	4.8/0/4/27.0	2.1/0/0/15.4	2.1/4/6/40.8	3.1/2/5/86.5	0.0/0/3/0.7	0.0/0/0/0.0	1.2/5/3/8.6	2/4/5/6/7.1
AutoWebGLM	0.0/0/0/0.0	0.0/0/0/0.0	5.0/0/0/5.0	5.0/0/0/0.0	30/0/0/0.7	19/0/0/36.0	0.0/0/2/27.5	10/0/0/52.2	0.0/2/5/5	0.0/0/0/0.0	1.7/8/2/2.3	5/0/0/0.0
WebArena	0.0/0/0/3.4	0.0/3/4/5.2	0.0/0/0/0.0	0.0/0/0/0.0	10/0/0/35/0.5	6.0/0/18/27.0	0.0/19/0/27.6	5.2/2/7/63.2	0.0/0/0.0	0.0/0/0.0	2.7/6/8/8.2	5/0/0/0.5
WebLNX	0.0/0/0/0.0	0.0/0/0/0.0	0.0/2/6/4.3	15.0/0/50/0.5	4.0/2/0/0.48.0	1.7/4/8/75.9	13/0/5/13/5.6	0.0/0/0.0	0.9/0/9.9	5/1/15/3/9.5	0/0/0/0.15.0	0/0/0/0.0
GUIAct	0.0/0/0/0.0	0.0/0/0/0.0	0.1/0/20.2	0.3/0/71.2	27.8/1/86.0	22/12/30/43.5	0.7/26/0/33.4	23/3/40/3/44.7	0.2/0/71.8	0.0/0/0.0	0.6/1/83.2	0/4/1/8/1.8
ColQwen-v1.0												
Mind2Web	0.0/1/6/3.1	0.0/1/6/4.0	4.0/1/4/20.2	1.0/8/87/11.0	20/6/28/0.49.2	13/0/2/0/36.8	1.2/6/3/27.7	2/4/6/6/37.7	1.9/6/6/11.2	2/8/11/0.18.8	5/4/18/0/25.3	11/1/21/4/27.0
AutoWebGLM	0.0/0/0/0.0	0.0/0/0/0.0	5.0/3/0/45.5	10/0/0/5/0.5	0.0/0/0/0.5	0.0/0/0/0.5	0.0/0/0/5/0.5	0.0/0/0/6/0.0	0.0/0/0/0.0	1.3/2/5/0/3.2	5/0/1/0/3/5.0	0/0/0/0.0
WebArena	0.0/0/0/3.4	0.0/3/4/3.4	0.0/0/0/0.0	0.0/0/0/0.0	10/0/0/35/0.5	11/0/2/0/45.0	1.7/20/75.9	3.4/3/1/0.43.1	0.0/1/75.2	0.0/0/0.0	3.4/6/9/10.3	0/0/0/0.15.0
WebLNX	0.0/0/0/0.9	0.0/0/0/9.0	3.5/1/2/21.7	1.7/6/16.1	20/0/5/0/0.70.0	16/0/4/0/1.1	2.6/4/8/75.2	10/4/5/48/3.8	0.0/0/9.0/9	0.9/7/89.6	4/2/11/0/1.2	5/0/1/5/0.5
GUIAct	0.1/0/3/1.6	0.0/0/4/2.2	1.0/5/14.1	2.7/8/21/12	71/1/85/69.0	28/9/5/6/16.7	0.5/4/1/42.0	23/5/4/6/53.2	1.6/6/5/12.2	0.6/2/4/3.8	0.6/2/9/4.7	2/1/7/0/10.2
GME-Qwen2VL-2B												
Mind2Web	0.0/1/7/6.5	0.0/1/7/6.5	4.2/1/8/21.3	30/2/5/7/1.7	59/5/2/1/47.6	1.2/5/3/27.5	2.3/6/6/5/7.8	3.0/13/4/27.9	3.3/13/1/26.5	5.5/15/4/21.1	4/0/14/3/18.3	5/0/0/0/0.0
AutoWebGLM	0.0/0/0/0.0	0.0/0/0/0.0	5.0/3/7/55.0	0.0/2/0/54/0.0	10/0/0/0/5/0.0	0.0/0/0/0.5	0.0/0/0/5/0.5	0.0/0/0/6/0.0	0.0/0/0/0.0	1.7/8/2/2.3	2/7/8/2/1.3	0/0/0/0.0
WebArena	0.0/3/4/5.2	0.0/9/8/6.9	0.0/3/4/8.6	0.0/6/9/10.3	25/0/7/0/7.0	15/0/4/4/0.65.0	1.7/20/72.9	5.2/2/7/34/1.4	3.4/6/9/20.7	0.0/4/1/5.5	0/0/0/0.20.0	0/0/0/0.0
WebLNX	0.0/1/7/1.7	0.0/2/6/5.2	7.8/2/8/63.4	5.2/1/6/20.0	60/0/6/0.95/0.0	29/0/5/0/7.0	2.6/5/2/55.7	11/3/5/39/59.1	0.0/6/19.6	0.9/8/71.1	4.2/7/1/2.7	10/0/20/20.0
GUIAct	0.0/0/0/0.0	0.0/0/0/0.0	2.2/8/6/11.9	3.6/1/18/16.1	66/2/8/3/79.3	22/5/2/6/26.4	1.1/5/6/44.2	23/3/4/7/59.3	1.0/2/6/4.2	2.6/9/7/15.4	0/4/3/10.6	0/0/0/0.0
UniSE-MLLM												
Mind2Web	0.0/1/9/4.7	0.3/2/1/5.1	3.3/8/7/12.2	1.9/6/3/10.1	4.8/8/7/16.7	4.1/9/4/21.1	6.3/6/5/4/7.5	2/4/6/3/0/7.3	3.0/13/0/15.9	0/9/4/28.2	4/7/12/2/1.4	0/3/2/7.9
AutoWebGLM	0.0/0/0/0.0	0.0/0/0/0.0	2.5/1/0/15.0	2.5/2/0/22.5	0.5/0/2/5.0	5.0/0/2/5.0	0.0/0/0/2/5.0	0.0/0/0/5/0.5	5/0/15/0/15.0	0/0/6/11.7	5/0/0/10/10.0	0/0/0/0.0
WebArena	1.7/3/4/6.9	0.0/6/9/6.9	1.0/0/0/0.1	1.7/1/73.4	0.0/1/0/5/2.0	8.6/3/1/0.5	3.4/3/6/21.4	0.0/5/28.6	0.0/0/0/1.7	2.7/2/72.7	0.0/0/0.0	0/0/0/0.0
WebLNX	0.0/0/0/0.9	0.0/0/0/9.4	0.9/1/7/2.6	15.0/2/0/6/5.0	7.0/2/3/0/36.0	3.5/0/4/9/1.5	11/3/5/13/5.7	1.7/5/27.0	0.0/0/9/1.7	0.0/0/0/0.0	0/0/0/0.0	0/0/0/0.0
GUIAct	0.0/0/0/0.1	0.0/0/0/0.1	0.3/0/8/1.2	0.5/1/5/2.5	3.9/1/0/21.4	1.8/8/8/6.6	1.6/3/6/38.3	23/5/4/2/64.8	0/1/0/8/1	0.0/0/0/0.0	0/1/0/30.4	0/0/0/0.40.7
VLM2Vec-Qwen2VL-2B												
Mind2Web	0.0/2/3/7.0	0.0/2/1/1.1	6.9/4/2/6/2.5	6.1/4/8/6/6.5	38/9/69/55.1	21/9/4/2/5.7	1.2/5/3/6/4.1	1.6/7/5/6/8.7	3.1/2/7/3/8.3	2/4/6/3/8/6.4	2/3/16/2/3/7.3	23/0/0/1/6.5
AutoWebGLM	0.0/2/0/0.5	0.0/2/5/7.5	7.5/6/0/0.72.5	7.5/7/2/5.7	50/0/0/0.9/0.5	50/0/1/0/7.5	0.0/8/5/0.9/5	0.0/8/5/0.9/5	7/5/4/0/5/7.0	5/0/30/0/5/7.5	36/7/7/0/0.8/1.7	20/0/0/0/0.0/0.0
WebArena	0.0/5/2/5.2	0.0/5/2/8.6	6.9/1/3/8/1.7	1.7/1/9/20.7	35/0/7/0/0.8/0.0	13/0/4/3/0.6/5.0	3/4/2/6/3/1.0	1.7/1/3/54.8	1.7/1/7/22.7	0.0/5/21/3.8	12/3/27/4/3/7.0	5/0/15/0/15.0
WebLNX	0.0/0/9/6.1	0.0/0/0/5.5	7.8/2/8/74.2	7.0/3/0/4/4/1.7	70/0/0/10/0.0	29/0/6/2/0.8/3.4	4/3/0/6/7.8	0.0/5/2/6/5.5	0.0/8/71.1	15/3/6/4/4/3.2	0/0/0/0/0.0/0.0	0/0/0/0.0
GUIAct	0.0/0/0/0.3	0.0/0/0/1.9	8/7/3/5/64.4	9/9/39/349.1	96/59/99/399.6	15/8/9/7/54.7	0.6/3/9/24/7.7	23/6/5/2/86/0.7	6/4/2/0/4/4.1	2/8/2/58/3/68.4	10/6/38/0/41.9	0/0/0/0.0
VLM2Vec-v1.2												
Mind2Web	0.0/1/7/10.8	0/0/3/5/13.4	1.0/1/57/67.7	2.8/5/8/67.9/2	79/4/9/0/95.5	21/9/4/2/5.7	1.2/5/3/85.3	2/3/7/3/86.4	2.3/7/3/86.4	2/7/6/7/17/9.6	37/3/6/2/83.3	0/0/0/0.0
AutoWebGLM	0.0/2/0/0.6/0.0	0/0/22/5/5.0	0/0/85/9.2/5	0/0/0/0/9.5/0.0	75/0/0/5/0.5/0.0	46/0/6/7/7/0.0	0/0/8/5/0.7/5	0/0/9/2/5/0.5/0	2/5/4/7/5/7.5	0/0/32/5/6/7.5	56/7/8/6/79.1/7	20/0/0/0/0.0/0.0
WebArena	0.0/8/6/10.3	0/0/10/3/12.1	1.7/1/38/24.1	3/4/2/3/44.8	55/0/5/7/0/5.0/0.0	31/0/6/0/75.0	1.0/3/17/9/4/8.8	6/9/3/4/54/4.8	3/14/17/2/5/9.5	0/0/5/21/9.0	20/5/42/5/45.8	25/0/45/0/55.0
WebLNX	0.0/0/9/6.1	0/0/0/0/5.5	7/0/6/5/27.8/3	3/5/0/4/6/7.0	0/0/0/10/0.0/0.0	2/0/0/7/4/8/9/9	6/2/0/7/4/8/9/6	12/2/6/5/2/72.4	4/3/10/4/2/0.0	2/6/9/6/1/8.3	36/4/7/7/18/9.0	30/0/0/0/0.7/0.0
GUIAct	0.0/0/1/1.4	0/0/0/0/1.9	8/7/3/5/64.4	9/9/39/349.1	96/59/99/399.6	39/7/7/2/86/0.7	6/4/2/0/4/4.1	2/8/2/58/3/68.4	6/4/2/0/4/4.1	2/8/2/58/3/68.4	10/6/38/0/41.9	0/0/0/0.0
UGround-VI-2B												
Mind2Web	0.0/2/0/7.0	0/0/2/0.2	0.0/0/2/0.2	0.0/0/0/0.7	0.0/0/1/6/4.8	0.0/1/6/5/4/7.4	0.0/1/5/6/2.5	2/1/5/0/1/5.7	0.0/0/2/0.5	0.0/0/0/0.0	0/3/0/6/0.7	0/0/0/0.0
AutoWebGLM	0.0/0/0/0.0	0/0/2/5/0.0	2/5/2/5/5.0	5/0/2/0/5/2.5	5/0/2/0/5/2.5	0/0/1/2/5/0.0	0/0/1/5/2/0.0	0/0/1/5/2/0.0	0/0/0/0/2/5	0/0/0/0/0.0	0/0/1/7/5.0	0/0/0/0.5/0.0
WebArena	0.0/0/0/0.7	0/0/1/7.4	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0/0.0	0.0/0/0/0/0.0	3/4/1/2/4/2.1	5/2/2/5/2/0.0	0/0/0/0/0.0	0/0/0/0/0.0	0/0/0/0/0.0	0/0/0/0/0.0
WebLNX	0.0/0/0/0.0	0/0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0.0	0.0/0/0/0/0.0	0.0/0/0/0/0.0	0.0/0/0/0/0.0	0/0/0/0/0.0	0/0/0/0/0.			

1188 similarity metric, and top- k . Any model-specific pre/post-processing strictly follows the official
 1189 recommendations in the cited papers to ensure fairness.
 1190

1191 **D.3 DETAILED ANALYSIS OF MULTIMODAL RETRIEVAL BASELINES**
 1192

1193 Among retrieval-centric baselines, **VLM2Vec-v2.2** provides both the highest scores and the lowest
 1194 variance across benchmarks. Its consistent improvements over **VLM2Vec-Qwen2VL-2B** are plau-
 1195 sibly attributed to the integration of visual document and video corpora during pretraining, which
 1196 expands the distributional coverage beyond image-text pairs. This aligns with the broader pattern that
 1197 the **VLM2Vec** family outperforms other approaches: training on fused-modality data (text, images,
 1198 documents, and videos) with interleaved batching appears to strengthen representation alignment and
 1199 negative sampling diversity, both crucial for trajectory-oriented retrieval.

1200 By contrast, **LamRA-Ret**, trained primarily on text-image datasets, lags behind models exposed
 1201 to structured visual content such as **ColQwen2-v1.0** and **GME-Qwen2VL-2B**. While LamRA-Ret
 1202 still improves over backbone-only baselines, its gap indicates that text-image supervision alone
 1203 is insufficient when queries/targets involve layout cues, dense text regions, or multi-panel context
 1204 common in GUI trajectories. Between the document-pretrained lines, **GME-Qwen2VL-2B** surpasses
 1205 **ColQwen2-v1.0**, likely benefiting from a broader mixture of visual and textual inputs beyond pure
 1206 documents, which improves cross-domain generalization. Finally, **UniSE-MLLM**, despite screenshot-
 1207 centric finetuning, records the weakest outcomes and occasionally underperforms backbone baselines,
 1208 suggesting that narrow, screenshot-only specialization can impede flexibility for open-domain retrieval
 1209 where both semantic breadth and structural variance matter.

1210
 1211 **D.4 PER-TASK EVALUATION**

1212 Figure 8 presents Recall@5 results broken down by retrieval task across multiple datasets for five
 1213 representative models: ColQwen2-v1.0, GME-Qwen2VL-2B, VLM2Vec-Qwen2VL-2B, VLM2Vec-
 1214 V2.2, and GAE-Retriever, under both in-domain and out-of-domain settings. As depicted, all models
 1215 achieve their best results on the $q \rightarrow \tau$ and $q \rightarrow s$ tasks, indicating that semantic retrieval tasks are
 1216 relatively simple. On the contrary, performance declines significantly on $(q, \tau) \rightarrow \tau'$ and $(q, s) \rightarrow \tau$,
 1217 which can be attributed to the absence of similar supervision in existing benchmarks.

