
Thinking vs. Doing: Improving Agent Reasoning by Scaling Test-Time Interaction

Junhong Shen^{1,2*} Hao Bai^{3*} Lunjun Zhang⁴ Yifei Zhou⁵ Amrith Setlur¹

Shengbang Tong⁷ Diego Caples⁶ Nan Jiang³ Tong Zhang³

Ameet Talwalkar¹ Aviral Kumar¹

¹CMU ²Scribe ³UIUC ⁴U of Tronoto ⁵UC Berkeley ⁶The AGI Company ⁷NYU

Abstract

The current paradigm of test-time scaling relies on generating long reasoning traces (“thinking” more) before producing a response. In agent problems that require interaction, this can be done by generating thinking traces before acting in the world. However, this process does not allow agents to acquire new information from the environment or adapt their behavior over time. In this work, we propose to scale **test-time interaction**, an untapped dimension of test-time scaling that increases the agent’s interaction horizon to enable running rich behaviors such as exploration, backtracking, and dynamic re-planning within a single rollout. To demonstrate the promise of this scaling dimension, we study the domain of web agents. We first show that even prompting-based interaction scaling without any training can improve task success on web benchmarks non-trivially. Building on this, we introduce **TTI** (Test-Time Interaction), a curriculum-based online reinforcement learning (RL) approach that trains agents by adaptively adjusting their rollout lengths. Using a Gemma 3 12B model, **TTI** produces state-of-the-art open-source, open-data web agents on WebVoyager and WebArena benchmarks. We further show that **TTI** enables agents to balance exploration and exploitation adaptively. Our results establish interaction scaling as a powerful, complementary axis to scaling per-step compute, offering new avenues for training adaptive agents.

Project page: <https://test-time-interaction.github.io>

Code: <https://github.com/test-time-interaction/TTI>

1 Introduction

Recent advances in foundation models have enabled a shift from static language models to interactive agents that perform multi-step tasks in dynamic environments like browsers [1–6], terminals [7], and the physical world [8–13]. These agents operate in closed-loop settings where each action changes the current state of the world and affects future interaction with the environment. As a result, interactive agents must plan under uncertainty and adapt to failures in real time to be successful. How can we build agents that succeed in such interactive settings?

Current post-training approaches produce *reactive* agents that respond to immediate observations but struggle with evolving or uncertain task dynamics. Methods like supervised fine-tuning (SFT) on expert demonstrations [14–18] or reinforcement learning (RL) with task rewards [19–23] typically train agents to predict a *single best action* at each step. Even with test-time scaling, where agents

*Equal contribution. Correspondence to: junhongs@andrew.cmu.edu, haob2@illinois.edu

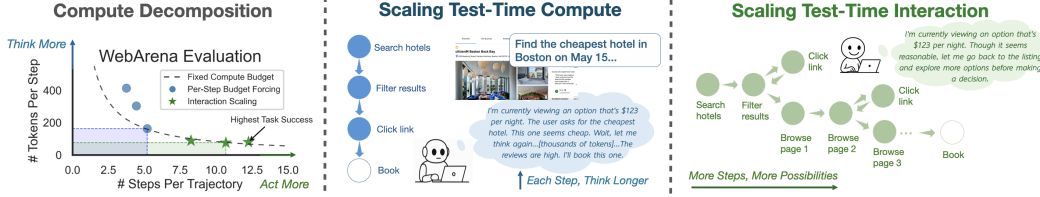


Figure 1: We propose a new-axis of test-time scaling for agents: **scaling the number of interaction steps**. Unlike traditional methods that emphasize longer reasoning per step, we show that acting more helps gain new information from the environment and improve task performance (detailed results of the left plot in Section 4.2).

are prompted to “think” longer before prescribing an action [24–26], they are still optimized to select the most effective action based on the agent’s internal state. While sufficient for fully observable and stationary tasks, reactive policies based on the agent’s internal estimate of the task state are often sub-optimal in partially observable (e.g., incomplete details visible on a page) or non-stationary (e.g., fluctuating prices during flight booking) settings, where adaptive, information-seeking behavior is critical.

