Branch-and-Browse: Efficient and Controllable Web Exploration with Tree-Structured Reasoning and Action Memory

Shiqi He[1], Yue Cui[2], Xinyu Ma[3], Yaliang Li[2], Bolin Ding[2], Mosharaf Chowdhury[1]
[1]University of Michigan, [2]Alibaba Group, [3]McMaster University
{shiqihe, mosharaf}@umich.edu
{ciwei.cy, yaliang.li, bolin.ding}@alibaba-inc.com
{ma209}@mcmaster.ca

Abstract

Autonomous web agents powered by large language models (LLMs) show strong potential for performing goal-oriented tasks such as information retrieval, report generation, and online transactions. These agents mark a key step toward practical embodied reasoning in open web environments. However, existing approaches remain limited in reasoning depth and efficiency: vanilla linear methods fail at multi-step reasoning and lack effective backtracking, while other search strategies are coarse-grained and computationally costly. We introduce *Branch-and-Browse*, a fine-grained web agent framework that unifies structured reasoning-acting, contextual memory, and efficient execution. It (i) employs explicit subtask management with tree-structured exploration for controllable multi-branch reasoning, (ii) bootstraps exploration through efficient web state replay with background reasoning, and (iii) leverages a page action memory to share explored actions within and across sessions. On the WebArena benchmark, Branch-and-Browse achieves a task success rate of 35.8% and reduces execution time by up to 40.4% relative to state-of-the-art methods. These results demonstrate that Branch-and-Browse is a reliable and efficient framework for LLM-based web agents. Code is available at https://github.com/SymbioticLab/Branch-and-Browse.

1 Introduction

Recent advances in large language models (LLMs) have enabled the development of intelligent agents capable of perceiving webpages, reasoning over content, and acting to automate end-to-end tasks such as online shopping, content management, and issue tracking. LLM-based agents have shown great potential in automating repetitive and programmatic workflows, thereby alleviating human effort in real-world web interaction scenarios (Gao et al., 2024; Wang et al., 2024a; Xi et al., 2025; Yang et al., 2024a). These capabilities stem from LLMs' strong generalization in perception, reasoning, and planning, acquired through large-scale pre-training and instruction tuning (Achiam et al., 2023; Dubey et al., 2024; Hurst et al., 2024; Yang et al., 2025).

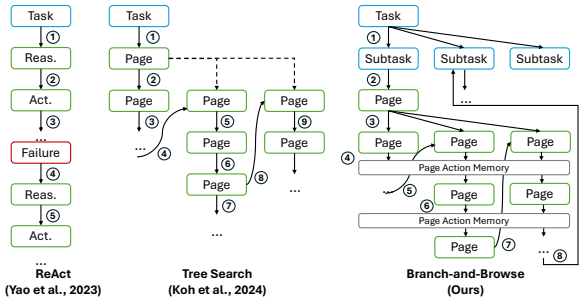

Figure 1: Comparison of web agent strategies. The left shows linear prompting (ReAct) with no backtracking, the middle shows inefficient tree search exploration, and the right illustrates our Branch-and-Browse framework, which enables fine-grained, memory-guided, and efficient multi-branch reasoning.

Unlike static natural language tasks such as summarization or dialogue (Zhang et al., 2025a; Chen et al., 2025; Cui et al., 2025a), web environments pose greater challenges due to their diverse, context-dependent action choices/function calling (e.g., `click`, `type`, `scroll`) (Masterman et al., 2024; Qu et al., 2025; Cui et al., 2025b) and partial observability (Ning et al., 2025; Gao et al., 2025). Solving such tasks requires long-horizon reasoning, explicit exploration, and efficient backtracking across dynamic interfaces. Yet LLM-based agents often fail to align their learned knowledge with task-specific observations and actions, leading to reasoning drift and execution errors. As a result, on benchmarks such as WebArena and VisualWebArena (Zhou et al., 2023; Koh et al., 2024a), current web agents significantly underperform human users, revealing fundamental challenges in integrating reasoning and decision-making within

interactive, real-world web environments.

To cope with these challenges, recent research has explored various strategies that combine reasoning and action generation. Linear prompting-based approaches (e.g., *ReAct*, as illustrated in Figure 1) (Yao et al., 2023) follow a single reasoning–action trajectory, which limits their ability to recover from early mistakes. Once an incorrect click or form submission is made—such as navigating to the wrong product page or misfilling a query field—the agent cannot efficiently backtrack, often restarting the entire sequence. In contrast, search-based methods such as *Tree Search* (Koh et al., 2024b) expand multiple states, thereby improving task completion rates through exploration of multiple pathways. Despite this advantage, these methods suffer from coarse granularity and high computational costs, as they explore each branch independently without effectively utilizing shared contextual information—such as pages previously visited or patterns of failed interactions. These limitations result in redundant exploration and inefficient decision-making processes.

To tackle these limitations, we propose *Branch-and-Browse*, a fine-grained exploration framework that integrates structured reasoning-acting, contextual memory, and efficient execution. Our method provides: (i) *fine-grained structured reasoning* through a subtask manager and tree-structured exploration that enable controllable multi-branch reasoning and principled backtracking; (ii) *web state replay*, which efficiently recover the next branch for exploration, and *background reasoning*, which leverages offline evaluations of unexplored nodes to prune unpromising branches, prioritize actionable steps, and significantly improve the effective branching factor; (iii) a *page action memory* that records explored actions and outcomes, sharing them across branches to reduce redundancy and accelerate decision-making.

We summarize our contributions as follows:

- We propose *Branch-and-Browse*, a novel subtask-aware, tree-structured exploration framework that enables fine-grained, structured reasoning-acting, supporting controllable multi-branch exploration, principled backtracking, and background branch evaluation to enhance reasoning flexibility and exploration efficiency.
- We design a *page action memory* mechanism that addresses contextual fragmentation by maintaining page-level summaries and cross-branch exploration histories. *Background reasoning* mechanism is also introduced to tackle the challenge of exploration efficiency in dynamic branching tasks.
- Our approach attains state-of-the-art performance on WebArena, with a 35.8% task success rate and up to a 40.4% reduction in execution time relative to existing methods, demonstrating its ability to balance reasoning depth and multi-branch exploration efficiency in dynamic web environments.