1218 GAE-Retriever attains the highest scores on nearly every task over all datasets, especially evident
 1219 on GUIAct and AutoWebGLM. Notably, it performs even better in out-of-domain scenarios than in-
 1220 domain ones for tasks like $q \rightarrow s$, $(q, \tau) \rightarrow \tau'$, and $(q, s) \rightarrow \tau$, showcasing the robust generalization
 1221 capability of our proposed framework.

1222 Among baseline methods, VLM2Vec-V2.2 stands out as the most competitive, showing solid results
 1223 across most tasks. It even surpasses GAE-Retriever on $q \rightarrow \tau$ in the out-of-domain WebArena, which
 1224 can be attributed to its enhanced transfer capabilities from video pretraining. However, it consistently
 1225 falls short of GAE-Retriever on more complex tasks, highlighting the necessity of retrieval models
 1226 optimized for the multimodal trajectory retrieval task.

1227 VLM2Vec-Qwen2VL-2B and GME-Qwen2VL-2B achieve moderate results but remain behind the
 1228 top two models. It can be observed that they outperform the document-only pretrained ColQwen2-
 1229 v1.0, proving the importance of training on diverse domains. The relatively poor performance of
 1230 ColQwen2-v1.0 suggests its limited adaptability beyond documents. Meanwhile, GME-Qwen2VL-
 1231 2B's gap with its VLM2Vec counterpart points to deficiencies in training.

1232 Overall, the consistent advantage of GAE-Retriever reinforces the value of trajectory-based modeling
 1233 and retrieval training for building versatile multimodal retrieval systems.

1234
 1235 **D.5 COMPARISON OF DIFFERENT RETRIEVAL METHODS.**

1236 To substantiate that our retrieval strategy surpasses the conventional text-only trajectory retrieval
 1237 $q \rightarrow \tau$, we conduct a controlled comparison on 50 tasks from Online-Mind2Web (13 Easy, 26 Medium,
 1238 11 Hard). The state-conditioned multimodal formulation $(q, s) \rightarrow s'$ encodes the current page state
 1239 (DOM/screenshot) together with the task instruction and directly retrieves an executable next step
 1240 under visually similar states, achieving visual alignment and localized guidance. In contrast, $q \rightarrow \tau$

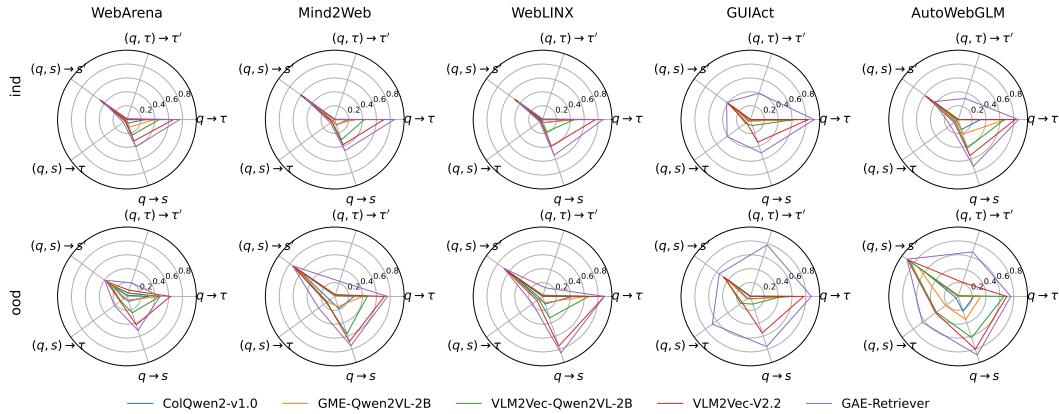


Figure 8: Per-Task Evaluation Results. Recall@5 is reported for five selected models across various retrieval tasks and datasets under in-domain (ind) and out-of-domain (ood) scenarios.

relies solely on textual semantic matching, which often introduces redundant exemplars misaligned with the current layout, leading to a mismatch between guidance and decision needs.

From the results in Figure 11, we observe that the multimodal approach consistently outperforms the text-only baseline across all difficulty levels: on *Easy* tasks, success increases from 69.23% to 76.92%, and on *Medium/Hard* tasks the gains are substantial (approximately +30.77 and +18.18 percentage points, respectively). These results corroborate the superiority of the proposed multimodal, state-conditioned retrieval.

Table 11: Comparison of Different Retrieval Methods.

Retrieval Strategy	Success Rate (%)				#Average	
	Easy	Medium	Hard	All	Step	Reward
$q \rightarrow \tau$	69.23	23.07	18.18	34	33.36	7.58
$(q, s) \rightarrow s'$	76.92	53.84	36.36	56	24.21	8.06

D.6 EVALUATING TRAJECTORY RETRIEVAL IN ADVANCED AGENTS

To validate the effectiveness of trajectory retrieval across different SOTA agent frameworks, we integrate GAE-Retriever as a plug-and-play module into SeeAct (Zheng et al., 2024) under the same experimental setup. As shown in Table 12, even without extensive prompt tuning, introducing trajectory knowledge consistently improves task performance, yielding an 8–10% gain in success rate. The improvement is particularly pronounced on medium and hard tasks. Comparing text-based and vision-based retrieval, we observe that visual knowledge performs better on Medium/Hard settings, indicating that visual exemplars are more helpful for fine-grained operations in complex interactive scenarios. These results demonstrate that our method is highly extensible across different agent frameworks.

Table 12: Performance of SeeAct with Trajectory Retrieval on Tasks of Different Levels.

Method	Easy (%)	Medium (%)	Hard (%)	Success Rate (%)
SeeAct	71.42	47.36	6.25	49.00
$+ \mathcal{D}_{\text{text}}$	78.57	55.26	18.75	57.00
$+ \mathcal{D}_{\text{vision}}$	76.19	60.52	25.00	59.00

D.7 INFERENCE-TIME EFFICIENCY OF WEBRAGENT WITH GAE-RETRIEVER

In WebRAGent, each planning step introduces an additional round of GAE Retriever encoding, vector retrieval, and LLM reranking. Compared with the baseline, this does introduce some extra computation, but the overhead is very small relative to the entire task execution: it results in roughly

1296 10% additional latency per step, which we consider fully acceptable in practice. We have applied
 1297 several engineering optimizations to the retrieval and LLM reranking stages, including a two level
 1298 cache on the LLM side and optimized and compressed image encoding. Our efficiency experiments
 1299 are mainly conducted on an NVIDIA RTX A6000 GPU. Table 13 shows that the dominant time cost
 1300 does not come from the GAE Retriever retrieval stage, but from the LLM reranking, while the overall
 1301 time per step remains modest. When the cache is hit, most queries require almost no extra reranking
 1302 time and the corresponding inference cost is nearly negligible.

Table 13: Inference-time efficiency of GAE-Retriever and LLM reranking.

Setting	Retrieval (s/step)	Rerank (s/step)	Total (s/step)	Speedup
No Cache	0.3877	3.5374	3.9251	1.0×
With Cache	0.3828	0.0002	0.3830	9.24×

1310 D.8 DISCUSSION ON THE RISK OF MISSING SMALL, LOW-CONTRAST UI ELEMENTS

1311 In our framework, the UI graph token selection module is only applied during training, with the
 1312 primary goal of saving GPU memory and enabling a larger batch size. During inference, we always
 1313 feed the full-resolution patch grid produced by the standard Qwen-VL preprocessing pipeline.

1314 To address the potential risk of losing small yet critical UI elements, we only merge patches that are
 1315 spatially adjacent and whose color difference is below a certain threshold (i.e., $\|\mathbf{c}_i - \mathbf{c}_j\|_2 \leq 1$ in
 1316 RGB space). In typical cases, common buttons or checkboxes exhibit a noticeable contrast against
 1317 the background (as shown in the left part of Figure 9). If a checkbox is so similar to the background
 1318 that it is indistinguishable even to the human eye (as in the right part of Figure 9), then, in such
 1319 an extreme case, the agent would also be unable to distinguish it. We thus treat this checkbox as
 1320 effectively nonexistent, which is reasonable from the agent’s perspective. Moreover, even in extreme
 1321 cases where a very small number of UI elements might be merged away, this can only happen during
 1322 training; at inference time, we still feed the full patch grid as input.

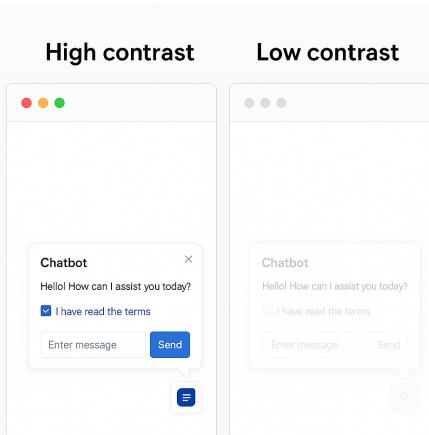


Figure 9: Comparison of High-Contrast vs. Low-Contrast UI Checkboxes

1340 E PROMPT

1341 E.1 ANNOTATION PROMPTS AND ACTION TEMPLATES

1342 This section demonstrates the prompts used in the paper. Table 14 includes prompts for data annotation
 1343 such as state description generation, HTML rendering, and silver data generation. Snippet 1 specifies
 1344 the action space definitions used for Mind2Web, WebLINX, WebArena, GUIAct, and AutoWebGLM.
 1345 Snippets 2 and 3 show the instruction templates used to augment retrieval queries, where *description*
 1346 refers to the natural language summary of a trajectory.

Table 14: Data annotation prompts.

Function	Content
Description Generation	Generate a concise one-sentence description of the content and layout of the provided webpage screenshot.
HTML Rendering	Your task is to convert the simplified HTML input provided by the user into a fully renderable, standard HTML format while preserving all original information intact. Enhance the HTML with appropriate styling to make it visually appealing and resemble a typical, functional website. Return only the HTML code without any additional text. HTML: <i>html</i> . Ensure that the returned HTML code includes the ID <i>[id]</i> (mentioned in <i>context</i>) with the same element exactly as provided.
Silver Generation	<p>Step 1: You are a helpful AI assistant proficient in Named Entity Recognition (NER). Analyze the following sentence and provide the most comprehensive NER results for each noun in JSON format, using greedy matching. Labels should be specific contextual descriptions of the entity. Sentence: <i>instruction</i>.</p> <p>Step 2: You are an AI assistant skilled in generating alternatives. Given a sentence and a list of named entities, generate five alternative texts for each entity that align with its semantic label while being entirely different in meaning from the original text. Ensure the alternatives fit naturally and consistently within the sentence, maintaining the original representation (e.g., text remains text, emojis remain emojis). Sentence: <i>instruction</i>, Named Entities: <i>ners</i>.</p> <p>Step 3: You are an AI assistant specialized in rewriting user queries. Your task is to refine the following five queries to ensure they are consistent, natural, concise, logical, and human-like. Rewrite each query by varying the wording, structure, and style to ensure diversity in expression. Your response should align with real-world common sense and factual accuracy.</p>

SNIPPET 1: ACTION SPACE	
1367	Mind2Web
1368	1. click: Simulates a mouse click on the target element (bounding box).
1369	2. type: Types the specified value (str) into the target text input element (bounding box).
1370	3. select: Selects the specified value (str) from a target dropdown element (bounding box).
1371	WebLINX
1372	1. click: Simulates a mouse click on the target element (bounding box).
1373	2. hover: Simulates hovering over the target element (bounding box).
1374	3. textInput: Types the value (str) into the target element (bounding box).
1375	4. change: Changes the value of the target element (bounding box) to the specified value (str).
1376	5. load: Loads the webpage at the specified url value (str).
1377	6. submit: Submits the form identified by the target element (bounding box).
1378	7. scroll: Scrolls the page to the specified coordinate values in the list of floats [x, y].
1379	8. copy: Copies the specified text value (str) from the target element (bounding box).
1380	9. paste: Pastes the specified text value (str) into the target element (bounding box).
1381	WebArena
1382	1. click: Simulates a mouse click on the target element (bounding box).
1383	2. press: Simulates the pressing of a key combination value (str) on the target element (bounding box).
1384	3. selectOption: Selects the specified option value (str) from the target dropdown element (bounding box).
1385	4. check: Checks the target checkbox element (bounding box).
1386	GUIAct
1387	1. click: Clicks on the target element (bounding box).
1388	2. hover: Hovers over the target element (bounding box).
1389	3. input: Inputs the given text value (str) into the target element (bounding box).
1390	4. scroll: Scrolls the screen by the values in the list of coordinate floats [down, right], where down represents vertical scroll and right represents horizontal scroll.
1391	5. select_text: Selects text by dragging across the specified coordinate values in the list of floats [x1, y1, x2, y2], where (x1, y1) is the starting point and (x2, y2) is the ending point.
1392	6. copy: Copies the specified text value (str) to the clipboard.
1393	7. enter: Simulates pressing the Enter key.
1394	8. select: Selects the text value (str) in the target element (bounding box).
1395	9. answer: Provides an answer or response specified by text value (str) to the user.
1396	AutoWebGLM
1397	1. click: Clicks on the target element (bounding box).
1398	2. hover: Hovers over the target element (bounding box).
1399	3. select: Selects the option value (str) from a dropdown target element (bounding box).
1400	4. type_string: Types the specified content (str) into the target element (bounding box) and presses Enter if press_enter (bool) is True. The action value is a list [content, press_enter].
1401	5. scroll_page: Scrolls the page in the specified direction value ('up' or 'down').
1402	6. go: Navigates browser history in the specified direction value ('forward' or 'backward').
1403	7. jump_to: Opens the specified url (str) and optionally in a new tab if new_tab (bool) is True. The action value is a list [url, new_tab].
1404	8. switch_tab: Switches to a browser tab specified by the value tab_index (int).
1405	9. user_input: Displays the specified message (str) to obtain user input.
1406	10. finish: Completes the task with an optional answer value (str or None).

1404

SNIPPET 2: INSTRUCTION TEMPLATE

1405

1406

 $(q, \tau_{1:i}) \rightarrow \tau_{i+1:n}$

1407

Determine the next web navigation trajectory using the task instruction "*description*" and the prior trajectory below.

1408

Retrieve the upcoming web navigation trajectory as specified by the task "*description*" and the previous trajectory provided.

1409

Search the next phase of the web navigation trajectory based on the user query "*description*" and the earlier trajectory.

1410

From the user input "*description*" and the past navigation steps, locate the subsequent navigation sequence for GUI agents.

1411

Apply the request "*description*" to the previous web navigation steps to derive the next trajectory.

1412

Represent the previous trajectory for web agents below to determine the next trajectory based on the task "*description*".

1413

According to the previous web agent trajectory below, identify the next sequence of steps to complete the user instruction "*description*".

1414

Based on the previous trajectory below, search the next interaction sequence of web agents for the user request "*description*".