In this paper, we argue that instead of reactive “optimal” policies, agents should learn adaptive policies that can collect new information from the environment and adjust their behaviors on-the-fly. A pre-requisite for such adaptability is the ability to *take more actions* during deployment than those prescribed by an expert trajectory. We therefore propose a new dimension of test-time scaling: **increasing the number of interaction steps of the agent**. This allows agents to have sufficient context and time to attempt different behaviors. For example, in a hotel booking task, an agent must first browse many listings to compare user reviews and check availability before selecting the best option. Interaction scaling is orthogonal to existing methods based on chain-of-thought (CoT), which emphasize deeper reasoning per step but do not support information-gathering from the environment. This notion of information gain is unique to agentic tasks with partial observability and requires interaction, not merely larger per-step compute. For instance, an agent that reasons deeply about one selected hotel without interacting further may miss better options that show up only after exploration.

Although the idea of interaction scaling is conceptually straightforward, extending it to post-training and teaching agents to scale interaction autonomously presents key challenges. Without appropriate training signals, agents may overfit to exploratory behaviors like blindly clicking links but not making progress toward the actual task objective, wasting the additional steps. To tackle this issue, we propose to combine online RL with a curriculum that prescribes how to scale the interaction horizon, training agents that first learn effective exploitation before extending their horizon to explore.

We instantiate our approach in the domain of web agents, a widely applicable setting with well-established benchmarks. We first show that scaling test-time interaction via prompting the agent to “think and act again” after it decides to terminate can already improve the task success rate from 23% to $\geq 28\%$ on WebArena [2] (see Figure 3 for details). While this increases trajectory length and the number of tokens generated, spending an equivalent amount of compute on conventional test-time scaling methods like forcing the agent to think for longer [27] or running best-of- n [28–30] yields less than a 3% gain. These findings validate the potential of interaction scaling.

We then move beyond prompting and develop **TTI** (Test-Time Interaction), a curriculum-based RL approach that trains agents to adaptively scale interaction by gradually increasing the rollout horizon. We scale **TTI** to large and diverse training sets ($>100K$ tasks across ~ 20 domains) by integrating an automated task generation pipeline [19]. **TTI** achieves state-of-the-art performance among open-source agents trained on open data on both WebVoyager [1] and WebArena [2], using only a 12B Gemma 3 model, improving over the non-fine-tuned agent by 9% and 8%, respectively. Our analysis further shows that curriculum training enables adaptive exploration: agents learn to initiate new searches or backtrack in complex tasks, while following efficient paths in simpler ones.

In summary, we introduce interaction scaling as a new dimension for test-time scaling of agents. We propose **TTI** to train agents by adjusting interaction horizon dynamically. Our results show that **TTI** yields strong empirical gains and offers promising directions for domains beyond web navigation.

2 Related Work

Scaffolded foundation models as web agents. Prior works use external control structures to scaffold foundation models via modular prompting [31, 6, 32–37], programs [38? –41], or feedback

mechanisms [42–45]. These methods often rely on proprietary models like GPT-4 [46] or Claude [47]. Thus, progress is driven by designing better prompts and workflows for planning [48–51], self-correction [52], self-evaluation [53, 54], or by integrating external modules such as memory [55] or retrieval systems [56]. More recently, developing specialized agents has become a promising direction. ScribeAgent [14] first uses real-world data to demonstrate that simple fine-tuning can outperform most prompt-based agents. Prior work also builds automated data curation workflows [57–59] and distillation methods [18]. Despite these efforts, scaffolding methods do not enable agents to improve through interaction, and rely on fixed wrappers that lack adaptability across diverse environments.

RL training for foundation model agents. RL-based approaches enable agents to autonomously improve through interaction. Prior work has explored DPO [60], actor-critic [20, 21, 61], or distributed sampling [62]. Pipelines like PAE [19] and Learn-By-Interact [63] support automatic task generation, exploration, and labeling. However, most of these approaches lack mechanisms for test-time exploration, limiting the agent’s ability to adapt its behavior over long horizons, especially under partially observable conditions. Our work addresses this limitation by scaling test-time interaction as an independent dimension, allowing agents to refine behavior while acting. Curriculum-based RL has been applied in AutoWebGLM [64] and WebRL [23], where curricula are based on estimated task difficulty derived from the complexity of LLM-generated instructions. While our approach also induces a progression from simpler to more complex behaviors, it does so by gradually increasing the allowed interaction horizon rather than relying on explicit measures of task difficulty.