Table 1: Action space $\mathcal{A}$ for web interaction. Each action $a \in \mathcal{A}$ defines a permissible atomic operation on a web page or browser context. This represents a subset of functions provided by the Playwright MCP toolkit.

Action	Description
`NAVIGATE(url)`	Navigate to a URL.
`NAVIGATE_BACK()`	Move backward.
`NAVIGATE_FORWARD()`	Move forward.
`CLICK(element)`	Click an element.
`TYPE(element, text)`	Type text into a field.
`SELECT(element, option)`	Select an option.
`HOVER(element)`	Hover over an element.
`DRAG(source, target)`	Drag an element.
`PRESS_KEY(key)`	Press a keyboard key.
`TAB_NEW()`	Open a new tab.
`TAB_SELECT(id)`	Switch to a specific tab.
`TAB_CLOSE(id)`	Close a specific tab.

2 Background

Web Agent Environment. Web agents are designed to perform goal-directed tasks through programmatic interaction with real or simulated web interfaces (Ning et al., 2025). Recent implementations leverage large language models (LLMs) as the core reasoning component (Gur et al., 2023; Boisvert et al., 2024; Drouin et al., 2024; Zheng et al., 2024), but these agents form part of a broader system that interfaces with websites through browser automation. Following Zhou et al. (2023), we formalize web agents within a sequential decision-making framework, where a task is specified as a natural language intent i (e.g., "find all shoes under 50 dollars and add them to my wishlist"). At each time step t, the agent observes $o_t \in \mathcal{O}$ and issues an action $a_t \in \mathcal{A}$ according to a policy $\pi(a_t|i, o_t, a_{1:t-1}, o_{1:t-1})$. Executing a_t transitions the environment to a new state $s_{t+1} = Trans(s_t, a_t)$ with an updated observation o_{t+1}. The action space $\mathcal{A}$ includes operations

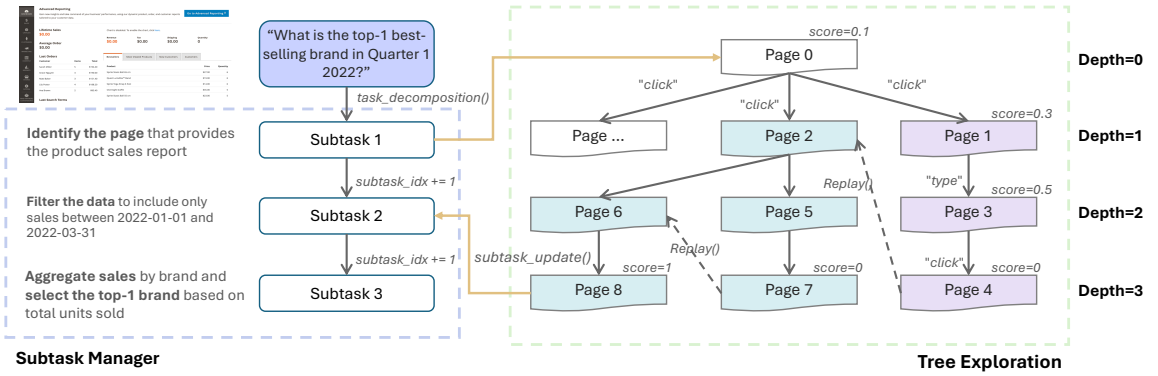

Figure 2: **Pipeline of the Branch-and-Browse framework.** Given an example task (Zhou et al., 2023), the *subtask manager* first decomposes the goal into three subtasks: (1) identify the page providing the product sales report, (2) filter data for the Q1 2022 period, and (3) aggregate brand sales to select the top-1 brand. Each subtask is explored through a tree-structured exploration consisting of iterative `Reason-Act-Evaluation` cycles. Nodes represent visited pages, and edges denote executed system actions (e.g., `click`, `type`). Depth corresponds to exploration steps, while replay links (dashed) enable efficient backtracking and reuse of prior context. As shown, successful branches (e.g., Page 8) trigger `subtask_update()` to advance to the next subgoal, allowing the agent to progress hierarchically through subtasks while avoiding redundant exploration.

analogous to human interactions—element selection (`click`), text input (`type`), URL navigation (`goto`), and tab management—implemented via automation frameworks such as Playwright.[1] The effectiveness of an agent can be assessed through a reward function $r(a_{1:T}, s_{1:T})$ that programmatically verifies whether the final state satisfies the task intent. Table 1 summarizes the action primitives used in our environment.

Reasoning and Action Generation. Recent studies have explored diverse strategies for integrating reasoning and action generation in web agents. Early approaches such as *ReAct* (Yao et al., 2023) follow a single reasoning–action trajectory, lacking robust backtracking, while *Tree Search* (Koh et al., 2024b) improves exploration through branching but remains coarse-grained and computationally costly. Subsequent frameworks like *ScreenAgent* (Niu et al., 2024), *OS-Genesis* (Sun et al., 2024), *WebDreamer* (Gu et al., 2024), and *AgentScope* (Gao et al., 2025) introduce structured subtasks, reflection, and predictive reasoning to enhance planning. However, no unified framework currently integrates these strengths—structured exploration, subtask awareness, and contextual reasoning—into a fine-grained architecture capable of maintaining control while mitigating exploration inefficiency.

Challenges and Opportunities. Despite recent progress, real-world web tasks remain dynamic and partially observable, requiring agents to reason over long horizons and adapt their strategies based on intermediate feedback. Three main challenges arise: (i) *branching uncertainty*, where multiple paths diverge from the goal—handled by Branch-and-Browse through a subtask manager and tree-structured planning for controlled exploration and backtracking; (ii) *exploration inefficiency*, caused by redundant branch expansion—alleviated via efficient web-state replay and background reasoning; and (iii) *contextual fragmentation*, where past interaction histories are lost—addressed by a page action memory that summarizes explored actions and shares context across branches.