1415

With the previous GUI navigation trajectory below as a guide, look for the next trajectory for the user intention "*description*".

1416

Given the goal "*description*" and the earlier navigation sequence from GUI agents, match the subsequent trajectory.

1417

 $(q, \tau_{i+1:n}) \rightarrow \tau_{1:i}$

1418

Find the previous web browsing trajectory based on the user input "*description*" and the current trajectory.

1419

Identify the former web navigation history by analyzing the user request "*description*" and the provided trajectory.

1420

With the instruction "*description*" and the following interaction sequence, extract the earlier trajectory for web agents.

1421

Determine the past trajectory using the goal "*description*" and the succeeding navigation sequence for GUI agents.

1422

Retrieve the preceding GUI navigation trajectory with the task description "*description*" and the succeeding trajectory below.

1423

Using the user query "*description*" and the succeeding navigation steps for web agents, locate the preceding interaction steps.

1424

Analyze the query "*description*" along with the provided web agent trajectory to derive the former navigation steps.

1425

Based on the request "*description*" and the later web interaction trajectory, look for the prior navigation sequence.

1426

Consider the task "*description*" together with the given trajectory to retrieve the prior web navigation trajectory.

1427

Represent the current trajectory for web agents below to determine the previous trajectory according to the task "*description*".

1428

 $(q, \tau_{1:i}) \rightarrow s_{i+1}$

1429

Identify the upcoming state from the earlier web navigation trajectory below and the instruction "*description*".

1430

Extract the next state from the provided previous navigation sequence for web agents and the directive "*description*".

1431

Locate the following state from the given former web navigation trajectory and the task input "*description*".

1432

Determine the subsequent observation from the task "*description*" and the earlier web interaction trajectory.

1433

Find the next observation by considering the command "*description*" and the former GUI navigation trajectory.

1434

Using the user instruction "*description*" and the former web navigation trajectory, ascertain the subsequent state.

1435

Based on the former web interaction history provided and the user request "*description*", deduce the next state.

1436

Use the user intention "*description*" and the earlier navigation sequence for GUI agents to derive the following state.

1437

Find the next state based on the goal "*description*" and the earlier interaction history for web agents.

1438

Represent the given GUI navigation history to locate the upcoming state according to the user intention "*description*".

1439

 $(q, \tau_{i+1:n}) \rightarrow s_i$

1440

Find the antecedent state using the command "*description*" and the current web navigation trajectory.

1441

Identify the prior state by applying the directive "*description*" along with the present navigation trajectory for GUI agents.

1442

Locate the former observation using the instruction "*description*" and the upcoming web interaction trajectory.

1443

Ascertain the preceding observation based on the task input "*description*" and the provided web agent browsing sequence.

1444

Determine the previous state by employing the goal "*description*" along with the upcoming navigation trajectory for web agents.

1445

Find the antecedent state with the user instruction "*description*" and the succeeding GUI navigation trajectory.

1446

Retrieve the former observation given the request "*description*" and the upcoming web navigation history.

1447

Identify the prior state based on the task "*description*" and the succeeding web interaction trajectory.

1448

Represent the provided GUI navigation history to retrieve the previous state using the user query "*description*".

1449

Recognize the prior observation using the user task "*description*" and the given web navigation trajectory.

1450

 $q \rightarrow \tau_{\equiv}$

1451

Determine the complete web navigation trajectory based on the following instruction "*description*".

1452

Locate the equivalent web navigation trajectory derived from the following user input "*description*".

1453

Find the GUI navigation history aligning with the following goal "*description*".

1454

Match the corresponding trajectory for web agents using the following user query "*description*".

1455

Pinpoint the equivalent navigation trajectory for GUI agents with the following task "*description*".

1456

Ascertain the corresponding web interaction trajectory using the request "*description*".

1457

Identify the unique navigation trajectory for web agents according to the provided instruction "*description*".

1458

Determine the complete GUI navigation trajectory from the user task "*description*".

1459

Ascertain the unique GUI interaction history by considering the task "*description*".

1460

Locate the exactly equivalent web navigation trajectory based on the given query "*description*".

1461

 $q \rightarrow \tau_{\sim}$

1462

Determine the analogous web navigation trajectory based on the following directive "*description*".

1463

Identify the similar web navigation history using the task input "*description*".

1464

Locate the akin navigation sequence for GUI agents as dictated by the user input "*description*".

1465

Retrieve the similar web browsing trajectory as specified by the instruction "*description*".

1466

Identify the similar GUI interaction history based on the task description "*description*".

1467

Locate the analogous navigation trajectory for web agents using the instruction "*description*".

1468

Retrieve the analogous interaction history for GUI agents based on the provided command "*description*".

1469

Find a similar GUI navigation history following the task "*description*".

1470

Extract a similar web browsing trajectory based on the instruction "*description*".

1471

From the user query "*description*", match the similar web agent navigation trajectory.

1458 SNIPPET 3: INSTRUCTION TEMPLATE (CONT.)
1459

1460 $(q, s_i) \rightarrow s_{i+1}$
1461

Retrieve the following web navigation state according to the instruction "*description*" and the prior state.
Determine the subsequent web navigation observation given the task description "*description*" and the preceding state.
Retrieve the upcoming observation for web navigation agents following the user input "*description*" and the previous state.
Identify the next navigation state for GUI agents using the goal "*description*" and the preceding state.
Using the provided instruction "*description*" and the former state, what is the next GUI navigation state?
From the query "*description*" and the prior observation, derive the next state in the web navigation trajectory.
Given the user input "*description*" together with the current state, find the next web navigation state.
Taking the task input "*description*" and the former state, what is the subsequent GUI navigation state?
Considering the directive "*description*" and the preceding observation, determine the next web navigation state.
With the task "*description*" and the previous state from web agents as inputs, determine the subsequent state.

1468 $(q, s_{i+1}) \rightarrow s_i$
1469

Retrieve the prior web navigation state using the task "*description*" along with the current state.
In light of the instruction "*description*" and the current state provided, deduce the prior GUI navigation state.
Based on the provided user input "*description*" and the current observation, find the prior navigation state for web agents.
With the directive "*description*" and the current state, determine the prior web browsing state.
Considering both the current web agent observation provided and the user intention "*description*", locate the prior navigation state.
Given the present state and the goal "*description*", determine the previous GUI navigation state.
Combine the task description "*description*" with the current state to identify the preceding navigation state for GUI agents.
Taking the description "*description*" and the current state into account, search the previous web agent state.
Utilize the user request "*description*" alongside the present state to extract the prior GUI navigation state.
Use the directive "*description*" with the current state to look for the state directly preceding in the web agent navigation trajectory.

1478 $(q, s_i) \rightarrow \tau_{i+1:n}$
1479

Find the subsequent web navigation trajectory based on the instruction "*description*" and the previous state.
Based on the task "*description*" and the previous observation, identify the subsequent GUI navigation trajectory.
Locate the next GUI navigation trajectory by applying the instruction "*description*" to the previous state.
With the user input "*description*" and the previous state in hand, identify the next navigation trajectory for web agents.
What is the following navigation trajectory for GUI agents when applying the user intention "*description*" to the previous state?
Given the user goal "*description*" and the previous state, search the next web navigation trajectory.
When given the user instruction "*description*" and the former state, what is the next trajectory for web navigation?
Identify the next web navigation trajectory by merging the task "*description*" with the previous state.
From the directive "*description*" and the prior state, look for the subsequent GUI navigation trajectory.
Determine the subsequent browsing trajectory for web agents with the task "*description*" and the previous state as references.

1487 $(q, s_{i+1}) \rightarrow \tau_{1:i}$
1488

Find the previous web navigation history based on the instruction "*description*" and the current state.
Retrieve the preceding web navigation trajectory using the intention "*description*" along with the present state.
From the instruction "*description*" and the present state, find the prior GUI navigation history.
What does the previous navigation history for web agents look like when derived from the user input "*description*" and the current state?
Locate the prior GUI navigation history by combining the description "*description*" with the current observation.
Identify the web navigation trajectory preceding the current state according to the task "*description*".
Derive the previous navigation trajectory for GUI agents by combining the instruction "*description*" with the current state.
Search the browsing history for web agents that came before the provided current observation with regard to the user query "*description*".
Based on the current state and the user intention "*description*", extract the trajectory that came before in the web navigation.
Recognize the GUI navigation history that predates the current state by considering the user need "*description*".

1496 $q \rightarrow s_i$
1497

Find the specific web navigation state corresponding to the description "*description*".
Identify the equivalent web navigation state as defined by the description "*description*".
Extract the GUI navigation observation that corresponds with "*description*".
Locate the web navigation state as dictated by the description "*description*".
Identify the navigation state for GUI agents that is equivalent to the details provided in "*description*".
Search the observation for web navigation that best fits the details described in "*description*".
Determine the precise GUI navigation observation that reflects the input "*description*".
Retrieve the navigation observation for web agents that best aligns with the input "*description*".
What is the corresponding web browsing state described by the input "*description*"?
From the description "*description*", identify the specific navigation observation for GUI navigation.

1505 $q \rightarrow s_n$
1506

Retrieve the last web navigation observation based on the task "*description*".
From the task "*description*", locate the final state in the web navigation sequence.
What is the ultimate web navigation state for the task instruction "*description*"?
Find the end state in the GUI navigation as defined by the task "*description*".
Determine the last state in the web browsing process for the task "*description*".
Locate the final observation of the web navigation for the task "*description*".
Identify the concluding status in the GUI agent trajectory for the task "*description*".
Based on the task "*description*", extract the final navigation observation for web agents.
What is the final GUI navigation status according to the task "*description*"?
Search the terminal observation in the web navigation for the task instruction "*description*".

1512 E.2 PROMPT FOR WEBRAGENT'S PLANNING MODEL
15131514 The system prompt used by WebRAGent's planning model is given in Snippet 4, while Snippet 5
1515 presents the reward prompt and Snippet 6 presents the LLM-reranking prompt.
15161517 SNIPPET 4: SYSTEM PROMPT USED FOR PLANNING MODEL
15181519 **Prompt**1520 You are an assistant who not only helps to browse and operate web pages to achieve certain goals, but also needs to explore the
1521 information on the page to answer the questions raised by the target task. Please answer the following questions as much as
1522 possible.

```

1522     ## There are key information you will get
1523     **Key Information**:
1524         - Previous trace: all thoughts, actions and reflections you have made historically.
1525         - Current webpage screenshot: The webpage screenshot of the current execution step.
1526         - Similar Task Reference: The solution of a similar task, which can be appropriately used as a reference.
1527     You should always consider previous and subsequent steps and what to do. Do not ask for permission or confirmation.
1528     Act decisively.
1529     **Thought Space**:
1530         - What action do you think is needed now to complete the task?
1531         - What's the reason of taking that action?
1532     You also need to provide an effective description of the current execution action.
1533     A proper description contains:
1534         - What website it is;
1535         - Which action you choose;
1536         - REMEMBER DO NOT LEAVE THE DESCRIPTION EMPTY!
1537     ## ALLOWED ACTIONS
1538     - click, double_click, type, scroll, keypress, drag, wait, get_final_answer
1539     ## COMPLETION RULES
1540     - INFORMATION tasks: when all requested info is visible use get_final_answer with the content.
1541     - ACTION tasks: when final action succeeds and success signal/confirmation is visible get_final_answer.
1542     **Special Circumstances Guidelines**:
1543         - When performing a search on a website, if you find the search results do not display sufficient content, consider
1544             simplifying or modifying your search query. Reducing the complexity of your search query or altering keywords may
1545             yield more comprehensive results.
1546     ## RAG Usage Protocol (Do Not Violate)
1547     You may receive a list of retrieved example steps from a knowledge base. Each example contains:
1548         - task_name: the original task title
1549         - step_number: the step index in that task
1550         - observation_description: what the page looked like in that step
1551         - action_description: what was done and why
1552     ### How to use RAG (5-stage procedure)
1553     1) Goal Alignment:
1554         - Summarize the current sub-goal in one short clause (mentally).
1555         - If the examples sub-goal differs substantially (domain mismatch, authentication state mismatch, paywall/login
1556             state mismatch, or different product/site section), DO NOT reuse low-level details.
1557     2) Evidence Anchors (from the accessibility tree):
1558         - Extract concrete anchors you can actually verify now: role (link/button/input/select), visible text substrings, aria-
1559             label/hint words, URL host/path pattern, presence of known sections (nav/search/filter/cart/profile).
1560         - Only consider an example highly similar if 2 anchors match.
1561     3) Reuse Level Decision (choose exactly one):
1562         - **NONE** (similarity too low): Ignore example details. Use only generic high-level strategy.
1563         - **HIGH_LEVEL** (some similarity, <2 anchors matched): Borrow the intent/strategy but **do not** reuse
1564             element-level moves.
1565         - **DETAIL** (2 anchors match and same site section): Borrow the next action **type** (click/fill/select/go_back/
1566             goto) and **selection logic**, but you MUST re-locate the element by current accessibility tree (NEVER copy
1567             example element.id).
1568     4) Adaptation Plan:
1569         - Translate example-specific values into current-task values (e.g., search keywords, product IDs, email addresses).
1570         - If an example uses a search then filter pattern, replicate the pattern only if the corresponding widgets exist now.
1571     5) Safety & Fallbacks:
1572         - Never reuse example 'element.id'.
1573         - If no viable anchor is found within the current accessibility tree, downgrade reuse level and choose a safer generic
1574             action (e.g., open site search, go to category page, reveal filters).
1575         - If the last action failed, avoid repeating it; try a sibling anchor or a simpler query.
1576     ### Mandatory Reporting (inside your 'description')
1577     At the END of your 'description', append a single line tag block (keep the exact keys):
1578     '[RAG_USED=<yes/no>; RAG_REUSE=<NONE|HIGH_LEVEL|DETAIL>; RAG_SOURCES=<task_name:step,...> or ->; RAG_REASON=<short>; CONF=<0.00-1.00>]***'
1579     ## Response Format:
1580     - "thought": Your reasoning and immediate action plan (use decisive language, no questions or confirmations)
1581     - "action": The specific action to execute NOW
1582     - "action_input": Parameters for the action
1583     - "element_id": Target element identifier
1584     - "description": Brief description of the action being executed (not what you "could" do)
1585     - "example_reference": How you applied insights from examples (if applicable)
1586
1587 .....
```


1620

SNIPPET 6: PROMPT FOR LLM-RERANKING

1621

Prompt

1623

.....

1624

You are a web interface image matching expert. I will provide you with one query web interface image, a query task description, and $\{\text{len}(\text{candidates}.\text{text})\}$ candidate web interfaces, each with corresponding web interface images and task descriptions.

1625

Query Task Description: $\{\text{query}.\text{task}.\text{description}\}$

1626

Please carefully analyze the query image and candidate images, and match them based on task descriptions, including:

1627

- Interface elements (buttons, input fields, text, etc.)