Scaling test-time compute. Increasing test-time compute via best-of- n sampling [29], beam search [65, 66], or verifiers [67–69] has shown to improve performance in reasoning-heavy tasks. In non-interactive settings like math and competitive coding, recent methods train models to generate long CoT and scale reasoning internally [e.g., 70, 27, 71?]. As for multi-turn interactive settings, most existing works simply integrate CoT prompting into the agent system to enhance per-step reasoning [e.g., 72, 51]. EXACT [73] scales up the search process for each action, GenRM-CoT [74] the number of verifiers, and Jin et al. [75] the number of agents. However, none of these efforts studies the benefits of scaling over the time horizon, where the agent can explore alternatives, backtrack, or gather more information before committing to certain actions. Our work extends this line of research by introducing test-time scaling of interaction. As we will show in our empirical results (Section 4.2), the benefits of scaling test-time interaction go beyond test-time scaling (or “reasoning”) within a given time step, because each extra step of interaction with the environment provides new information to the policy, whereas thinking for longer simply reorganizes information that the agent already has.

3 Problem Setup

We consider solving a web task as a finite-horizon sequential decision-making process guided by a reward objective². Formally, the environment implements a transition function that evolves over time and provides an observation o_t at step t reflecting the current task state. The agent policy π is parameterized by a multi-modal foundation model that maps observation history $o_{1:t-1}$ and action history $a_{1:t-1}$ to the next action a_t . These histories allow the policy to represent rich, context-dependent behaviors enabled via interaction. We denote the environment horizon, or the maximum number of interaction steps allowed in the environment, as h . For each task, the actual interaction process ends when the agent issues a stop signal or reaches the step limit h . Let $h_{\text{stop}} \in (0, h]$ denote the actual number of steps taken. The agent receives a reward of 1 for task success, and 0 otherwise.

Observation space design. Following [14, 19, 2], we consider an observation space consisting of the task goal, the URL, and a structured representation of the current web page that includes both the accessibility tree of the web page and a screenshot augmented with a set-of-marks overlay [1], which assigns a unique identifier to each element that the agent can interact with. While the agent has access to all past observations in principle, doing so quickly exhausts the context window in practice, so we truncate observation history to the most recent three steps, similar to prior works [19]. However, the agent still has access to all of its past actions. We show in Appendix B an example observation.

Action space parameterization. We adopt a discrete action space with six actions: click, type, scroll, go back, search (e.g., Google or Bing), and stop the task with an answer. Following Yang et al. [33], we do not consider compound actions like `goto[url]`, as complex action spaces can hinder performance. For a detailed definition of each action, see the agent prompt in Appendix C.1.

²While this work centers on web agents, we believe the insights should generalize to other agent problem domains, and we hope future work will extend these ideas beyond web agents and web navigation.

4 Scaling Test-Time Interaction: A New Dimension of Agent Scaling

Prior methods for LLM test-time compute scaling usually scale the number of thinking tokens at each step [76? ? ?], but this does not enable an agent to engage in longer interactions with the environment to collect new information. In principle, scaling the maximum number of interaction steps should allow the agent to employ richer behavioral strategies such as re-attempts, backtracking, and recovery. We will now verify this hypothesis via controlled experiments on WebArena [2]. We will then build upon these insights to develop *TTI*, which trains agents to optimize test-time interactions.

Controlled experiment setup. We choose WebArena [2] as our testbed, primarily because it enables reproducible interaction with diverse domains (OneStopShop, Reddit, GitLab, CMS, and OpenStreetMap) and is equipped with ground truth evaluators. We randomly sample 62 tasks for testing and reserve the remaining 750 for online training (see Section 5.1). To ensure sufficient interaction, we set a generous test-time limit of $h = 30$, which is well above the average length of around 6 steps required by most tasks [55, 77]. Note that experiments in this section are for analysis rather than benchmarking purposes. We will show more experimental results on multiple benchmarks in Section 6.

To study the effect of increasing h , we use a simple prompting-based agent with Gemma 3 12B [78] base model, which observes the web page and outputs an action via a *single* model call. It does not leverage any retrieval, verifiers, or other external modules, ensuring any performance gains come solely from increased h but not auxiliary scaffolding. We also study whether it is beneficial to prompt the agent to generate a reasoning trace before acting (see Appendix C.1 for the templates). As Table 1 shows, CoT prompting yields significantly higher task success rate (SR) than direct action generation, setting a baseline of 23.81% on the test tasks. We thus adopt a simple chain-of-thought (CoT) approach as the default prompting strategy.