3 Design

We formulate long-horizon web interaction as a structured search problem over a dynamic environment. Given a natural language intent i and the corresponding initial observation o_0, the agent explores possible trajectories under a sequential decision process guided by structured planning, subtask-aware control, and page-level memory. A trajectory $\tau = (o_0, a_0, o_1, a_1, \ldots, o_T)$ is generated by alternately selecting actions $a_t \in \mathcal{A}$ according to the policy $\pi(a_t|i, o_t, a_{1:t-1}, o_{1:t-1})$ and applying the environment transition operator T such that $o_{t+1} = Trans(o_t, a_t)$. The objective is to reach a terminal observation o_T that satisfies the task goal $\mathcal{G}(i)$, indicating successful completion of the user intent.

3.1 Fine-Grained Structured Planning

Branch-and-Browse performs structured planning through hierarchical task decomposition and tree-

[1] https://playwright.dev

based exploration. Given a global instruction i, the agent first generates subtasks $\{u_1, \ldots, u_K\}$ via `task_decomposition()`, then explores each through iterative `Reason-Act-Evaluation` cycles. During exploration, active subtasks are adaptively refined using `subtask_update()` based on the current page context and progress (Figure 2).

Subtask Manager. Similar to general agentic tasks (Plaat et al., 2025; Acharya et al., 2025; Xu and Peng, 2025), complex web instructions often require sequential reasoning over intermediate objectives. The *subtask manager* provides structured workflow control by maintaining an active subtask u_k at each iteration. Each subtask is defined by an objective function $\mathcal{O}(u_k)$ and a success predicate $\mathcal{A}(u_k)$ that indicates whether progress toward the goal has been achieved. Initially, the set of subtasks $\{u_1, \ldots, u_K\}$ is generated by

$$\{u_1, \ldots, u_K\} = \texttt{task\_decomposition}(i),$$

where i is the natural language task intent. Since this decomposition occurs before any real page context is available, some subtasks may not align with the actual site structure—e.g., a subtask expecting a "sales report" section on an e-commerce homepage where such content does not exist. To address this, after each round of exploration, the manager performs a contextual update using

$$u_k \leftarrow \texttt{subtask\_update}(u_k, o_t, \tau_{\leq t}),$$

even when no progress is made. This adaptive refinement allows the agent to reinterpret or reformulate the current subtask based on observed evidence—such as replacing an unreachable objective with a semantically related alternative discovered through browsing. If $\mathcal{A}(u_k)$ is satisfied, the subtask is marked as complete and the next subtask u_{k+1} is activated. This dynamic adjustment prevents dead-end reasoning and ensures that exploration remains grounded in the real page context.

Tree Exploration. The exploration process is formulated as a search over a tree $\mathcal{T}$, where each node represents a visited web page (state) and each edge corresponds to a candidate system action $a_t \in \mathcal{A}$. The agent maintains a frontier $\mathcal{F}$ of expandable nodes, each associated with a value estimate guiding prioritization (Koh et al., 2024b). At each iteration, the agent selects the node with the highest utility value v from the frontier:

$$(o, v) \leftarrow \arg \max_{(o', v') \in \mathcal{F}} v'.$$

Each v is given by an LLM-as-a-judge evaluator following the tree-search scoring principle (Koh et al., 2024b), but conditioned on the *active* subtask u_k, the current observation and browsing history, and page action memory so the judge can distinguish redundant retries from new progress. The structured model outputs are mapped to a scalar $v \in [0, 1]$. For each subtask we store the best v seen on any explored prefix and update it when a descendant improves that level, so frontier ordering tracks subtask-local progress rather than a single global score alone. The selected node o is then expanded by generating a set of candidate actions $\{a_i\}_{i=1}^{b}$, where b is the branching factor controlling exploration breadth. Each action yields a successor observation $o^{(i)} = Trans(o, a_i)$, which inherits contextual memory and the current subtask state from its parent.

Exploration proceeds through iterative `Reason-Act-Evaluation` cycles, during which the agent reasons about the next action, executes it, and evaluates the resulting page state. Branches that achieve subtask goals trigger `subtask_update()` to transition to the next subgoal, while low-value or repetitive branches are pruned from the frontier. Replay links (dashed edges in Figure 2) allow fast backtracking and reuse of previously visited contexts, preventing redundant page loads. The process continues until a node satisfies the overall task goal $\mathcal{G}(i)$ or the exploration budget c is consumed. This hierarchical, subtask-aware search structure ensures controllable exploration depth, efficient reuse of prior context, and robust completion of multi-step web tasks.

3.2 Exploration Acceleration

To enhance the efficiency of structured planning, Branch-and-Browse incorporates two complementary acceleration mechanisms: *nearest-URL state replay* and *background reasoning*. These components reduce redundant execution, enable fast recovery from failed trajectories, and improve search efficiency by prioritizing promising branches without exhaustive expansion.