1628

- Interface layout and design

1629

- Interface visual similarity

1630

- Matching degree between query task and candidate task descriptions

1631

Candidate Information:

1632

$\{\text{",join}(\text{candidates}.\text{text})\}$

1633

Please select the most matching candidate from the above options, considering the following main factors:

1634

1. Image visual similarity

1635

2. Interface element matching degree

1636

3. Relevance between query task and candidate task descriptions

1637

Please answer:

The most matching candidate number ($1-\{\text{len}(\text{candidates}.\text{text})\}$)

1638

Answer format:

Best matching candidate: [number]

.....

1640

1641

1642

1643

1644

F CASE STUDY

1645

F.1 CASE STUDY: TEXTUAL TRAJECTORIES OF SUCCESSFUL RETRIEVAL

1646

Below is the trajectory data (text) of some successful cases using the RAG method. Purple text denotes the models internal thoughts, while red text highlights the portions that reference retrieved knowledge.

1647

RAG Successful Case 1

1648

Task name: Browse recipes for gluten-free chocolate chip cookies that can be made without nuts.

1649

Step 1

1650

[think]

To search for gluten-free chocolate chip cookie recipes without nuts, I should enter a precise query into the search bar on the Cookpad homepage. **The example confirms this is the correct first interaction, recommending using the search bar with a clear query.**

1651

[/think]

1652

[action_history]

gluten-free chocolate chip cookies without nuts fill_form

1653

[/action_history]

1654

Step 2

1655

[think]

Since I have already entered the search query into the input for 'gluten-free chocolate chip cookies without nuts', the correct next step is to submit the search using the associated button to trigger recipe results. **The example confirms this is the standard flow.**

1656

[/think]

1657

[action_history]

button click

1658

[/action_history]

1659

1674

1675

1676

1677

1678

1679

1680

1681

1682

1683

1684

1685

1686

1687

1688

1689

1690

1691

1692

1693

1694

1695

1696

1697

1698

1699

1700

1701

1702

1703

1704

1705

1706

1707

1708

1709

1710

1711

1712

1713

1714

1715

1716

1717

1718

1719

1720

1721

1722

1723

1724

1725

1726

1727

Step 3

[think]

The search button has been clicked, so the next step is to review the displayed search results for recipes matching 'gluten-free chocolate chip cookies without nuts'. I should identify recipe cards or links that describe gluten-free chocolate chip cookies and assess their suitability, and if possible, further refine the filtering to exclude nuts using any visible filters or input boxes just as done in **the example, where a filter input for 'Show me recipes without' was used**. First, I need to look for any recipe list or individual results related to the search to begin this review.

[/think]**[action_history]**

Extract and cache the titles and descriptions of recipe links/results visible after submitting the search for 'gluten-free chocolate chip cookies without nuts', as well as any available filters or search refinement tools (such as input boxes for excluding ingredients). This data will help determine whether the resulting recipes fit the criteria (gluten-free and nut-free) and guide the next steps if further filtering is needed. cache_data

[/action_history]

Step 4

[think]

The search did not return a focused list of recipe results; instead, the page mostly displays links to popular ingredients and general featured recipes. There do not appear to be targeted search results or a visible recipe list matching 'gluten-free chocolate chip cookies without nuts.' Given that the relevant recipes are not surfaced, I should simplify the search - entering just 'gluten-free chocolate chip cookies' may return more comprehensive results or trigger filters, as shown in **the RAG example when initial queries do not immediately yield the desired recipe list**. After submission, I will check for an ingredient exclusion filter where 'nuts' can be entered. First, I will try making the search query less restrictive.

[/think]**[action_history]**

gluten-free chocolate chip cookies fill_form

[/action_history]

Step 5

[think]

The accessibility tree shows a 'Filters' link and also an input labeled 'Type ingredients...' under the recipe/filter section. To align with the workflow in the similar example, the next step is to use the available filter to exclude nuts. I should first click the 'Filters' link to open the filtering interface. After this, I can use the input field to specify 'nuts' for exclusion, ensuring the recipe results fit the user's dietary requirements.

[/think]**[action_history]**

link click

[/action_history]

Step 6

[think]

Now that the 'Filters' interface has been opened, my next step is to exclude 'nuts' using the appropriate filter input. The accessibility tree shows two input boxes labeled 'Type ingredients...' under the filter section. **Based on the example** and the standard UI flow, I should type 'nuts' into the second filter input to specify ingredients to exclude, as it most likely corresponds to 'Show me recipes without:' from the similar example. This will update the recipe list to only include gluten-free chocolate chip cookies without nuts.

[/think]**[action_history]**

nuts fill_form

[/action_history]

1728

1729

1730

Step 7

1731

[think]
 Now that the nut exclusion filter has been applied for 'gluten-free chocolate chip cookies', I should review the updated recipe list and open a few promising recipes to verify their ingredient lists for both gluten-free and nut-free compliance. Recipes such as 'GF Chocolate Chip Cookies', 'Gluten-Free Chocolate Chip Cookies', 'Gluten-Free! Chocolate Chip Cookies', and similar titles are likely candidates. I will select one, open the recipe, and check the detailed ingredients. This is exactly what the similar example did after setting the dietary filters.

1732

[/think]

1733

[action_history]

1734

link click

1735

[/action_history]

1736

Step 8

1737

[think]

1738

To validate if 'GF Chocolate Chip Cookies' are both gluten-free and nut-free, I need to review its ingredient list. I will extract and analyze the main listed ingredients now, specifically checking for any mention of wheat, barley, rye (gluten sources), or nuts including almond flour (as 'almond flour' is often used in gluten-free baking but is a tree nut, which may not meet the 'nut-free' requirement). I need to capture the precise ingredient data before determining suitability.

1739

[/think]

1740

[action_history]

1741

Extract and analyze the full ingredient list used in the 'GF Chocolate Chip Cookies' recipe to check for the presence of nuts or gluten-containing ingredients. cache_data

1742

[/action_history]

1743

Step 9

1744

[think]

1745

Having determined that 'GF Chocolate Chip Cookies' are not nut-free due to the inclusion of almond flour, the correct next step is to return to the recipe results list. I need to open another promising recipeone with a title like 'Classic Chocolate Chip Cookies', 'EZ Chocolate Chip Cookies', or 'Best Chocolate Chip Cookies' to review its ingredients and check compliance. I will select the next most promising recipe from the visible results list.

1746

[/think]

1747

[action_history]

1748

link click

1749

[/action_history]

1750

Step 10

1751

[think]

1752

To determine if 'Classic Chocolate Chip Cookies' by Joraine Villanueva meets the gluten-free and nut-free requirements, I need to extract and review the recipe's ingredient list. The key checkpoints: (1) if 'all purpose flour' is used, it contains gluten (so not gluten-free); (2) check for nuts or nut flours among the ingredients to ensure it is nut-free.

1753

[/think]

1754

[action_history]

1755

Extract the detailed ingredient list for 'Classic Chocolate Chip Cookies' by Joraine Villanueva: butter, granulated sugar, brown sugar, vanilla, all purpose flour, baking soda, salt, chocolate chips. Analyze for presence of gluten or nuts. cache_data

1756

[/action_history]

1757

Step 11

1758

[think]

1759

Having determined that the current recipe ('Classic Chocolate Chip Cookies') is not gluten-free (contains all-purpose flour), I should proceed to open the next promising recipe in

1760

1761

1762

1763

1764

1765

1766

1767

1768

1769

1770

1771

1772

1773

1774

1775

1776

1777

1778

1779

1780

1781

1782
 1783 the filtered results list with a gluten-free-focused title, such as 'Best Chocolate Chip Cookie',
 1784 and repeat the ingredient review process. This aligns closely with both the detailed workflow
 1785 and fallback loop in the provided **RAG examples** systematically opening subsequent recipes
 1786 and checking ingredients until a compliant gluten-free and nut-free option is found or all
 1787 candidates are exhausted.

1788 **[/think]**

1789 **[action_history]**

1790 link click

1791 **[/action_history]**

Step 12

1792 **[think]**

1793 I need to examine the ingredient list for the 'Best Chocolate Chip Cookie' recipe by Helen
 1794 Klassen to determine if it is both gluten-free and nut-free. Specifically, I am looking for
 1795 signs of gluten (such as wheat flour) or any nuts/nut-derived ingredients. The ingredient
 1796 section typically appears near prominent keyword links on Cookpad recipes, e.g., 'butter',
 1797 'brown sugar', 'flour', 'baking soda', 'chocolate chips'. I will extract these linked ingredient
 1798 keywords for analysis.

1799 **[/think]**

1800 **[action_history]**

1801 Extract the main ingredients for 'Best Chocolate Chip Cookie' by Helen Klassen, focusing
 1802 on links near ingredient-related keywords such as 'salted butter', 'brown sugar', 'vanilla',
 1803 'flour', 'baking soda', 'baking powder', 'salt', 'chocolate chips', and 'chocolate' to verify the
 1804 presence or absence of gluten and nuts. `cache_data`

1805 **[/action_history]**

RAG Successful Case 2

1810 **Task name:** Browse the latest negative reviews from players with over 100 hours of playtime
 1811 for the game that won the 2023 VR Game of the Year Award.

Step 1

1812 **[think]**

1813 To determine the 2023 VR Game of the Year winner, I should navigate to the 'Steam Awards'
 1814 section. I see a link labeled 'The Steam Awards' under 'New & Noteworthy,' which likely
 1815 leads to the awards list. Opening this should reveal the winning VR title, which is my
 1816 required first step before proceeding to reviews.

1817 **[/think]**

1818 **[action_history]**

1819 link click

1820 **[/action_history]**

Step 2

1821 **[think]**

1822 To identify the 2023 VR Game of the Year winner, I should closely examine the content of
 1823 this 'Steam Awards Archive' page for the relevant award and the corresponding game name
 1824 before proceeding to its store page to seek reviews.

1825 **[/think]**

1826 **[action_history]**

1827 Extract the name of the game that won the 2023 VR Game of the Year Award from this
 1828 'Steam Awards Archive' page. `cache_data`

1829 **[/action_history]**

Step 3

1830 **[think]**

1836

The critical next step is to determine which game won the 2023 VR Game of the Year Award by examining the text or content on the current 'The Steam Awards Archive' page. There isn't a direct text node or easy-to-access label in the accessibility tree showing the award winners, nor are there section headings or static content nodes revealing the information. Therefore, in the absence of any clearly labeled VR Game of the Year winner in the tree, I should use the page's search functionality to look up '2023 VR Game of the Year' or similar, so I can quickly find the specific winner's name before proceeding to its store page for the review substeps.

1844

[/think]

1845

[action_history]

1846

2023 VR Game of the Year fill_form

1847

[/action_history]

1848

Step 4

1849

[think]

1850

The current step is to identify the game that won the 2023 VR Game of the Year Award so I can proceed to negative reviews filtering. **From the RAG example, the user used the Steam search function with the query**

1851

[/think]

1852

[action_history]

1853

link 'Arizona Sunshine Remake' click

1854

[/action_history]

1855

Step 5

1856

[think]

1857

Now that I am on the Arizona Sunshine Remake store page, my immediate sub-goal is to display user reviews and apply filters. Based on the accessibility tree, there is a cluster of review filter buttons ('Review Type', 'Playtime', etc.) around element_ids 258-264. The first logical move is to open the 'Review Type' or 'Playtime' filter so I can start applying the conditions (over 100 hours, negative). **Following the example, the correct pattern is to interact with these filter controls. 'Playtime' is crucial for the 100+ hours condition, so I'll open that filter first.**

1858

[/think]

1859

[action_history]

1860

button 'Playtime' click

1861

[/action_history]

1862

Step 6

1863

[think]

1864

Having opened the 'Playtime' filter, the next logical step is to select the minimum hours played option and set it to 100 to isolate users with more than 100 hours. In the accessibility tree, there is a link labeled 'Minimum hours played' (element_id 265), which likely leads to an interface where the minimum threshold can be set. Selecting this will allow me to input or choose 100 as required for the task.

1865

[/think]

1866

[action_history]

1867

Minimum hours played click

1868

[/action_history]

1869

Step 7

1870

[think]

1871

Having clicked the 'Minimum hours played' option, my next step is to actually set the minimum value to 100, as required by the task. There does not appear to be a slider or dropdown, so the next logical interaction is to check for an input form or field to directly specify the hours. If present, I should type '100' into this field, fulfilling the playtime criteria. After this, I will continue with negative review filtering.

1872

[/think]

1873

[action_history]

1874

100 fill_form

1875

1876

1877

1878

1879

1880

1881

1882

1883

1884

1885

1886

1887

1888

1889

```

1890
1891 [action_history]
1892
1893 Step 8
1894 [think]
1895 Now that the playtime limit UI is open from previous steps, I need to locate and interact with
1896 a form or input field where I can enter '100' as the minimum hours played. There are no
1897 non-form filter sliders visible in the accessibility tree, so the likely next step is to find and fill
1898 any visible input for minimum hours with '100'. This mirrors the RAG example in setting a
1899 precise threshold, which is required before proceeding to filter for negative reviews.
1900 [/think]
1901 [action_history]
1902 100 fill_form
1903 [/action_history]
1904 Step 9
1905 [think]
1906 With the playtime filter set to a minimum of 100 hours, the next critical step is to filter
1907 reviews to only show negative ones. The accessibility tree shows a button labeled 'Review
1908 Type' (element_id 258) near the other filter controls. Clicking this button should open options
1909 (such as 'All', 'Positive', 'Negative'), allowing me to select 'Negative' so that only negative
1910 reviews from players with over 100 hours are displayed. This directly mirrors the example
1911 workflow after applying the playtime filter.
1912 [/think]
1913 [action_history]
1914 button click
1915 [/action_history]
1916 Step 10
1917 [think]
1918 Having opened the 'Review Type' filter, the next required move is to select 'Negative' so that
1919 reviews are limited to only negative ones from players with over 100 hours, as per the target
1920 task. This mirrors the action taken in the example after playtime criteria were applied.
1921 [/think]
1922 [action_history]
1923 Negative click
1924 [/action_history]
1925 Step 11
1926 [think]
1927 Now that the negative review filter has been activated, the final step is to sort these filtered
1928 reviews by 'most recent' so that the latest negative feedback from players with over 100
1929 hours is shown on the Arizona Sunshine Remake Steam store page. In the accessibility
1930 tree, there is a 'Display' button at element_id 267, which is likely to serve as a sorting
1931 or display options dropdown this follows the workflow from the similar example, where
1932 the reviews were explicitly sorted by latest/most recent after filtering by review type and
1933 playtime. Clicking this will allow me to ensure the reviews are displayed from newest to
1934 oldest, finishing the multi-filter process required by the task.
1935 [/think]
1936 [action_history]
1937 button click
1938 [/action_history]
1939 Step 12
1940 [think]
1941 The display/sort menu has just been opened (by clicking the 'Display' button at element_id
1942 267) and, per the workflow from both my prior steps and the highly similar RAG example,
1943 the critical next step is to explicitly choose the sort orderspecifically, to select 'Most Recent'
```

1944

1945 or 'Latest' from the available display/sort options. This will sort the already filtered reviews
 1946 (negative, 100+ hours) so that the most recent feedback is shown at the top, precisely meeting
 1947 the requirements of the task. The accessibility tree does not show child elements under
 1948 'Display,' so a click is necessary to reveal/select the sort order. Once sorted, I will be able to
 1949 confirm and browse these filtered, latest negative reviews.