Table 1: **Results averaged over three runs.**

Prompt	Task SR (%)
Action Only	14.76
CoT	23.81

4.1 Scaling Test-Time Interaction by Acting Longer

To study the impact of interaction scaling, we introduce a purely inference-time “check-again” mechanism: after the agent issues task completion, we explicitly prompt it to reconsider its decision by “You just signaled task completion. Let’s pause and think again...” We can extend re-checking to multiple checks. Detailed prompts are in Appendix C.4.

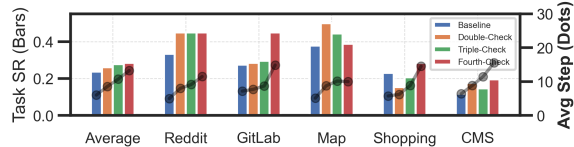


Figure 2: **Scaling test-time interaction by prompting the agent to “re-check” its answer.**

As shown in Figure 2, prompting the agent to re-check not only increases the interaction length h_{stop} (line plots), but also improves the success rates on most domains (bar plots). When being asked to “check-again”, the agent either reaffirms its decision (e.g., “I previously stated the driving time was approximately 30 minutes....30 minutes seems plausible with typical traffic conditions. I’ll stick with my previous answer.”) or revises it upon reflection (e.g., “My apologies. Although the address book *displays* the desired address, the task requires me to *change* it....I should click button [24] to modify the address.”). In particular, it changes its action $\sim 25\%$ of the time after double-checking. This highlights the potential of interaction scaling: when given sufficient time, the agent is likely to explore alternatives before reaching an answer. The chances of the answer being correct could thus be higher.

However, we do observe that repeatedly prompting the agent to re-check can sometimes lead to confusion or hallucination, causing it to revise correct answers into incorrect ones. This may explain the performance drop observed in domains like Map. This is perhaps an inevitable limitation of scaling test-time interaction via *prompting* alone, akin to how prompting is not an effective way to even scale per-step test-time compute (see self-correction results for prompted models in ?). We discuss this limitation further in Section 4.2 and address by *training* the agents explicitly.

4.2 Scaling Test-Time Interaction vs. Per-Step Test-Time Compute

Next, we examine the effect of scaling interaction compared to scaling per-step reasoning: Given a total token budget, should agents prioritize more interaction steps or generating longer reasoning traces at each step? To explore this, we study two conventional test-time compute scaling methods.

Per-step budget forcing. Following Muennighoff et al. [27], we prompt the agent to “wait and think again” after generating an initial CoT, encouraging more intermediate reasoning before it commits to an action. We vary the number of forced waits from 1 to 4. Despite trying various prompts (see Appendix C.4), our agent changes its actions only 15% of the time, and most of these changes involve reordering subtasks rather than exploring alternative paths or error correction.

Per-step best-of- n . At each step, we sample $n \in \{3, 5, 7\}$ candidate actions and select one via majority voting, similar to [14]. We did not scale n up further because best-of- n is compute-intensive for multi-step settings ($n \cdot h$ more expensive than the baseline per rollout), and we observe diminishing returns from sampling more actions, despite tuning sampling hyperparameters such as the temperature.

Finding 1: Gaining new information via interaction beats thinking more a single step. Figure 3 (top) plots the task success against total compute, measured by the number of tokens per trajectory in log scale. Among the three strategies, interaction scaling (green) shows the steepest upward trend, achieving the highest success rate as the allowed token budgets increase. Budget forcing (blue) yields moderate gains but plateaus around 0.26. Despite incurring the highest cost, best-of- n (orange) brings the least improvements, suggesting that repeatedly sampling actions per step is a less effective use of compute in interactive tasks.

A natural question that arises here is: *how should we distribute a bounded compute budget between running more interaction steps vs. reasoning longer?* These two dimensions present different costs per unit, which may not be known apriori. Hence, we illustrate how scaling the number of interaction steps and the number of tokens per step affect performance and the total number of tokens individually. Figure 3 (bottom) decomposes total compute into tokens per step (y-axis) and steps per rollout (x-axis), so the shaded areas indicate the average total compute per trajectory. Interaction scaling extends along the x-axis, while per-step reasoning scales along y-axis. We find that scaling *across* steps is more effective than scaling *within* steps in WebArena (Figure 3, top), likely because the former enables the agent to gather new information and enrich its context. This ability to query and observe external feedback is unique to agentic settings but not single-turn question-answering tasks. While standard per-step reasoning is constrained by the information already available at each step, our approach takes advantage of this dynamic interaction.