Nearest-URL State Replay. Backtracking is frequently required when exploration paths fail or deviate from the goal. A naive approach of replaying only by navigating directly to a saved URL often misses crucial intermediate actions—such as form filling or tab switching—that are nec-

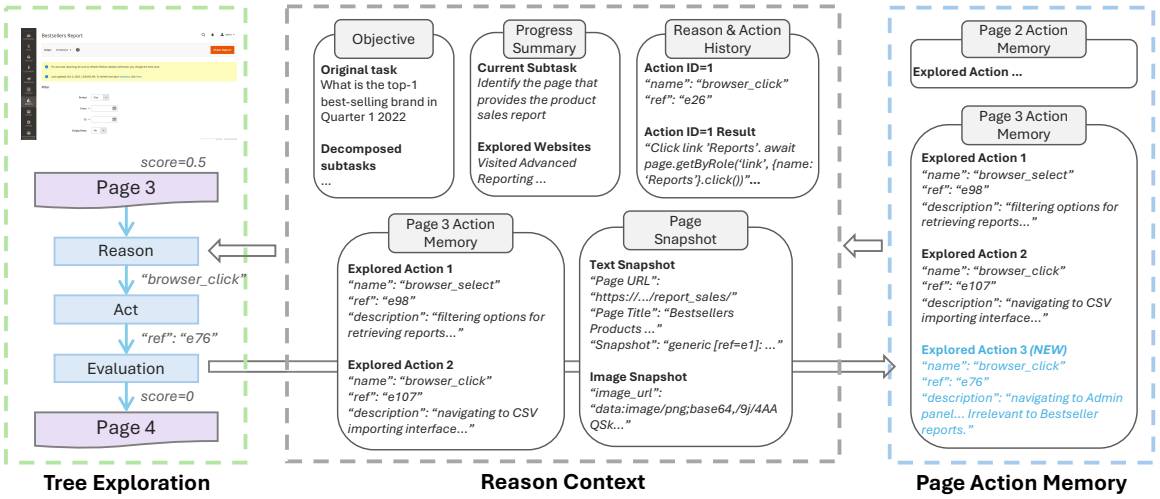

Figure 3: **Example of reason context in branch-and-browse.** During tree exploration, the agent on Page 3 reasons to execute a web interaction (`browser_click`), acts on the referenced element, and evaluates the outcome (score = 0), leading to Page 4. The page-level context records the task objective, progress summary, reason–action history, and page snapshot. The `Page 3 Action Memory` is updated by appending the new action (highlighted in blue) while marking it as irrelevant to the current goal. This cached record guides future reasoning, allowing the agent to avoid redundant exploration and efficiently backtrack during replay.

essary to reconstruct the precise state. Conversely, fully re-executing the entire trajectory $\tau = (o_0, a_0, o_1, \ldots, o_t)$ can be prohibitively time-consuming, especially for long sequences involving multiple page loads. To balance these trade-offs, Branch-and-Browse employs a nearest-URL replay strategy: to revisit an earlier state o_j $(j < t)$, the agent restores the closest cached URL url_c $(c \leq j)$ and replays only the remaining actions $(a_c, \ldots, a_{j-1})$:

$$\textsc{Replay}(\tau, j) = \textsc{Load}(url_c) \xrightarrow{a_c} \ldots \xrightarrow{a_{j-1}} o_j.$$

This hybrid approach preserves essential interaction context while avoiding redundant re-execution, achieving both accuracy and efficiency in backtracking.

Background Reasoning. To further reduce redundant interactions, unexplored frontier nodes $o \in \mathcal{F}$ are evaluated through offline reasoning without active execution. Each node is represented by a context $C(o)$ derived from its DOM snapshot o, and URL. The reasoning model analyzes $C(o)$ to infer plausible next actions and their relevance to the current subtask. When the inferred action corresponds to a `click` operation explicitly linked to a valid URL in the page text, the node is pre-expanded in the background by simulating this transition, i.e., following the referenced URL to create a new state. For all other actions (e.g., `type`, `select`), execution is deferred until the node becomes actively focused in the main exploration branch, since such interactions require a live web context. This selective pre-expansion allows efficient evaluation of deterministic navigation steps, thereby accelerating overall search without compromising correctness.

3.3 Page Action Memory

Global context that logging all agent steps across branches often becomes redundant and entangled, making retrieval inefficient and replay unreliable (Yao et al., 2023; Koh et al., 2024b). To address this, Branch-and-Browse introduces a *page action memory* module that maintains structured reasoning and interaction records at the granularity of each visited page URL. This design allows the agent to retrieve, summarize, and update information efficiently during both online reasoning and replay. As illustrated in Figure 3, each reasoning context consists of the following components:

- **Objective:** Stores both the global task intent and the currently active subtask. This maintains hierarchical reasoning context, enabling consistent goal alignment throughout exploration.

- **Progress Summary:** Provides a concise textual overview of exploration progress—listing visited sites, their relevance to the subtask, and key intermediate findings. These summaries

guide subsequent action generation and serve as inputs for background reasoning.

- **Reason–Action History:** Records the sequence of reasoning and executed system actions (e.g., `click`, `select`, `type`), together with element references and post-action results. Each record maintains `{ActionID, name, ref, result}`, supporting deterministic replay and efficient backtracking during nearest-URL restoration.
- **Page Snapshot:** Captures both textual and visual representations of the page. The text snapshot includes the URL, title, and compressed DOM structure for contextual grounding, while the image snapshot stores a base64-encoded screenshot reference for multimodal reasoning and verification.
- **Action Memory:** Maintains a structured log of all actions attempted on the page, together with metadata and success indicators. Each new action (e.g., `click` $\rightarrow$ "Admin panel") is appended while marking redundant or irrelevant ones as low priority. This evolving cache prevents repeated exploration of unproductive branches and serves as a shared knowledge source for background reasoning and replay.

After each `Reason-Act-Evaluation` cycle, the memory for the current page is updated and serialized into an external cache. When the agent revisits the same URL, the stored summary and cached actions are reloaded to reconstruct local context, enabling consistent reasoning, efficient replay, and branch pruning. Moreover, during subsequent task decomposition, the visited actions recorded in the page memory are incorporated into the reasoning context, allowing the agent to refine subtasks based on explored website structures and improve decomposition accuracy. By structuring memory at the page level and coupling it with system-level command traces, the agent can (i) avoid redundant exploration, (ii) compress historical traces into meaningful summaries, and (iii) recover relevant state rapidly during nearest-URL replay or background reasoning.

4 Evaluation

In this section, we systematically evaluate our approach in a realistic environment for testing autonomous language agents on complex web tasks. Our evaluation is designed to answer the following questions:

- **Q1:** How does Branch-and-Browse perform compared to existing web agent baselines across different website categories and task types?
- **Q2:** What are the individual and combined contributions of the proposed acceleration mechanisms—nearest-URL replay and background reasoning—to efficiency?
- **Q3:** How do key hyperparameters, i.e., depth and branching factor, affect the performance and stability of the framework?