1950 **[/think]**

1951 **[action_history]**

1952 Sort reviews by 'Most Recent' (likely via the display dropdown that was opened) click

1953 **[/action_history]**

1954

1955

1956

1957

1958 F.2 CASE STUDY: VISUAL TRAJECTORIES OF SUCCESSFUL RETRIEVAL

1959

1960 Tables 15–17 present successful cases under the Vision RAG mode in a visualized manner. For each
 1961 step, we provide the corresponding screenshot and Action information, enabling straightforward and
 1962 intuitive inspection.

1963

1964

1965

1966

1967

1968

1969

1970

1971

1972

1973

1974

1975

1976

1977

1978

1979

1980

1981

1982

1983

1984

1985

1986

1987

1988

1989

1990

1991

1992

1993

1994

1995

1996

1997

1998

1999

2000

2001

2002

2003

2004

2005

2006

2007

2008

2009

2010

2011

2012

2013

2014

2015

2016

2017

2018

2019

2020

2021

2022

2023

2024

2025

2026

2027

2028

2029

2030

2031

2032

2033

2034

2035

2036

2037

2038

2039

2040

2041

2042

2043

2044

2045

2046

2047

2048

2049

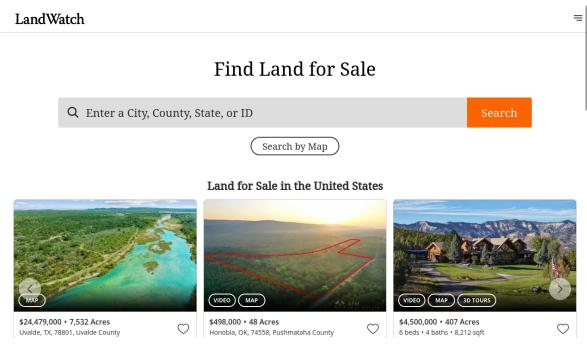
2050

2051

Table 15: Vision RAG Success Example Steps 0-2

Instruction: Find Farms land in Wilkes County, NC with the lowest price.

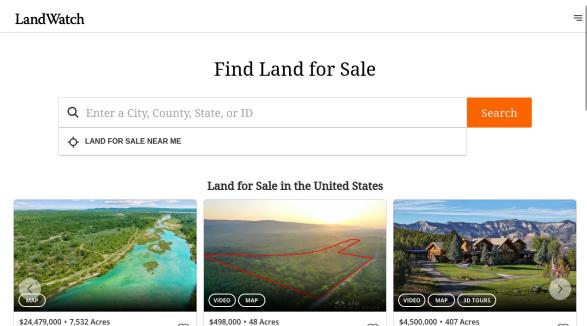
Observation 0



Action 1

operator_click(328, 233)

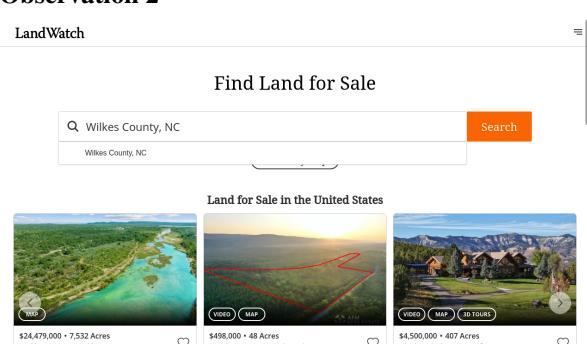
Observation 1



Action 2

operator_type(Wilkes County, NC)

Observation 2

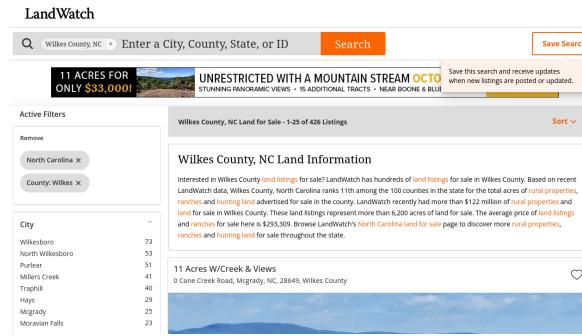


Action 3

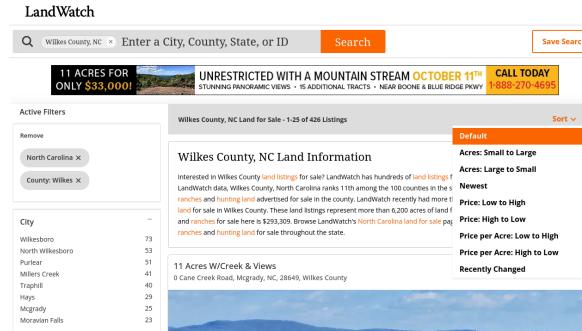
operator_click(445, 292)

2052
2053
2054
2055
2056
2057

Table 16: Vision RAG Success Example Steps 3-5

Observation 3**Action 4**

operator_click(1219, 255)

Observation 4**Action 5**

operator_click(1005, 440)

Observation 5**Action 6**

operator_scroll(0, 420)

2100
2101
2102
2103
2104
2105

2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117

Table 17: Vision RAG Success Example Steps 6-7

Observation 6

LandWatch

Wilkes County, NC Land for Sale - 1-25 of 426 Listings

Wilkes County, NC Land Information

Wilkesboro 73
North Wilkesboro 53
Purlear 51
Mills Creek 41
Traphill 40
Hays 29
McGrady 25
Moreau Falls 23

11 ACRES FOR ONLY \$33,000!

UNRESTRICTED WITH A MOUNTAIN STREAM OCTOBER 11th STUNNING PANORAMIC VIEWS • 15 ADDITIONAL TRACTS • NEAR BOONE & BLUE RIDGE PKWY

CALL TODAY! 866-270-4855

Sort ▾

Sort v

Contact Seller

Action 7`operator.scroll(0, 420)`**Observation 7**

LandWatch

Wilkesboro 73
North Wilkesboro 51
Purlear 41
Mills Creek 40
Traphill 36
Hays 26
McGrady 25
Moreau Falls 23

See More

Price

\$0 - \$49,999 98
\$50,000 - \$99,999 65
\$100,000 - \$249,999 99
\$250,000 - \$499,999 86
\$500,000 - \$749,999 38
\$750,000 - \$999,999 22
\$1,000,000 and up 13

Custom Price

Parcel Size

0 - 10 Acres 108

Sort ▾

Sort v

Contact Seller

Action 8`operator.wait(1000)`

2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159

2160 F.3 TEXTUAL COMPARISON: RAG vs. Non-RAG
21612162 We present below, in textual form, whether RAG is utilized or not, followed by a detailed comparative
2163 analysis.2164 RAG vs. Non-RAG 1
21652166 **Task name:** Browse iPhone X for sale that is in good condition, has a max price of 400, and
2167 searches in titles only.2168 **RAG Step 1**
21692170 **[think]**
2171 Based on the typical Craigslist flow and the strong match with the example, my next step
2172 should be to enter 'iPhone X' into the search box to start browsing listings relevant to the
2173 task requirements. This aligns with the similar example, where the search begins by entering
2174 the product name.
21752176 **[/think]**
2177 **[action_history]**
2178 iPhone X fill_form
2179 **[/action_history]**2180 **RAG Step 2**
21812182 **[think]**
2183 The next step is to filter the results by 'good' condition since we've just completed entering
2184 the search term. The example shows that clicking the 'condition' filter button will bring up
2185 the options for selecting item states like 'good.' This must be done before applying other
2186 filters.
21872188 **[/think]**
2189 **[action_history]**
2190 button 'condition' click
2191 **[/action_history]**2192 **RAG Step 3**
21932194 **[think]**
2195 The condition filter pop-up is now open, so I need to select the 'good' condition option. The
2196 RAG example shows this as the immediate next step. After checking 'good', I'll apply the
2197 filter, then advance to pricing and titles-only filters.
21982199 **[/think]**
2200 **[action_history]**
2201 Select 'good' within the condition filter pop-up to filter listings to only those described as in
2202 'good' condition. click
2203 **[/action_history]**2204 **RAG Step 4**
22052206 **[think]**
2207 With the 'good' condition selected, the next RAG-aligned step is to click 'apply' to confirm
2208 the filter within the pop-up, just as in the example. This ensures the search is updated to only
2209 show items in good condition before proceeding to max price and titles-only filters.
22102211 **[/think]**
2212 **[action_history]**
2213 button click
2214 **[/action_history]**2215 **RAG Step 5**
22162217 **[think]**
2218 With the 'good' condition filter applied, the next step is to set the maximum price by
2219 entering '400' into the appropriate input field. According to both the RAG example and
2220 the accessibility tree, the 'max' input field is available and ready for interaction. This will
2221 further refine the search to only include iPhone X listings with prices that meet the specified
2222 maximum.
2223

2214

[action_history]
400 fill_form
[/action_history]

RAG Step 6

2219

[think]
With the 'good' condition and max price filters set, the last required filter is 'search in titles only.' This is usually a checkbox near the search box or within sidebar filters on Craigslist. Scanning the provided accessibility tree, the related option isn't immediately visible, suggesting it may be in a collapsed, sidebar, or advanced filter section often found near the main search input. **To proceed like in the RAG example**, I should look for a button or section that could expand, reveal, or activate additional search options, especially one near the search form. The 'submit' button (element_id 11) is next to the 'search for sale' input and is a likely action to refresh/apply current filters and surface any missing advanced options, especially if UI renders more filtering tools after submission. **Clicking it is a safe way to expose or set the titles-only filter, aligning with common Craigslist interface patterns and the analogous example.**

2230

[/think]

[action_history]
button 'submit' click
[/action_history]

RAG Step 7

2235

[think]
The 'search in titles only' option is a required filter for this task, but it's not visibly present in the current accessibility tree even after refreshing search filters. The rest of the filters (max price and condition) are already set. Since there's no titles-only checkbox, **the best next step in reference to Craigslist's interface (and the RAG example, where this option is often in the sidebar near search or under advanced search options)** is to look for any additional buttons or expandable areas near the search input or filter section that could reveal more search options. The 'for sale' filter button (element_id 4) is at the top of the category/filter controls, and sometimes Craigslist nests advanced filters under category/section buttons. By clicking this, I may reveal or trigger additional options, including the sought 'titles only' filter.

2245

[/think]

[action_history]
button click
[/action_history]

RAG Step 8

2249

[think]
To comprehensively address the remaining requirement of 'search in titles only', I need to perform a final thorough check for any advanced, sidebar, or otherwise hidden filter settings that could reveal a corresponding option since nothing titled 'titles only' appears in the main or expanded filter controls thus far. Scanning all buttons near the filter/search area (such as the gear/settings icon, secondary search controls, or miscellaneous options buttons) is the most methodical approach. The 'related searches' button (element_id 29) stands out, as sometimes additional or advanced filters are exposed via such miscellaneous controls. Activating this may reveal an area not currently visible in the accessibility tree and **aligns with the example's step of checking all sidebar/advanced controls**. If it doesn't reveal a titles-only option, this will serve as final confirmation that the filter is unavailable in this Craigslist region, allowing the task to be concluded accordingly.

2261

[/think]

[action_history]
button click
[/action_history]

Non-RAG Step 1

2265

[think]

2268

2269 To browse iPhone X for sale, I need first to navigate to the 'for sale' category, and more
 2270 specifically to 'cell phones', to better target the listings. This will allow me to subsequently
 2271 apply relevant filters for condition, price, and title search.

2272 **[/think]**

2273 **[action_history]**

2274 link click

2275 **[/action_history]**

2276

Non-RAG Step 2

2277 **[think]**

2278 Now that I'm in the 'cell phones' for sale section of Craigslist, I need to locate and interact
 2279 with the search form. This will allow me to enter 'iPhone X' as the keyword, filter for 'good'
 2280 condition, set the max price to 400, and select 'search titles only'.

2281 **[/think]**

2282 **[action_history]**

2283 iPhone X fill_form

2284 **[/action_history]**

2285

Non-RAG Step 3

2286 **[think]**

2287 I've previously indicated the intent to enter 'iPhone X' into the search input. Now, to further
 2288 meet the requirements, I should fill in the 'max' input with '400' to set the maximum price.
 2289 This incrementally builds the required search criteria.

2290 **[/think]**

2291 **[action_history]**

2292 400 fill_form

2293 **[/action_history]**

2294

Non-RAG Step 4

2295 **[think]**

2296 My next step is to enter 'iPhone X' in the search input to begin filtering for listings matching
 2297 the target device. The search input has element_id 10, and this will set up the query before
 2298 applying other filters like condition and title search.",

2299 **[/think]**

2300 **[action_history]**

2301 iPhone X fill_form

2302 **[/action_history]**

2303

Non-RAG Step 5

2304 **[think]**

2305 To continue with the next most logical step and further refine the search as specified, I should
 2306 now click the 'condition' button (element_id 18) so that I can select 'good' as the target
 2307 condition for listings. This directly addresses a key filter in the task description.

2308 **[/think]**

2309 **[action_history]**

2310 iPhone X fill_form

2311 **[/action_history]**

2312

Non-RAG Step 6

2313 **[think]**

2314 Now that the 'condition' filter is open, the next logical action is to select 'good' as the target
 2315 condition for the listings. This will ensure that only listings in good condition are shown, as
 2316 required by the task.

2317 **[/think]**

2318 **[action_history]**

2319 button click

2320 **[/action_history]**

2321

Non-RAG Step 7

2322

2323

2324

2325

2326

2327

2328

2329

2330

2331

2332

2333

2334

2335

2336

2337

2338

2339

2340

2341

2342

2343

2344

2345

2346

2347

2348

2349

2350

2351

2352

2353

2354

2355

2356

2357

2358

2359

2360

2361

2362

2363

2364

2365

2366

2367

2368

2369

2370

2371

2372

2373

2374

2375

[think]

The 'condition' filter panel is open, so now I need to choose 'good' as the desired condition. This action is essential to limit results to iPhone X listings that are in good condition, as required by the task. Based on the accessibility tree, the interface may use button selections for conditions. Likely options such as 'all', 'owner', or 'dealer' do not fit 'good' condition; the 'condition' filter may require another button press or direct text selection, but 'apply' is typically used to confirm a selection. Therefore, the correct next step is to click 'apply' (element_id 3) to confirm the current 'good' condition setting and return to the main filter area.

[/think]**[action_history]**

apply click

[/action_history]**Non-RAG Step 8****[think]**

Now that 'good' condition should be applied, the next step is to look for the 'search titles only' option. Since there does not appear to be an explicit 'titles only' checkbox in the accessibility tree, the most likely way to proceed is to try the main 'submit' button (element_id 14) to run the search with all entered criteria. Sometimes Craigslist restricts 'titles only' to advanced search, or it is embedded as an option within the main search submission. This step is necessary to see if the results page or subsequent filters will expose the 'search titles only' option, or if search results already reflect it.