Finding 2: Prompting alone is insufficient for interaction scaling. While our results highlight the potential of scaling interaction, the “check-again” strategy only allows the agent to revisit its behavior upon task completion, it does not enable it to implement nuanced behaviors such as switching between exploration and exploitation in the middle of a rollout. We also experimented with combining interaction scaling with budget forcing and best-of- n (Appendix Table 6) and observe that simply increasing test-time compute via prompting does not yield additive gains. In fact, the final downward trend in Figure 3 (top) suggests that asking the agent to re-check too many times or think for too long can confuse it and degrade performance. This shows the need for methods that *train* agents to optimize for best behavior when scaling test-time interaction, rather than naïve prompting.

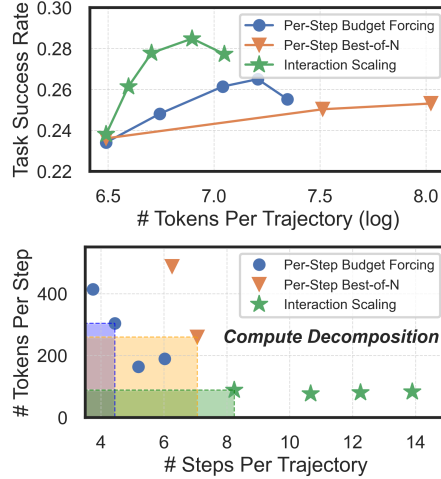


Figure 3: **Top: Task success rate vs. total compute cost** measured in tokens per trajectory, log scale. **Bottom: Decompose compute** into tokens per step and number of interaction steps (excluding the initial point for each approach).

5 TTI: Curriculum-Based Online RL for Scaling Interaction

How can we extend beyond prompting-based scaling to training agents that can effectively utilize interaction scaling? A natural starting point is to draw inspiration from current approaches for optimizing test-time compute [30, 71, 70] and extend these ideas to interactive settings. Specifically, we can run reinforcement learning (RL) with binary task rewards and longer task horizons. However, is this approach sufficient? We first describe the key challenges in learning to scale test-time interaction, and then develop our approach to address them via curriculum learning.

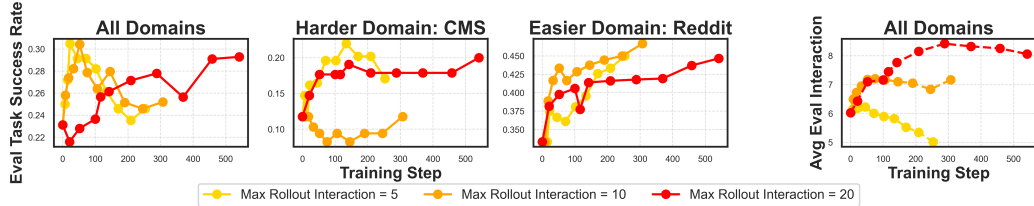


Figure 4: **Online RL with different values of maximum interaction horizon.** **Left:** success rates for different domains. “Harder” means lower success rate. **Right:** average rollout length (h_{stop}) on evaluation tasks.

5.1 Challenges of Training Agents with Fixed, Long Horizons

A natural way to encourage the agent to interact is to train at long horizons. To study this, we run the simplest form of REINFORCE [79] with binary rewards $R(\cdot)$, also known as online filtered behavior cloning (BC) or online STaR [20?]. Only successful trajectories are retained, and the agent is updated by maximizing the log-likelihood of actions conditioned on those high-reward rollouts:

$$\arg \max_{\theta} \mathbb{E}_{\mathcal{T} \sim \text{tasks}} \left[\mathbb{E}_{o_{0:h}, a_{0:h-1} \sim \pi(\cdot | \mathcal{T})} \left[\underbrace{\left(\sum_{t=0}^{h-1} \log \pi_{\theta}(a_t \mid o_{\leq t}, \mathcal{T}) \right)}_{\text{likelihood of trajectory}} \cdot \mathbb{1} \left[\underbrace{R(o_{0:h}, \mathcal{T}) = 1}_{\text{reward}} \right] \right] \right] \quad (1)$$

We use filtered BC as it is stable throughout training (no negative gradient [?]), has been utilized previously for training agents [20], and is a good “first-choice” algorithm for studying the role of interaction scaling. We scale the agent’s horizon on WebArena, varying $h \in \{5, 10, 20\}$. Smaller h exposes the agent only to exploitative rollouts that succeed within allowed time steps, while larger h also includes more exploratory rollouts. We use the non-test tasks for rollout. Details are in Appendix D.