We first present the experimental setup, followed by the main results, ablations, and sensitivity analyses.

4.1 Experimental Setup

Datasets. We evaluate our method on the WebArena (WA) benchmark (Zhou et al., 2023), a realistic web environment designed to assess the performance of autonomous language agents on complex web-based tasks. WebArena bridges the gap between synthetic evaluation settings and real-world scenarios by providing fully functional websites across multiple domains, including an e-commerce website (OneStopShop), GitLab, Reddit, an online store content management system (Shopping Admin), a map, and an English Wikipedia. The benchmark consists of 812 long-horizon tasks instantiated from 241 task templates. We additionally evaluate on VisualWebArena (Koh et al., 2024a), a multimodal dataset built on the same WebArena-style stacks, including images paired with instructions.

Baseline Methods. We divide the baselines into three categories. WebArena (Zhou et al., 2023) and BrowserGym (Drouin et al., 2024) are vanilla ReAct agents that operate within a single reasoning–action loop. Policy-based strategies such as SteP (Sodhi et al., 2023) and AgentOccam-Judge (Yang et al., 2024b) decompose complex web tasks into modular or hierarchical policies for more structured control. In contrast, Tree Search (Koh et al., 2024b) and our Branch-and-Browse agent are search-based strategies that explicitly explore and evaluate multiple reasoning trajectories, orthogonal to policy-based methods. All evaluations are using isolated server instances with standardized browser instrumentation. Agents interact via

Table 2: Comparison of the success rate (SR) of our method with baseline agents on the WebArena (WA) benchmark. Published baselines are shown in the upper block, while tree-search-based methods, our method, and a SteP-augmented hybrid are reported in the lower block.

Method	Model	SR (%)	Shopping	Shopping Admin	GitLab	Map	Reddit	Multisite
WebArena (Zhou et al., 2023)	GPT-4-Turbo	16.5	16.6	15.9	10.0	22.9	21.7	16.7
BrowserGym (Drouin et al., 2024)	GPT-4o	23.5	–	–	–	–	–	–
AutoWebGLM(Lai et al., 2024)	ChatGLM3-6B	18.2	–	–	–	–	–	–
AutoEval (Pan et al., 2024)	GPT-4	26.9	39.6	20.9	25.0	27.5	20.8	16.7
SteP (Sodhi et al., 2023)	GPT-4-Turbo	33.3	33.2	32.4	26.7	35.8	52.8	12.5
AWM (Wang et al., 2024b)	GPT-4	35.5	32.1	29.1	35.0	42.2	54.7	18.8
API Hybrid Agent (Song et al., 2024a)	GPT-4o	38.9	25.7	41.2	44.4	45.9	51.9	16.7
WebPilot (Zhang et al., 2025b)	GPT-4o	37.2	36.9	24.7	39.4	33.9	65.1	*
AgentOccam-Judge (Yang et al., 2024b)	GPT-4-Turbo	45.7	43.3	46.2	38.9	52.3	67.0	16.7
Tree Search (Koh et al., 2024b)	GPT-4o	19.2	27.8	16.5	13.9	26.6	11.3	16.7
Branch-and-Browse (Ours)	GPT-4o	35.8	34.6	26.4	36.7	46.8	50.9	18.8
Branch-and-Browse + SteP (Ours)	GPT-4o	39.8	41.2	32.4	37.8	47.7	54.7	18.8

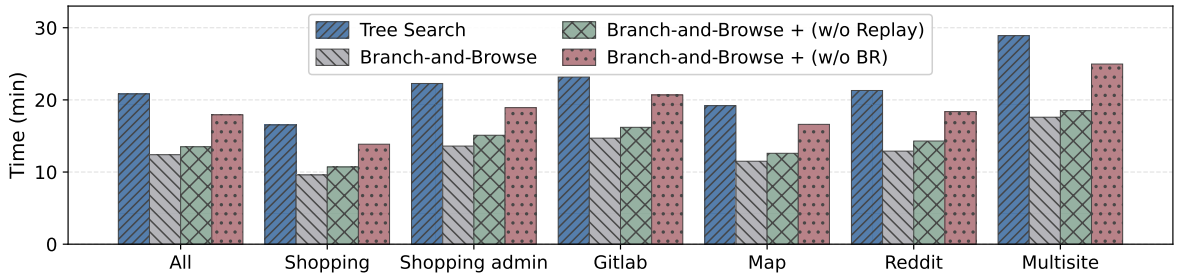

Figure 4: Ablation results on WebArena (812 tasks): average time per successful task across all sites. Failed tasks are excluded since they often fall into repetitive exploration loops, leading to disproportionately high and uninformative time consumption.

natural-language instructions and web-navigation actions, and performance is measured by success rate (SR)—the percentage of tasks achieving the goal state.

Implementation Details. Our system is implemented on top of the Playwright MCP framework,[2] which provides browser-level automation for programmatic web interaction, and is integrated within the `AgentScope` framework (Gao et al., 2025) for unified agent orchestration and reasoning. The backend language model is `gpt-4o-2023-12-12`, serving as the core reasoning and evaluation module throughout task decomposition, action generation, and node scoring. Unless otherwise stated, the tree search uses a depth factor $d = 5$ and branching factor $b = 5$ with an exploration budget of 10 steps per task. All experiments are conducted under identical browser configurations, with page-level memory and background reasoning enabled by default.