[/think]**[action_history]**

submit click

[/action_history]**Non-RAG Step 9****[think]**

Since there still does not appear to be a 'search titles only' checkbox or filter explicitly shown in the accessibility tree, the next step is to check for advanced search options, nearby filters, or contextual help that may reveal or explain this feature. There is a 'search cell phones' input and standard Craigslist UI sometimes places a 'titles only' checkbox near the search input but it's not present here. I will investigate the nearby filter buttons (e.g., 'condition', 'mobile OS'), or look for additional options or advanced search links. Clicking 'related searches' (element_id 26) may also reveal advanced filtering options or extra search configurations including titles-only selection.

[/think]**[action_history]**

button click

[/action_history]**Non-RAG Step 10****[think]**

I've checked for 'related searches' and did not see a 'search titles only' option in the accessibility tree. There are no dedicated buttons or checkboxes labeled for searching in titles only near the search input or in filters. The logical next move is to closely examine the displayed search results and see if the search results are already filtered by title (i.e., do they display posts with 'iPhone X' in their titles?). This will confirm if further filtering is necessary or if the system already applies a titles-only filter by default. To proceed, I will scan the search result section for listings, titles, or headers matching 'iPhone X.' If there are no search results, I will document this and reflect that either no matches exist or the query returns no results under the given filters. Browsing the actual results is essential to complete the assessment.

[/think]**[action_history]**

Extract the displayed search results and their titles for the 'iPhone X' query under the

2376

2377 specified filters. Determine if any listings are returned and check if 'iPhone X' appears in the
 2378 titles, confirming if Craigslist's search function matches title-only by default or if no listings
 2379 match. cache_data
 2380 [/action_history]

Analysis

1. **DOM-RAG-Assisted Agent Performance:** The RAG-assisted agent demonstrated a clear, methodical approach, consistently validating each step against a known-good workflow. Its cognitive trace is replete with references to the "**RAG example**" and "**typical Craigslist flow**." This external knowledge provided a precise sequence for applying multiple filters: `search term condition apply max price titles only`. Crucially, when the "**titles only**" option was not immediately visible, the RAG example guided the agent to explore all sidebar and advanced controls, a sophisticated heuristic that a non-guided agent might miss.
2. **DOM-Non-RAG Agent Performance:** The Non-RAG agent attempted a different, less efficient strategy by first navigating to a sub-category (`cell phones`). Its filtering process was disorganized and redundant, for example, filling the "**iPhone X**" search term multiple times. Most importantly, when it could not find the "**search in titles only**" checkbox, it lacked a clear strategy. Its final action was to give up on filtering and attempt to infer the answer by manually inspecting the results.
3. **Role of the RAG System:** The RAG system's primary role was to provide a correct and robust procedural script for multi-filter application. It transformed the task from a chaotic exploration of UI elements into a deterministic, sequential process. It also supplied the vital heuristic for locating hidden or advanced options, preventing the agent from prematurely abandoning a crucial task constraint.

2381

2382

2383

2384

2385

2386

2387

2388

2389

2390

2391

2392

2393

2394

2395

2396

2397

2398

2399

2400

2401

2402

2403

2404

2405

2406

2407

RAG vs. Non-RAG 2

2408

2409

Task name: Show me the comparison of the first two personal credit cards that do not charge foreign transaction fees.

RAG Step 1

2410

2411

[think]

To progress towards comparing the first two personal credit cards with no foreign transaction fees, I need to access the list of personal credit cards. The best approach is to click 'View All Credit Cards' under the 'Personal Cards' sectionthis should display a comprehensive card selection, including features relevant to the task.

[/think]

[action_history]

View All Credit Cards click

[/action_history]

RAG Step 2

2420

2421

2422

[think]

The next required action is to click the 'No Foreign Transaction Fee' button to filter the list of personal credit cards to only those without foreign transaction fees. **This aligns with my plan and the example workflow**, allowing us to efficiently identify eligible cards for comparison.

[/think]

[action_history]

No Foreign Transaction Fee filter button click

[/action_history]

2423

2424

2425

2426

2427

2428

2429

2430

2431

2432

RAG Step 3**[think]**

Now that I am on the American Express personal credit card selection page, the immediate action is to apply the 'No Foreign Transaction Fee' filter. This will show only those personal credit cards that meet the user's criteria, making it easy to select the first two for comparison in subsequent steps.

[/think]**[action_history]**

No Foreign Transaction Fee filter button click

[/action_history]**RAG Step 4****[think]**

Having already identified the 'No Foreign Transaction Fee' filter button, the logical next step is to apply this filter to show only the relevant credit cards. By clicking this filter now, the listing will update to display only personal credit cards that do not charge foreign transaction fees, enabling me to easily select the first two for side-by-side comparison as required by the task.

[/think]**[action_history]**

No Foreign Transaction Fee button click

[/action_history]**RAG Step 5****[think]**

The next immediate step is to select the first credit card for comparison by clicking its 'Compare' button. The accessibility tree shows each card listing has a 'Compare' button (button role, label 'Compare'), and the sequence in the list matches the task requirement to select the first two. By clicking the first available 'Compare' button after applying the 'No Foreign Transaction Fee' filter, I will set up for a side-by-side comparison.

[/think]**[action_history]**

Compare click

[/action_history]**RAG Step 6****[think]**

To proceed, I need to select the second credit card for comparison by clicking the 'Compare' button for the next card in the filtered list. Having already clicked 'Compare' on the first card (element_id 153), the second card's 'Compare' button is the next in sequence, which corresponds to element_id 169. **This step will mark the two cards so I can use the comparison feature to display them side-by-side.**

[/think]**[action_history]**

button click

[/action_history]**RAG Step 7****[think]**

After clicking 'Compare' for the first card in the list, the next logical step is to select the 'Compare' button for the second card in the filtered results. This will mark both cards for side-by-side comparison. Once both are marked, I will look for a 'Compare Cards' button to generate the comparison. As the click for the first 'Compare' succeeded, I now proceed to the second card in sequence.

[/think]**[action_history]**

2482

2483

2484

2485 Compare button for the second card in the filtered list click
 2486 **[/action_history]**

2487

RAG Step 8

2488

[think]
 Since clicking the 'Compare' button for the first card resulted in an error, it's possible that the standard multi-step compare workflow may not function as expected. However, the accessibility tree reveals that each card panel has 'View Details' links adjacent to their 'Compare' button. Instead of relying on the compare overlay, I can open both cards' detail pages (in new tabs or windows) by clicking the first two 'View Details' links after applying the 'No Foreign Transaction Fee' filter. This enables me to manually collect and present their fees, rewards, and other features side-by-side, **fulfilling the main intent of the comparison**. Therefore, the next step is to click 'View Details' for the first card.

2489

[/think]2490 **[action_history]**

2491 View Details click

2492 **[/action_history]**

2493

RAG Step 9

2494

[think]

The current step is to gather details for the second card in the list of personal credit cards with no foreign transaction fees. The previous step opened the detail page for the first card, so now I need to click the 'View Details' link for the second eligible card (after the first one just viewed). This will allow me to manually compare the two cards' fees, rewards, and benefits.

[/think]2495 **[action_history]**

2496 View Details click

2497 **[/action_history]**

2498

Non-RAG Step 1

2499

[think]

To proceed with the task, I need to access the section which lists all personal credit cards. The most logical action is to click the 'View all Personal Cards' link, as this should provide a comprehensive view of card options, enabling me to identify those with no foreign transaction fees.

[/think]2500 **[action_history]**

2501 View all Personal Cards click

2502 **[/action_history]**

2503

Non-RAG Step 2

2504

[think]

I've reached the section showing all personal cards, and there are multiple 'Compare Card' buttons. To find cards with no foreign transaction fees, I need to examine the details. The best next step is to cache the feature blurbs or descriptive text next to the first few 'Compare Card' buttons, so I can check for a 'no foreign transaction fees' statement and then select the first two eligible cards for comparison.

[/think]2505 **[action_history]**

Extract all visible card names and their visible features/benefits, especially relating to foreign transaction fees, in the region near the first few 'Compare Card' buttons (element_ids 231, 235, 239, 256). Focus on feature lists or descriptions mentioning 'no foreign transaction fees' or similar wording. cache_data

[/action_history]

2506

2507

2508

2538

2539

2540

2541

2542

2543

2544

2545

2546

2547

2548

2549

2550

2551

2552

2553

2554

2555

2556

2557

2558

2559

2560

2561

2562

2563

2564

2565

2566

2567

2568

2569

2570

2571

2572

2573

2574

2575

2576

2577

2578

2579

2580

2581

2582

2583

2584

2585

2586

2587

2588

2589

2590

2591

Non-RAG Step 3

[think]

The accessibility tree shows several 'Compare Card' buttons (element_ids 231, 235, 239, 256, etc.) next to 'Learn More' links, likely each for a different credit card. However, the tree does not provide detailed feature text about each card directly. To identify cards with no foreign transaction fees, I need to examine the visible details/benefits associated with each card. The wisest next step is to cache (extract and capture) the feature blurbs or descriptive text next to the first few 'Compare Card' buttons, so I can check for a 'no foreign transaction fees' statement and then select the first two eligible cards for comparison.

[/think]**[action_history]**

Extract the descriptive and feature text or blurbs associated with cards having the first few 'Compare Card' buttons (element_ids 231, 235, 239, 256) to identify which explicitly offer no foreign transaction fees. cache_data

[/action_history]

Non-RAG Step 4

[think]

Having already cached relevant card feature descriptions, the next logical step is to identify the first two personal credit cards with no foreign transaction fees. Then, use the existing 'Compare Card' buttons to initiate a direct side-by-side comparison of those two cards, if the website supports comparing two cards simultaneously. This involves clicking 'Compare Card' for the first qualifying card, then for the second, to trigger the comparison functionality.

[/think]**[action_history]**

Compare Card button click

[/action_history]

Non-RAG Step 5

[think]

Since the previous step involved clicking 'Compare Card' for the first qualifying card, the next logical action is to click 'Compare Card' for the second eligible card (adjacent to element_id 235). This should trigger the built-in comparison view for both cards side-by-side, allowing for direct comparison, which aligns with the user's request.

[/think]**[action_history]**

Compare Card click

[/action_history]

Non-RAG Step 6

[think]

After clicking 'Compare Card' for the first two qualifying cards, the site should present a side-by-side comparison, either as an overlay or as a new section. The most effective next step is to extract and summarize this comparison, highlighting whether both cards have no foreign transaction fees and displaying their key features, thus directly answering the user's request.

[/think]**[action_history]**

Extract the contents of the side-by-side comparison table or feature display for the two personal credit cards just selected, with a focus on their foreign transaction fees and main benefits. cache_data

[/action_history]

Analysis

1. DOM-RAG-Assisted Agent Performance: The RAG-assisted agent demonstrated a clear, methodical approach, consistently validating each step against a known-good workflow. Its

2592
2593
2594
2595
2596
2597

cognitive trace is replete with references to the "RAG example" and "typical Craigslist flow." This external knowledge provided a precise sequence for applying multiple filters: search term condition apply max price titles only. Crucially, when the "titles only" option was not immediately visible, the RAG example guided the agent to explore all sidebar and advanced controls, a sophisticated heuristic that a non-guided agent might miss.

2598
2599
2600
2601
2602
2603
2604

2. DOM-Non-RAG Agent Performance: The Non-RAG agent attempted a different, less efficient strategy by first navigating to a sub-category (cell phones). Its filtering process was disorganized and redundant, for example, filling the "iPhone X" search term multiple times. Most importantly, when it could not find the "search in titles only" checkbox, it lacked a clear strategy. Its final action was to give up on filtering and attempt to infer the answer by manually inspecting the results.

2605
2606
2607
2608
2609
2610
2611
2612
2613

3. Role of the RAG System: The RAG system's primary role was to provide a **correct and robust procedural script for multi-filter application**. It transformed the task from a chaotic exploration of UI elements into a deterministic, sequential process. It also supplied the vital heuristic for locating hidden or advanced options, preventing the agent from prematurely abandoning a crucial task constraint.

2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645

RAG vs. Non-RAG 3

Task name: Check the specifications of the best-selling HP FHD laptop with 16 GB RAM and core i7 running on Windows 11.

RAG Step 1

[think]

To proceed, I need to click the search bar on the Best Buy homepage so I can enter a query for HP FHD laptops. This **matches the next logical step in the provided example** and aligns with my overall plan.

[/think]

[action_history]

Search Best Buy click

[/action_history]

RAG Step 2

[think]

To advance the task, I need to enter the search query 'HP FHD laptop 16GB RAM Core i7 Windows 11' into the search bar to find relevant products matching all criteria. This **matches the next logical action from the similar example** and my own planned approach.

[/think]

[action_history]

HP FHD laptop 16GB RAM Core i7 Windows 11 fill_form

[/action_history]

RAG Step 3

[think]

To ensure the search results match all target criteria (HP brand, 16GB RAM, Core i7, Windows 11), I need to use the sidebar filters for HP (brand), 16GB RAM, and Intel Core i7 processor. This **directly matches the high-level approach from the similar example** and ensures the results are not cluttered with non-matching models, maximizing my chance to identify the best-selling product quickly.