As shown in Figure 4 (left), the agent trained with $h = 5$ learns quickly, likely because on-policy RL is more sample-efficient at smaller horizons, but it also quickly overfits, and performance decreases with more training (x-axis). This agent often terminates prematurely during evaluation despite being allowed to interact for much longer time. Conversely, agents trained at longer horizons generally learn policies that are quite stochastic and learn significantly more slowly due to higher variance of the loss and credit assignment challenges due to longer horizons [e.g., 80–82]. We manually inspect the trajectories and find that the $h = 20$ agent tends to associate exploratory actions such as “going back” or “trying random links” with high rewards initially. Noisy credit assignment with $h = 20$ slows learning, and only after several iterations do the agents begin to recover and produce more robust policies. The impact of horizon is domain-dependent: in complex domains requiring exploration like CMS, long-horizon agents outperform, while in simpler settings like Reddit, performance differences are minimal. We also note that the total number of tokens generated per *action* slightly decreases throughout training, so the training does not incentivize increasing the length of reasoning directly.

Importantly, although the interaction length increases as expected for $h = 20$ (Figure 4, right), noisy credit assignment and slower learning suggests that simply setting h to be large is insufficient to learn to scale test-time interaction reliably. These observations motivate our core idea: rather than fixing the horizon throughout training, our approach aims to scale their interaction length dynamically.

5.2 Our Approach: Curriculum Over Interaction Horizon

To address these challenges, we propose **TTI** (Test-Time Interaction), a curriculum-based online RL approach that trains the agent with short trajectories initially and gradually exposes it to longer ones. Existing curriculum learning methods in RL [e.g., 83–87] or web agents [23, 64] typically prioritize easier tasks before harder ones, often relying on predefined heuristics or external measures of task complexity. In contrast, we define curriculum progression in terms of the maximum number of steps an agent is allowed per trajectory. While interaction length can serve as a rough proxy for task difficulty, our approach does not require explicit labeling or estimation of task complexity.

How do we design a curriculum over the interaction horizon h ? Ideally, the curriculum should allow the agent to first learn basic, “atomic” skills to solve easier tasks, then progressively tackle complex

ones via skill chaining and exploration. To enable this, we explored two strategies: **(1)** a conservative, additive increase in h per iteration, giving the agent sufficient time to solidify core task-solving skills; and **(2)** a more aggressive, multiplicative increase, which assumes the agent can quickly acquire the basic skills and benefit from earlier exposure to exploratory behavior. Formally, for iteration i :

$$h_i := \text{clip}(h_{\min} + i, h_{\max}) \quad (\text{Additive curriculum}) \quad (2)$$

$$h_i := \text{clip}(h_{\min} \cdot i, h_{\max}) \quad (\text{Multiplicative curriculum}) \quad (3)$$

We store the rollouts in a replay buffer and assign higher weights to more recent trajectories. The full pseudocode for *TTI* and implementation details are provided in Appendix E.

Empirical insights. We instantiate these two strategies in WebArena, using the non-test tasks for online training. We set $h_{\min}$ to 10 and $h_{\max}$ to 30, and apply the schedules on top of filtered BC. Evaluation results after 10 iterations are shown in Table 2. Multiplicative curriculum outperforms the additive one, possibly because it exposes the agent to longer horizons early on and helps prevent it from overfitting prematurely to shortcut behaviors like always taking the shortest path. Based on these findings, we adopt the multiplicative curriculum as the default for *TTI*. Table 2 further shows that even with limited data (~ 700 training tasks), *TTI* outperforms fixed $h = 20$ in Figure 4 by nearly 3%, using 40% fewer training steps over 10 iterations. Next, we demonstrate that this advantage carries over to large-scale online RL training.

Table 2: *Comparing various curricula.* A multiplicative curriculum produces the best success rate.

Schedule	Task SR (%)
Additive	29.50
Multiplicative	32.25

6 Experiments: Scaling Up to Realistic Benchmarks

We now provide a comprehensive evaluation of *TTI* in large-scale, realistic environments. More generally, we demonstrate the effectiveness of training models to make good use of test-time interaction.