4.2 Main Results

Results on WebArena. Table 2 summarizes the performance of Branch-and-Browse compared to the *Tree Search* baseline across 812 tasks. Naive strategies with ReAct exhibit limited success due to their lack of structured exploration and contextual recovery, often failing on long-horizon or dynamically changing pages. Our method achieves substantial gains overall (SR: 35.8% vs. 19.2%), with particularly strong improvements in domains requiring dynamic reasoning and interactive operations. For example, performance increases by 39.6 points on *Reddit* and 22.8 on *GitLab*. These domains demand multi-step exploration and contextual consistency, where fine-grained subtask control and replay mechanisms are most beneficial. In contrast, the gain on *Multisite* tasks is smaller (i.e., 2.1 points), as such tasks often span multiple websites and require longer trajectories, making them inherently more challenging. While several policy-based agents (e.g., SteP, API Hybrid Agent, AgentOccam-Judge) achieve higher absolute scores by leveraging website-specific heuristics or optimized input representations (Sodhi et al., 2023; Song et al., 2024a; Yang et al., 2024b), these methods are orthogonal to—and potentially complementary with—our structured, search-based framework.

[2] https://github.com/microsoft/playwright-mcp

Table 3: Success rate (SR, %) on VisualWebArena (Koh et al., 2024a) by site category and overall.

Method	Classifieds	Reddit	Shopping	Overall
GPT-4o	18.4	17.1	20.0	18.9
Tree Search	26.5	20.5	29.0	26.4
Branch-and-Browse	**37.2**	**25.5**	**33.8**	**32.7**

Combining with SteP. Following prior work that composes web agents with additional policy layers to study how modular controllers interact with environment actions (Yang et al., 2024b), we instantiate a *Branch-and-Browse + SteP* hybrid to demonstrate this complementarity empirically rather than only by analogy. The framework is unchanged except at node expansion: instead of a single ReAct-style policy, the agent invokes SteP (Sodhi et al., 2023), which selects actions through a small library of prompt-based sub-policies (optionally nested on a stack) but ultimately returns one executable web action per expansion, which is executed and recorded as a tree edge; search, replay, background reasoning, pruning, and page action memory follow the default Branch-and-Browse logic. Table 2 shows that this hybrid raises overall SR from 35.8% to 39.8%, with gains on Shopping, Shopping Admin, GitLab, Map, and Reddit and unchanged Multisite performance. The pattern aligns with SteP reducing common interaction mistakes in domains that benefit from structured routines—for instance, map tasks where sub-policies for nearest-place search and directions avoid brittle misuse of generic widgets—while Branch-and-Browse still supplies multi-branch exploration and the acceleration mechanisms absent from policy-only stacks.

Results on VisualWebArena. Beyond text-centric WebArena, we report results on VisualWebArena (Koh et al., 2024a), which keeps the same reproducible, self-hosted sites but shifts the observation interface toward multimodal inputs: agents must ground decisions in rendered pages and visual layout, complementing language-only benchmarks. Table 3 compares a GPT-4o reactive baseline, *Tree Search* (Koh et al., 2024b), and Branch-and-Browse with identical model and budget conventions as our WebArena experiments. Branch-and-Browse improves over *Tree Search* on every reported slice (Classifieds, Reddit, Shopping) and raises overall success rate from 26.4% to 32.7%, indicating that subtask-aware tree exploration with replay and page action memory remains effective when observations emphasize screenshots rather than purely textual state abstractions.

Table 4: Sensitivity of success rate and runtime to search depth and branching factor on WebArena.

Depth d	Branch b	SR (↑)	Time (↑)
0	1	23.9%	8.2 min
1	3	25.3%	8.7 min
	5	31.5%	10.9 min
2	3	30.9%	10.7 min
	5	33.7%	11.6 min
3	5	34.4%	11.9 min
5	5	35.8%	12.4 min

4.3 Ablation Studies

To analyze the contribution of each efficiency mechanism, we conduct ablation studies on the *Replay* and *Background Reasoning* modules. Figure 4 compares the average completion time per task against the *Tree Search* baseline across all domains. For a fair comparison, we report the average time only on *successful* tasks, as failed trajectories often enter repetitive loops and inflate runtime without contributing meaningful progress.

Overall, Branch-and-Browse substantially reduces execution time compared to *Tree Search*, lowering the average task time from 20.8 to 12.4 minutes (a 40.4% reduction). When *Replay* is disabled, the average time increases slightly, showing that nearest-URL replay helps avoid redundant re-navigation but contributes less to total efficiency gains. In contrast, removing *Background Reasoning* leads to a larger increase in task time, since offline inference overlaps with active exploration and prunes unpromising branches early. This difference is expected, as reasoning typically dominates runtime compared to replay operations. Together, these results confirm that both mechanisms are effective, with background reasoning providing the greater contribution to overall efficiency.

4.4 Sensitivity Studies

We analyze the sensitivity of Branch-and-Browse to search hyperparameters, including the tree depth (d) and branching factor (b). All experiments are conducted under the same overall search budget as the default setting (equivalent to the number of `ReAct` steps performed by the baseline), ensuring a fair comparison in total interaction count and model queries.

As shown in Table 4, when both $d = 0$ and $b = 1$, the framework degenerates to a linear ex-

ecution mode identical to `ReAct` augmented with the page action memory. This configuration still benefits from context reuse but lacks acceleration from structured exploration, resulting in a success rate of 23.9%. As we increase either the branching factor or the search depth, performance improves significantly while runtime grows moderately. For instance, expanding the search to $b = 5$ and $d = 2$ yields a success rate of 33.7% with only a 3.4-minute increase in average task time. The gains are largely attributed to the efficiency mechanisms—nearest-URL replay and background reasoning—that enable deeper and broader exploration without linear growth in cost. Overall, Branch-and-Browse demonstrates strong robustness: larger search budgets consistently improve success rates, while execution time remains well within practical limits, confirming the scalability and efficiency of the framework.