[/think]

2646
 2647
 2648
 2649
 2650
 2651
 2652
 2653
 2654
 2655
 2656
 2657
 2658
 2659
 2660
 2661
 2662
 2663
 2664
 2665
 2666
 2667
 2668
 2669
 2670
 2671
 2672
 2673
 2674
 2675
 2676
 2677
 2678
 2679
 2680
 2681
 2682
 2683
 2684
 2685
 2686
 2687
 2688
 2689
 2690
 2691
 2692
 2693
 2694
 2695
 2696
 2697
 2698
 2699

[action_history]
 checkbox click
[/action_history]

RAG Step 4

[think]
 To further refine the search results and meet the task specifications, I need to apply the 16GB RAM and Intel Core i7 processor filters from the sidebar. This ensures only relevant HP laptops with required memory and CPU are shown, minimizing irrelevant results before sorting by best selling. Applying RAM first will remove non-matching options.
[/think]

[action_history]
 checkbox click
[/action_history]

RAG Step 5

[think]
 The next step is to apply the Intel Core i7 processor filter in the sidebar to further narrow down to only those HP laptops with 16GB RAM and a Core i7, as this matches one of the primary task requirements. This will minimize irrelevant results before I further filter by Windows 11 and look for the best seller.
[/think]

[action_history]
 checkbox click
[/action_history]

RAG Step 6

[think]
 The next step is to apply the 'Windows' OS filter in the sidebar to ensure the search results only show HP laptops with 16GB RAM, Intel Core i7, and running Windows meeting all main requirements. This will further refine the product list before sorting for best selling and reviewing individual specs.
[/think]

[action_history]
 checkbox 'Windows' click
[/action_history]

RAG Step 7

[think]
 Now that all relevant filters are applied (HP brand, 16GB RAM, Intel Core i7, Windows), I need to sort the search results by 'Best Selling' to easily identify the best-selling HP FHD laptop that matches all the required criteria. The sort button is usually labeled something like 'Best Match' or similar, and this matches the next step in the example which targets the sort dropdown to select 'Best Selling'.
[/think]

[action_history]
 button click
[/action_history]

RAG Step 8

[think]
 After sorting by 'Best Selling', I need to select the first laptop listing in the filtered results, as it should be the top-selling HP FHD laptop with 16GB RAM, Core i7, and Windows 11. This mirrors the approach from the example where the product link at the top of the list is chosen after best-selling sorting, enabling review of full specifications on the detail page.
[/think]

2700
 2701 [action_history]
 2702 link click
 2703 [/action_history]
 2704
 2705 **RAG Step 9**
 2706 [think]
 2707 To complete the task, I need to review the 'Specifications' tab on the HP laptop's detail page.
 2708 This action will enable me to confirm and summarize all required technical details (FHD
 2709 display, 16GB RAM, Core i7, Windows 11) as in the **example**, thus fulfilling the original
 2710 **requirement**.
 2711 [/think]
 2712 [action_history]
 2713 Specifications tab click
 2714 [/action_history]
 2715
 2716 **Non-RAG Step 1**
 2717 [think]
 2718 To proceed to the U.S. version of Best Buy in English, I should click the 'United States' link
 2719 under the 'English' section. This will grant access to the main site and allow me to perform
 2720 targeted searches for HP FHD laptops.
 2721 [/think]
 2722 [action_history]
 2723 link 'United States' click
 2724 [/action_history]
 2725
 2726 **Non-RAG Step 2**
 2727 [think]
 2728 To begin the process of searching for the best-selling HP FHD laptop with 16 GB RAM and
 2729 core i7 running on Windows 11, I need to use the site's search bar. This will allow me to
 2730 input a refined query (e.g., 'HP laptop 16GB RAM Core i7 Windows 11 FHD'), after which I
 2731 can further filter and sort the results.
 2732 [/think]
 2733 [action_history]
 2734 HP laptop 16GB RAM Core i7 Windows 11 FHD fill_form
 2735 [/action_history]
 2736
 2737 **Non-RAG Step 3**
 2738 [think]
 2739 To accurately fulfill the task, I need to ensure that the search results are sorted by Best
 2740 Selling. The first visible sort option button is labeled 'Best Match', which typically reveals
 2741 sorting options; I should click it and then choose 'Best Selling' if available. Afterward, I will
 2742 proceed to visit the link for the top-listed HP laptop for specification details.
 2743 [/think]
 2744 [action_history]
 2745 button 'Best Match' click
 2746 [/action_history]
 2747
 2748 **Non-RAG Step 4**
 2749 [think]
 2750 To fulfill the task, I need to check the detailed specifications for the best-selling HP FHD
 2751 laptop with 16 GB RAM and Core i7 running Windows 11. The first result - '15.6...'
 2752 [/think]
 2753 [action_history]
 2754 link click
 2755 [/action_history]

2754

2755

2756

2757

2758

2759

2760

2761

2762

2763

2764

2765

2766

2767

2768

2769

2770

2771

2772

2773

2774

2775

2776

2777

2778

2779

2780

2781

2782

2783

2784

2785

2786

2787

2788

2789

2790

2791

2792

2793

2794

2795

2796

2797

2798

2799

2800

2801

2802

2803

2804

2805

2806

2807

Non-RAG Step 5

[think]

To complete the task, I need to check the detailed specifications of the laptop to confirm the operating system is Windows 11 and to verify the other specs. The 'Specifications' button is likely to provide this information. Therefore, I should click the 'Specifications' button to get the relevant details.

[/think]**[action_history]**

Specifications click

[/action_history]

Non-RAG Step 6

[think]

To complete the task, I need to review the full specifications of the best-selling HP FHD laptop with 16GB RAM and Core i7 running Windows 11. The accessibility tree shows a 'Specifications' button (element.id 47) which likely leads to a dedicated section or tab with these details. Since clicking it failed previously due to a selector error, I should manually cache or review the key accessible elements already visible since much of the relevant specifications (screen size, resolution, RAM, CPU, OS) are shown as buttons. I can extract these for the final answer.

[/think]**[action_history]**

Processor: Intel 13th Gen Core i7, RAM: 16GB, Display: 15.6-inch Full HD (1920x1080), Storage: 512GB SSD, Graphics: Intel Iris Xe, Operating System: (to confirm, check visible description/specs but expect Windows 11), Product: HP 15.6\\ cache_data

[/action_history]

Analysis

1. DOM-RAG-Assisted Agent Performance: The RAG-assisted agent executed a highly structured and comprehensive plan. It understood that a complex query requires a **multi-step filtering and sorting process**, not just a single search. Its thoughts consistently reference an "**example**", showing it followed a validated workflow: search filter by brand filter by RAM filter by processor filter by OS sort by best-selling select top result view specifications. This systematic application of constraints ensured the final result was highly accurate and met all task requirements.

2. DOM-Non-RAG Agent Performance: The Non-RAG agent's approach was far more simplistic and ultimately incomplete. It relied entirely on a single, long search query to do all the filtering work. While it did attempt to sort the results, it failed to apply the necessary sidebar filters for RAM, processor, and OS. This means its "best-selling" result was likely incorrect because it was chosen from a pool of items that did not meet all the user's criteria.

3. Role of the RAG System: RAG's crucial function was to provide the **knowledge of a complete, multi-step filtering workflow**. It taught the agent that for complex e-commerce queries, relying solely on the search bar is insufficient. Instead, a robust strategy involves sequentially applying dedicated filters to systematically narrow down the product space before making a final selection. The Non-RAG agent lacked this strategic depth, leading it to a superficially correct but fundamentally flawed result.

2808

Table 18: RAG Success Example Step 0-2

2809

2810

2811

2812

2813

2814

2815

2816

2817

2818

2819

2820

2821

2822

2823

2824

2825

2826

2827

2828

2829

2830

2831

2832

2833

2834

2835

2836

2837

2838

2839

2840

2841

2842

2843

2844

2845

2846

2847

2848

2849

2850

2851

2852

2853

2854

2855

2856

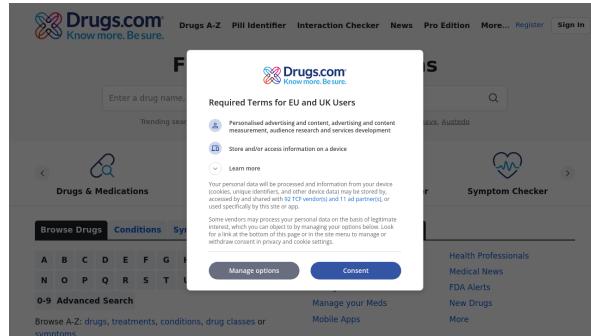
2857

2858

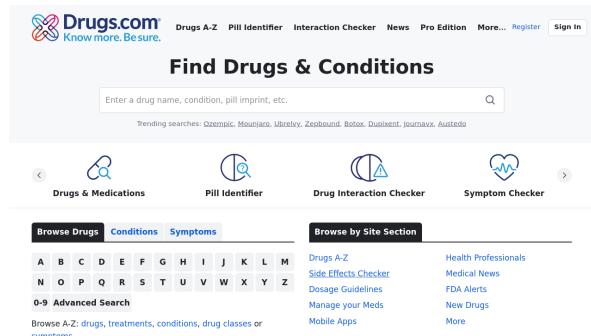
2859

2860

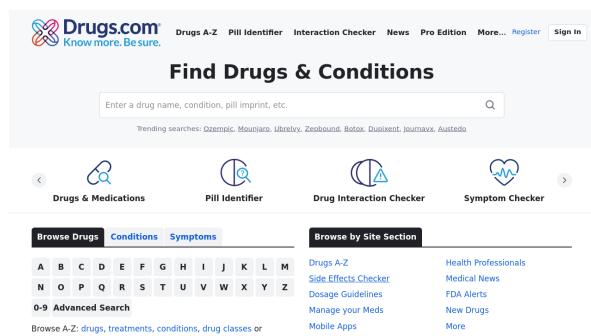
2861

Instruction: Check drug interaction for melatonin and Folate Forte.**Observation 0****Action 1**

operator.click(752, 576)

Observation 1**Action 2**

operator.wait(1000)

Observation 2**Action 3**

operator.click(766, 369)

F.4 VISUAL COMPARISON: RAG vs. NON-RAG

Tables 18–24 present screenshot comparisons illustrating the differences with and without RAG, followed by a detailed comparative analysis at the end.

2862
2863
2864
2865
2866
2867

Table 19: RAG Success Example Step 3-5

Observation 3
Action 4`operator_wait(1000)`**Observation 4**
Action 5`operator_click(233, 461)`**Observation 5**
Action 6`operator_type(melatonin)`

2909
2910
2911
2912
2913
2914
2915

2916
2917
2918
2919
2920
2921

Table 20: RAG Success Example Step 6-8

Observation 6

The screenshot shows the Drugs.com website with the 'Drug Interaction Checker' page. The search bar at the top contains 'melatonin'. Below the search bar, a list of results is displayed, with the first item being 'melatonin' and the second item being 'Melatonin Time Release (melatonin)'. The page also includes a sidebar with 'Popular tools & resources' such as 'My Med List', 'Drug interaction checker', 'Pill identifier', and 'Compare drugs'.

Action 7

operator_click(144, 504)

Observation 7

The screenshot shows the Drugs.com website with the 'Drug Interaction Checker' page. The search bar at the top is empty. Below the search bar, a list of results is displayed, with the first item being 'melatonin'. The page also includes a sidebar with 'Popular tools & resources' such as 'My Med List', 'Drug interaction checker', 'Pill identifier', and 'Compare drugs'.

Action 8

operator_click(186, 430)

Observation 8

The screenshot shows the Drugs.com website with the 'Drug Interaction Checker' page. The search bar at the top is empty. Below the search bar, a list of results is displayed, with the first item being 'melatonin'. The page also includes a sidebar with 'Popular tools & resources' such as 'My Med List', 'Drug interaction checker', 'Pill identifier', and 'Compare drugs'.

Action 9

operator_type(Folate Forte)

2966
2967
2968
2969

2970
2971
2972
2973
2974
2975

Table 21: RAG Success Example Step 9-11

Observation 9

Drugs.com Know more. Be sure. Search All ▾ Register Sign In

Browse all medications: ABCDEFGHIJKLMNOPQRSTUVWXYZ Advanced Search

Drugs A-Z Pill Identifier Drug Interaction Checker Compare Drugs News Pro Edition More... Help & Support

Drug Interaction Checker

Check interactions with multiple drugs, vaccines, supplements, alcohol, food and diseases.

Drug	Interaction
Folate Forte (multivitamin)	L-Methylfolate Forte (L-methylfolate)
melatonin	

Popular tools & resources

- My Med List
- Drug interaction checker
- Pill identifier
- Compare drugs

Sign in to view your previously saved lists.

Action 10

operator_click(180, 475)

Observation 10

Drugs.com Know more. Be sure. Search All ▾ Register Sign In

Browse all medications: ABCDEFGHIJKLMNOPQRSTUVWXYZ Advanced Search

Drugs A-Z Pill Identifier Drug Interaction Checker Compare Drugs News Pro Edition More... Help & Support

Drug Interaction Checker

Check interactions with multiple drugs, vaccines, supplements, alcohol, food and diseases.

Drug	Interaction
Folate Forte (multivitamin)	L-Methylfolate Forte (L-methylfolate)
melatonin	

Popular tools & resources

- My Med List
- Drug interaction checker
- Pill identifier
- Compare drugs

Action 11

operator_click(185, 665)

Observation 11

Drugs.com Know more. Be sure. Search All ▾ Register Sign In

Browse all medications: ABCDEFGHIJKLMNOPQRSTUVWXYZ Advanced Search

Drugs A-Z Pill Identifier Drug Interaction Checker Compare Drugs News Pro Edition More... Help & Support

Home > Drug Interactions... > Search > Report

Drug Interaction Report

1 potential interaction and/or warning found for the following 2 drugs:

- Folate Forte (multivitamin)
- melatonin

Add another drug

Consumer Professional

Major (0) Moderate (0) Minor (0) Food/lifestyle (1) Therapeutic duplication (0)

How to Prevent Deadly Drug Interactions

Some mixtures of medications can lead to serious and even fatal consequences.

Here are 9 ways to stay safe

Action 12

operator_wait(1000)

3006
3007
3008
3009
3010
3011
3012
3013
3014
3015
3016
3017
3018
3019
3020
3021
3022
3023

3024

3025

3026

3027

3028

Table 22: Non-RAG Failure Example Step 0-2

3029

3030

3031

3032

3033

3034

3035

3036

3037

3038

3039

3040

3041

3042

3043

3044

3045

3046

3047

3048

3049

3050

3051

3052

3053

3054

3055

3056

3057

3058

3059

3060

3061

3062

3063

3064

3065

3066

3067

3068

3069

3070

3071

3072

3073

3074

3075

3076

3077

Instruction: Check drug interaction for melatonin and Folate Forte.**Observation 0**
Action 1

operator_click(819, 406)

Observation 1
Action 2

operator_click(164, 457)

Observation 2
Action 3

operator_type(melatonin)

3078
3079
3080
3081
3082
3083

Table 23: Non-RAG Failure Example Step 3-5

Observation 3
Action 4

operator_click(140, 509)

Observation 4
Action 5

operator_click(162, 436)

Observation 5
Action 6

operator_type(Folate Forte)

3128
3129
3130
3131

3132 Table 24: Non-RAG Failure Example Step 6
31333134 **Observation 6**

3135 **Action 7**3136 `operator.click(228, 482)`

3140 Analysis of the Task 07ec4a12cba8090e2dc524d558ac7675

3141 **Task name:** Check drug interaction for melatonin and Folate Forte.3142 **RAG's Specific Role:**3143 RAG played a crucial role by visually identifying and confirming the correct input fields and
3144 buttons for entering the drug names and initiating the interaction check. It ensured the agent
3145 followed the correct sequence of actions by retrieving relevant visual cues from the user
3146 interface at each step. This was achieved by retrieving screenshots that validated the correct
3147 UI elements were being targeted.3148 **Advantage Identification:**3149 The primary advantages were the accurate identification of UI elements, a reduced risk of
3150 operational errors, and efficient navigation through the task's interface. RAG prevented errors
3151 by ensuring that each click and text input was made on the correct UI element, avoiding
3152 potential misclicks or incorrect data entry. This demonstrated a comprehensive understanding
3153 of the task interface layout, which the NoRAG agent lacked.3154 **Critical Steps Analysis:**

- 3155 • **Step 1: `operator.click(752,576)`:** RAG influenced this step by retrieving a visual
3156 confirmation of the correct button to start the drug interaction check process, ensuring
3157 the task was initiated correctly from the outset.
- 3158 • **Step 5: `operator.click(233,461)`:** RAG ensured that the correct input field for the
3159 first drug, 'melatonin', was selected by retrieving the visual layout of the input form.
3160 This prevented data entry errors by confirming the correct target field.
- 3161 • **Step 11: `operator.click(185,665)`:** RAG confirmed the correct button to submit
3162 the interaction check by retrieving the visual layout of the final confirmation button.
3163 This ensured the task was completed successfully, reducing the risk of an incomplete
3164 execution.