5 Related Work

Perception and Web Environmental Understanding: Modern web agents leverage diverse modalities for environmental perception. Text-based methods (Deng et al., 2023; Koh et al., 2024b; Gur et al., 2023; Ma et al., 2023) utilize HTML metadata and accessibility trees, such as the approach in (Koh et al., 2024b), which takes accessibility trees of web pages as input. Screenshot-based techniques incorporate vision-language models to interpret graphical user interfaces (GUIs) (Singh et al., 2025; Cheng et al., 2024; Wu et al., 2024; Sun et al., 2024; Zhang and Zhang, 2023; Kil et al., 2024). For example, SeeClick (Cheng et al., 2024) exclusively uses screenshots as observations and enhances grounding through targeted pre-training. Multi-modal approaches (Song et al., 2024b; He et al., 2024) integrate the complementary capabilities of various modalities. Notable examples include MMAC-Copilot (Song et al., 2024b), which integrates GPT-4V (OpenAI, 2023) for visual content, and Gemini Vision (Li et al., 2024), which processes video data—significantly improving multi-modal data handling.

Memory and Knowledge Integration. Advanced web agents utilize both short-term and long-term memory systems to enhance decision-making and task performance. For instance, AutoWebGLM (Lai et al., 2024) models web browsing as a sequential decision-making process, where actions are determined based on the current state and an integrated history of previous webpage states and actions. Similarly, Agent S (Agashe et al., 2024) combines external online web searches for supplementary knowledge with an internal narrative memory to draw on task-specific experiences. This includes generating sub-tasks derived from summaries of both successful and unsuccessful trajectories.

6 Conclusion

We introduced Branch-and-Browse, a fine-grained framework for efficient and reliable LLM-based web agents. By combining subtask-aware tree exploration, page action memory, and acceleration mechanisms, it enables structured and efficient reasoning. On the WebArena benchmark, Branch-and-Browse achieves a 35.8% success rate and reduces execution time by 40.4%, demonstrating a scalable path toward practical, high-performing web agents.

Limitations

While Branch-and-Browse improves the efficiency and reliability of web agents through structured exploration, it still operates within a single-browser session setting and does not parallelize exploration across multiple branches. This limits scalability in highly complex or time-sensitive tasks where concurrent reasoning could further accelerate progress. Additionally, our framework primarily focuses on search-based control and has not yet been fully integrated with policy-based approaches, which may offer complementary benefits. Future work will explore multi-session and hybrid policy–search integration to enhance both scalability and adaptability in dynamic web environments.

Ethical Considerations

Branch-and-Browse aims to enhance autonomous web exploration in a controlled and responsible manner. All experiments were conducted in simulated environments (i.e., WebArena (Zhou et al., 2023)) to avoid real-world data access or unintended interactions. However, when deployed on live websites, automated agents may raise concerns related to privacy, consent, and site policies. We recommend incorporating safeguards such as rate limiting, permission control, and transparent disclosure to ensure ethical and compliant use. For this paper, large language models were used solely for text polishing and formatting assistance, not for idea generation or substantive content creation.

Acknowledgements

We thank the reviewers, as well as members of SymbioticLab, for their helpful feedback. This work was supported in part by NSF grants CCF-2450085 and CNS-2106184, and by grants from Ford and Cisco.

References

Deepak Bhaskar Acharya, Karthigeyan Kuppan, and B Divya. 2025. Agentic ai: Autonomous intelligence for complex goals–a comprehensive survey. *IEEE Access*.

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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

Léo Boisvert, Megh Thakkar, Maxime Gasse, Massimo Caccia, Thibault de Chezelles, Quentin Cappart, Nicolas Chapados, Alexandre Lacoste, and Alexandre Drouin. 2024. Workarena++: Towards compositional planning and reasoning-based common knowledge work tasks. *Advances in Neural Information Processing Systems*, 37:5996–6051.

Chaoran Chen, Bingsheng Yao, Ruishi Zou, Wenyue Hua, Weimin Lyu, Yanfang Ye, Toby Jia-Jun Li, and Dakuo Wang. 2025. Towards a design guideline for rpa evaluation: A survey of large language model-based role-playing agents. *arXiv preprint arXiv:2502.13012*.

Kanzhi Cheng, Qiushi Sun, Yougang Chu, Fangzhi Xu, Li YanTao, Jianbing Zhang, and Zhiyong Wu. 2024. Seeclick: Harnessing gui grounding for advanced visual gui agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9313–9332.

Yue Cui, Liuyi Yao, Zitao Li, Yaliang Li, Bolin Ding, and Xiaofang Zhou. 2025a. Efficient leave-one-out approximation in llm multi-agent debate based on introspection. *arXiv preprint arXiv:2505.22192*.

Yue Cui, Liuyi Yao, Shuchang Tao, Weijie Shi, Yaliang Li, Bolin Ding, and Xiaofang Zhou. 2025b. Enhancing tool learning in large language models with hierarchical error checklists. *arXiv preprint arXiv:2506.00042*.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. Mind2web: Towards a generalist agent for the web. *Advances in Neural Information Processing Systems*, 36:28091–28114.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H Laradji, Manuel Del Verme, Tom Marty, Léo Boisvert, Megh Thakkar, Quentin Cappart, David Vazquez, and 1 others. 2024. Workarena: How capable are web agents at solving common knowledge work tasks? *arXiv preprint arXiv:2403.07718*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Chen Gao, Xiaochong Lan, Nian Li, Yuan Yuan, Jingtao Ding, Zhilun Zhou, Fengli Xu, and Yong Li. 2024. Large language models empowered agent-based modeling and simulation: A survey and perspectives. *Humanities and Social Sciences Communications*, 11(1):1–24.

Dawei Gao, Zitao Li, Yuexiang Xie, Weirui Kuang, Liuyi Yao, Bingchen Qian, Zhijian Ma, Yue Cui, Haohao Luo, Shen Li, and 1 others. 2025. Agentscope 1.0: A developer-centric framework for building agentic applications. *arXiv preprint arXiv:2508.16279*.

Yu Gu, Kai Zhang, Yuting Ning, Boyuan Zheng, Boyu Gou, Tianci Xue, Cheng Chang, Sanjari Srivastava, Yanan Xie, Peng Qi, and 1 others. 2024. Is your llm secretly a world model of the internet? model-based planning for web agents. *arXiv preprint arXiv:2411.06559*.

Izzeddin Gur, Hiroki Furuta, Austin Huang, Mustafa Safdari, Yutaka Matsuo, Douglas Eck, and Aleksandra Faust. 2023. A real-world webagent with planning, long context understanding, and program synthesis. *arXiv preprint arXiv:2307.12856*.

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

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, and 1 others. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

Jihyung Kil, Chan Hee Song, Boyuan Zheng, Xiang Deng, Yu Su, and Wei-Lun Chao. 2024. Dual-view visual contextualization for web navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14445–14454.

Jing Yu Koh, Robert Lo, Lawrence Jang, Vikram Duvvur, Ming Lim, Po-Yu Huang, Graham Neubig, Shuyan Zhou, Russ Salakhutdinov, and Daniel Fried. 2024a. Visualwebarena: Evaluating multimodal agents on realistic visual web tasks. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*.

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

Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, and 1 others. 2024. Autowebglm: A large language model-based web navigating agent. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5295–5306.

Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. 2024. Mini-gemini: Mining the potential of multi-modality vision language models. *arXiv preprint arXiv:2403.18814*.

Kaixin Ma, Hongming Zhang, Hongwei Wang, Xiaoman Pan, and Dong Yu. 2023. Laser: Llm agent with state-space exploration for web navigation. In *NeurIPS 2023 Foundation Models for Decision Making Workshop*.

Tula Masterman, Sandi Besen, Mason Sawtell, and Alex Chao. 2024. The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey. *arXiv preprint arXiv:2404.11584*.

Liangbo Ning, Ziran Liang, Zhuohang Jiang, Haohao Qu, Yujuan Ding, Wenqi Fan, Xiao-yong Wei, Shanru Lin, Hui Liu, Philip S Yu, and 1 others. 2025. A survey of webagents: Towards next-generation ai agents for web automation with large foundation models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6140–6150.

Runliang Niu, Jindong Li, Shiqi Wang, Yali Fu, Xiyu Hu, Xueyuan Leng, He Kong, Yi Chang, and Qi Wang. 2024. Screenagent: A vision language model-driven computer control agent. *arXiv preprint arXiv:2402.07945*.

OpenAI. 2023. GPT-4V(ision) System Card. https://openai.com/index/gpt-4v-system-card/. Accessed: 2025-09-28.

Jiayi Pan, Yichi Zhang, Nicholas Tomlin, Yifei Zhou, Sergey Levine, and Alane Suhr. 2024. Autonomous evaluation and refinement of digital agents. *arXiv preprint arXiv:2404.06474*.

Aske Plaat, Max van Duijn, Niki van Stein, Mike Preuss, Peter van der Putten, and Kees Joost Batenburg. 2025. Agentic large language models, a survey. *arXiv preprint arXiv:2503.23037*.

Changle Qu, Sunhao Dai, Xiaochi Wei, Hengyi Cai, Shuaiqiang Wang, Dawei Yin, Jun Xu, and Ji-Rong Wen. 2025. Tool learning with large language models: A survey. *Frontiers of Computer Science*, 19(8):198343.

Kunal Singh, Shreyas Singh, and Mukund Khanna. 2025. Trishul: Towards region identification and screen hierarchy understanding for large vlm based gui agents. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 170–179.

Paloma Sodhi, SRK Branavan, Yoav Artzi, and Ryan McDonald. 2023. Step: Stacked llm policies for web actions. *arXiv preprint arXiv:2310.03720*.

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

Zirui Song, Yaohang Li, Meng Fang, Yanda Li, Zhenhao Chen, Zecheng Shi, Yuan Huang, Xiuying Chen, and Ling Chen. 2024b. Mmac-copilot: Multi-modal agent collaboration operating copilot. *arXiv preprint arXiv:2404.18074*.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, and 1 others. 2024. Os-genesis: Automating gui agent trajectory construction via reverse task synthesis. *arXiv preprint arXiv:2412.19723*.

Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, and 1 others. 2024a. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18(6):186345.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. 2024b. Agent workflow memory. *arXiv preprint arXiv:2409.07429*.

Zhiyong Wu, Zhenyu Wu, Fangzhi Xu, Yian Wang, Qiushi Sun, Chengyou Jia, Kanzhi Cheng, Zichen Ding, Liheng Chen, Paul Pu Liang, and 1 others. 2024. Os-atlas: A foundation action model for generalist gui agents. *arXiv preprint arXiv:2410.23218*.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, and 1 others. 2025. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101.

Renjun Xu and Jingwen Peng. 2025. A comprehensive survey of deep research: Systems, methodologies, and applications. *arXiv preprint arXiv:2506.12594*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Ke Yang, Jiateng Liu, John Wu, Chaoqi Yang, Yi R Fung, Sha Li, Zixuan Huang, Xu Cao, Xingyao Wang, Yiquan Wang, and 1 others. 2024a. If llm is the wizard, then code is the wand: A survey on how code empowers large language models to serve as intelligent agents. *arXiv preprint arXiv:2401.00812*.

Ke Yang, Yao Liu, Sapana Chaudhary, Rasool Fakoor, Pratik Chaudhari, George Karypis, and Huzefa Rangwala. 2024b. Agentoccam: A simple yet strong baseline for llm-based web agents. *arXiv preprint arXiv:2410.13825*.

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

Haopeng Zhang, Philip S Yu, and Jiawei Zhang. 2025a. A systematic survey of text summarization: From statistical methods to large language models. *ACM Computing Surveys*, 57(11):1–41.

Yao Zhang, Zijian Ma, Yunpu Ma, Zhen Han, Yu Wu, and Volker Tresp. 2025b. Webpilot: A versatile and autonomous multi-agent system for web task execution with strategic exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Zhuosheng Zhang and Aston Zhang. 2023. You only look at screens: Multimodal chain-of-action agents. *arXiv preprint arXiv:2309.11436*.

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

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, and 1 others. 2023. Webarena: A realistic web environment for building autonomous agents. *arXiv preprint arXiv:2307.13854*.