3165 **F.5 QUALITATIVE ANALYSIS ON HARD TASK EXAMPLES**3166 In this subsection, Tables 25–27 present the execution processes of two challenging tasks. We
3167 qualitatively explain why retrieved knowledge brings clear gains on such difficult cases. Due to space
3168 limitations and the large number of steps, we only analyze screenshots from a subset of key steps.

3186

3187

Table 25: Hard Task Success Example 1 (step 1&2)

3188

3189

Task description: Find a condo for rent in Houston, TX, with a monthly rent of no more than 30% of an income of \$8000. The condo should have a minimum area of 600 square feet, and the move-in date is the 1st of next month.

3190

3191

Observation 1

3192

3193

3194

3195

3196

3197

3198

3199

3200

3201

3202

3203

3204

3205

Thought

To proceed with the task, I need to specifically select 'Houston Condos for Rent' from the main navigation to narrow the search to condos in Houston, TX. This matches the example's move to directly filter by city and property type before applying further filters.

Analysis of retrieved knowledge utilization

It determines the high-level execution logic via retrieved knowledge and follows a broad-to-narrow search strategy, first applying filters on geographic location and property type.

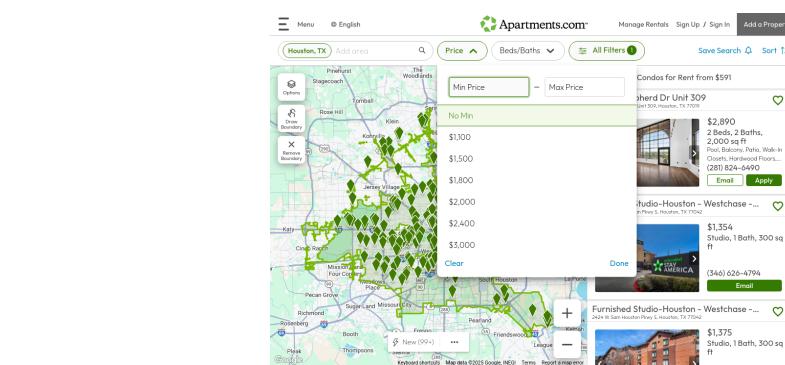
Observation 2

3210

3211

3212

3213



Thought

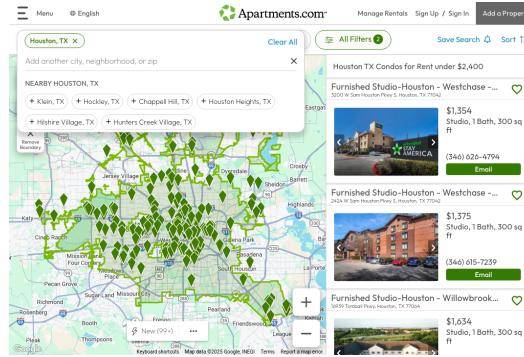
The next step is to set the maximum rent to \$2,400. The accessibility tree shows that under the 'Price' filter, there are input boxes labeled 'Max Price'. Following the example, the typical flow is to click the price filter, enter the cap, and confirm. This aligns with the task's next constraint and the example's approach to applying budget criteria after location and property type.

Analysis of retrieved knowledge utilization

The next high-priority condition, guided by the retrieved knowledge, is the budget. This ensures that the filtering process remains hierarchical and well-structured.

3240
3241
3242
3243
3244
3245
3246
3247
3248
3249
3250
3251
3252
3253

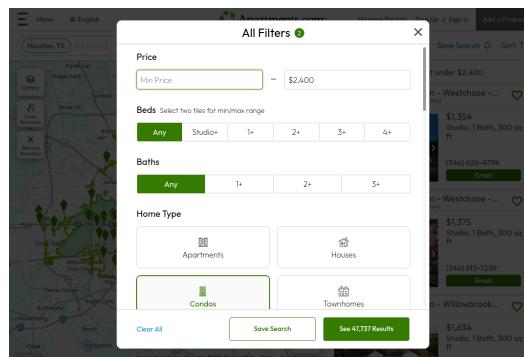
Table 26: Hard Task Success Example 1 (step 19&20)

Observation 1**Thought**

Since all direct attempts to set the minimum area filter in the main and secondary dropdowns did not update the listings (they still show units <600 sq ft), my next step is to reset all filters by clicking the 'Clear All' button, then re-apply every constraint from scratch via the 'All Filters' link. **This systematic reset (as outlined in my plan and corroborated by the RAG example) maximizes the chance of enforcing all constraints, particularly the 600 sq ft minimum, then reviewing listings for a true match.**

Analysis of retrieved knowledge utilization

When the task encounters a failure where the filter stops working partway through, the retrieved knowledge makes it reset all filters instead of continuing trial and error. This enables the model to take decisive initiative in error correction and the recovery stage.

Observation 2**Thought**

After clearing all filters, my next move is to open the 'All Filters' panel. From there, I can comprehensively reapply every necessary constraint: setting the maximum rent to \$2,400, selecting 'Condo', establishing a minimum area of 600 sq ft, and designating the move-in date as the 1st of next month. **This flows directly both from the troubleshooting plan and the similar example, which confirms that a full reset followed by methodical use of the consolidated filters panel is the best recovery when filters are not operating properly.**

Analysis of retrieved knowledge utilization

Retrieved knowledge directs the model to use the integrated filter panel for rescreening instead of applying filters piecemeal.

3294

Table 27: Hard Task Success Example 2 (step 3&7)

3295

3296

Task description: Check the specifications of the best-selling HP FHD laptop with 16 GB RAM and core i7 running on Windows 11.

3297

3298

Observation 1

3299

3300

3301

3302

3303

3304

3305

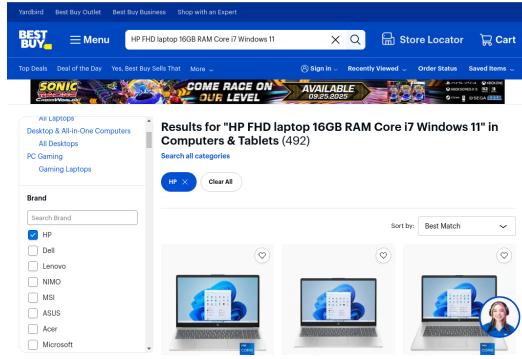
3306

3307

3308

3309

3310



3311

Thought

3312

Now that all relevant filters are applied (HP brand, 16GB RAM, Intel Core i7, Windows), I need to sort the search results by 'Best Selling' to easily identify the best-selling HP FHD laptop that matches all the required criteria. The sort button is usually labeled something like 'Best Match' or similar, and this matches the next step in the example which targets the sort dropdown to select 'Best Selling'.

3318

Analysis of retrieved knowledge utilization

3319

WebRAGent uses the best selling filter based on retrieved knowledge to ensure the answer is correct. In contrast, the non RAG agent skips this filtering step.

3323

Observation 2

3324

3325

3326

3327

3328

3329

3330

3331

3332

3333

3334

3335

Thought

3336

The next step is to set the maximum rent to \$2,400. The accessibility tree shows that under the 'Price' filter, there are input boxes labeled 'Max Price'. Following the example, the typical flow is to click the price filter, enter the cap, and confirm. This aligns with the task's next constraint and the example's approach to applying budget criteria after location and property type.

3344

3345

Analysis of retrieved knowledge utilization

3346

3347

The next high-priority condition, guided by the retrieved knowledge, is the budget. This ensures that the filtering process remains hierarchical and well-structured.

3348

SNIPPET 7: TEXT GUIDANCE

3349

3350

3351

3352

3353

3354

3355

3356

3357

3358

3359

3360

3361

3362

3363

3364

3365

3366

3367

3368

3369

3370

3371

3372

3373

3374

3375

3376

3377

3378

3379

3380

3381

3382

3383

3384

3385

3386

3387

3388

3389

3390

3391

3392

3393

3394

3395

3396

3397

3398

3399

3400

3401

task name

Find discussions of the community and open one with the most replies on flightaware.

website

<https://www.flightradar24.com/>

steps 1

"observation.description": "The image displays the homepage of the FlightAware website as viewed in a web browser. The browser's tab shows the title \"FlightAware - Flight Tracker,\" and the URL in the address bar is \"flightradar24.com.\" The page features some promotional content about global flight tracking and other FlightAware services, including fleet tracking and a section about unlocking real-time flight tracking data. There are images of aircraft (an airplane wing in-flight and a jet on the ground). At the bottom, there is a cookie consent banner with options to \"Allow All\" or adjust \"Cookie Settings.\" There are also navigational links such as \"See all products\" and \"Compare tiers.\"",

"action.description": "The action being taken is launching or opening the FlightAware website by entering its URL (\"<https://www.flightradar24.com/>\") in a web browser. This initiates access to the homepage, preparing the user to start navigating the site to continue with the next steps of the task."

steps 2

"observation.description": "The current screen displays the main page of the FlightAware website. The visible content includes a promotional banner at the top right advertising real-time global flight tracking and ADS-B flight data, along with a call-to-action link to learn more about FlightAware Firehose. There's also a prominent image of a jet on a runway with a headline about secure, private fleet tracking and another call-to-action to compare tiers. At the bottom, there is a visible cookie consent banner with options for \"Allow All\" and \"Cookie Settings.\"",

steps 3

"observation.description": "The image displays the footer section of the FlightAware website. In this section, there are various categories such as \"ABOUT,\" \"COMMUNITY,\" \"PRODUCTS,\" and \"CUSTOMER SERVICE.\" Under the \"COMMUNITY\" category, several links are listed, including \"Squawks,\" \"Discussion,\" \"Photos,\" and \"ADS-B Statistics.\" The \"Discussion\" link is visually highlighted, indicating that it is being hovered over or is the intended target for the next action.",

"action.description": "The action being taken is a user clicking on the \"Discussion\" link under the \"COMMUNITY\" section of the website's footer. This action is intended to navigate to the community discussions area of the FlightAware website, where users can view and participate in forum-style discussions.",

steps 4

"observation.description": "The screenshot displays the \"FlightAware Discussions\" page, which appears to be a forum or discussion board for the FlightAware community. The interface shows several navigation tabs at the top, including \"Categories,\" \"Categories\" (selected), and \"Latest.\" Each category (such as \"FlightAware\" and \"Aviation\") is outlined with subcategories listed below, and to the right, a list of recent discussion threads is visible, showing thread titles, categories, and the number of replies for each (e.g., \"Welcome to the new and improved Discourse!\" with 52 replies, \"Running BananaPI M2 Zero\" with 102 replies, and \"HOWTO: Airspy mini and Airspy R2: Piaware / dump1090-fa configuration\" with 4.5k replies). One specific area of the interface the \"Latest\" tab is visually highlighted with a red box, indicating focus.",

"action.description": "The action being taken is a click operation on the \"Latest\" tab (the tab in the top navigation menu just right of \"Categories\" and specifically marked with a red box). This click will update the view to show the most recent discussion threads in the community, likely ordered by most recent activity instead of by category or another grouping. This will help the user find the most active or current discussions on the FlightAware forum."

steps 5

"observation.description": "The screenshot displays the \"FlightAware Discussions\" page, specifically showing the \"Top\" discussions for the year range May 11, 2024 \u2013 May 11, 2025. The list is organized in a table with columns for Topic, Replies, Views, and Activity. Each row represents a discussion topic, accompanied by its category, participant avatars, number of replies, views, and the date of last activity. A red rectangle highlights the \"Replies\" column header, indicating that it is being interacted with or selected.",

"action.description": "The action being taken is a click on the \"Replies\" column header. This interaction is likely intended to sort the discussion topics by the number of replies, either in ascending or descending order. Sorting by replies will make it easier to identify the discussion with the most replies, which is relevant to the task of selecting the most active or engaged community discussion."

3402

SNIPPET 8: VISUAL GUIDANCE

3403

task name

3405

Find the list of neighborhood maps for Brooklyn on new.mta.info.

3406

website

3407

<https://new.mta.info/>

3408

Action Space

3409

3410

3411

1. click: Clicks on the target element (bounding box).\n2. drag: Drags the target element (bounding box) to a new position.\n3. launchApp: Launches a new URL/app with the specified value (URL).\n4. scroll: Scrolls the page by the specified value (pixels) in the given direction.\nPositions are represented in relative coordinates within the range [0,1] on the observation screenshot.

3412

steps 1

3413

Observation 1: <|image_1|>\n

3414

Action 1: {\"operation\": \"launchApp\", \"value\": \"<https://www.mta.info/>\", \"target\": null}\n\"cand_image_path\": \"[Online-Mind2Web/4091bdd3fa64a5b0d912bc08ea9c824_step_0.png\"]

3415

steps 2

3416

Observation 2: <|image_1|>\n

3417

Action 2: {\"operation\": \"click\", \"value\": null, \"target\": {\"x\": 0.44659, \"y\": 0.0, \"width\": 0.04845, \"height\": 0.084656}},\n\"cand_image_path\": \"[Online-Mind2Web/4091bdd3fa64a5b0d912bc08ea9c824_step_1.png\"]

3418

steps 3

3419

Observation 3: <|image_1|>\n

3420

Action 3: {\"operation\": \"scroll\", \"value\": 1200, \"target\": null},\n\"cand_image_path\": \"[Online-Mind2Web/4091bdd3fa64a5b0d912bc08ea9c824_step_2.png\"]

3421

steps 4

3422

Observation 4: <|image_1|>\n

3423

Action 4: {\"operation\": \"click\", \"value\": null, \"target\": {\"x\": 0.174728, \"y\": 0.809524, \"width\": 0.105969, \"height\": 0.025397}},\n\"cand_image_path\": \"[Online-Mind2Web/4091bdd3fa64a5b0d912bc08ea9c824_step_3.png\"]

3424

steps 5

3425

Observation 5: <|image_1|>\n

3426

Action 5: {\"operation\": \"click\", \"value\": null, \"target\": {\"x\": 0.4036, \"y\": 0.326587, \"width\": 0.059051, \"height\": 0.02328}},\n\"cand_image_path\": \"[Online-Mind2Web/4091bdd3fa64a5b0d912bc08ea9c824_step_4.png\"]

3427

steps 6

3428

Observation 6: <|image_1|>\n

3429

Action 6: {\"operation\": \"click\", \"value\": null, \"target\": {\"x\": 0.174728, \"y\": 0.308598, \"width\": 0.640327, \"height\": 3.518519}},\n\"cand_image_path\": \"[Online-Mind2Web/4091bdd3fa64a5b0d912bc08ea9c824_step_5.png\"]

3430

steps 7

3431

Observation 7: <|image_1|>\n

3432

Action 7: {\"operation\": \"drag\", \"value\": null, \"target\": {\"x\": 0.0, \"y\": 0.084656, \"width\": 0.989782, \"height\": 3.835582}},\n\"cand_image_path\": \"[Online-Mind2Web/4091bdd3fa64a5b0d912bc08ea9c824_step_6.png\"]\",

3433

3434

3435

3436

3437

3438

3439

3440

3441

3442

3443

3444

3445

3446

3447

3448

3449

3450

3451

3452

3453

3454

3455