Synthetic task generation and evaluation. To enable large-scale training without training on the benchmark itself, we adopt synthetic task generation inspired by PAE [19]. Apart from prompting LLMs with seed examples or demos, we also deploy an exploratory agent to freely interact with websites and propose more grounded, diverse tasks. For evaluation, we leverage a prompting-based verifier that uses action histories and screenshots to label rollouts, achieving 88.9% accuracy compared to WebArena’s ground-truth evaluator. Details and prompt templates are in Appendix F.1. We evaluate agents on (1) WebVoyager [1] with 427 tasks across 13 domains (we replace Google search with Bing due to ReCaptcha issues); and (2) full WebArena [2] with 812 tasks.

Training. We obtain 128K synthetic tasks across diverse real-world domains for WebVoyager and 11K tasks for WebArena’s self-hosted domains (data released with the code). We train separate agents for each benchmark to avoid contaminating real-domain agents with synthetic environment data, as WebArena domains still have many differences from their real-world counterparts. We use Gemma 3 12B [78] as the base model, sampling 512 tasks per iteration for rollouts and updating with 512 successful on-policy trajectories. We apply a multiplicative curriculum with $h_{\min} = 10$ and $h_{\max} = 30$. We use vLLM [88] to sample rollouts and use DeepSpeed Zero 3 [89] with NVIDIA H100 GPUs for training. The evaluator is a Gemma 3 27B model, prompted to detect successful trajectories, which can then be used for the online filtered BC procedure. Other hyperparameters such as the number of iterations, learning rate, and the exact schedule of *TTI* can be found in Appendix F.3.

Comparisons. We evaluate zero-shot Gemma 3 12B and approaches that utilize a fixed horizon with $h \in \{10, 30\}$. We also compare to closed-source agents (e.g., those based on GPT-4 [46] and Claude [47]), open-weight models trained on proprietary data (e.g., UI-TARS [90]), and fully open-weight, open-data models (e.g., PAE [19]).

6.1 WebVoyager Results and Analysis

State-of-the-art open-weight, open-data performance. We report the overall task success rates (SR) on WebVoyager in Table 3 and Figure ?? (left). The *TTI*-trained Gemma 3 12B achieves an average SR of 64.8%, setting a new state-of-the-art among open agents trained purely on public data. While previous methods such as UI-TARS achieves a strong SR of 84.8%, they rely on private human-annotated data that remains inaccessible to the open-source community. In contrast, *TTI* is trained entirely on synthetic data and interestingly, this data is generated by the base model (Gemma

Table 3: *WebVoyager results*. *TTI* attains the best performance among open-weight agents trained on open-source synthetic data. Baseline results are taken from Zhou et al. [19], Qin et al. [90].

Model	Average	Allrecipes	Amazon	Apple	ArXiv	GitHub	ESPN	Coursera	Cambridge	BBC	Map	Search	HuggingFace	WolframAlpha
Proprietary Model														
Claude 3.7	84.1	-	-	-	-	-	-	-	-	-	-	-	-	-
Claude 3.5	50.5	50.0	68.3	60.4	46.5	58.5	27.3	78.6	86.0	36.6	58.5	30.2	44.2	66.7
OpenAI CUA	87.0	-	-	-	-	-	-	-	-	-	-	-	-	-
Agent E	73.1	71.1	70.7	74.4	62.8	82.9	77.3	85.7	81.4	73.8	87.8	90.7	81.0	95.7
Open Model, Proprietary Human-Annotated Data														
UI-TARS-1.5	84.8	-	-	-	-	-	-	-	-	-	-	-	-	-
Open Model, Open Synthetic Data														
LLaVa-34B SFT	22.2	6.8	26.8	23.3	16.3	4.9	8.6	26.8	67.4	16.7	12.2	23.3	20.9	38.1
PAE-LLaVa-7B	22.3	14.3	37.5	17.5	19.0	14.6	0.0	33.3	52.4	18.6	22.5	23.3	19.0	24.4
PAE-LLaVa-34B	33.0	22.7	53.7	38.5	25.6	14.6	13.6	42.9	74.4	39.0	22.0	18.6	25.6	42.9
Gemma 3 12B	55.8	25.7	32.3	45.5	60.6	54.8	60.6	56.3	69.6	65.6	54.8	72.7	66.7	61.1
Fixed $h = 10$	59.1	25.7	74.1	51.5	75.7	70.9	44.1	59.3	66.7	50.0	41.9	60.6	75.5	72.2
Fixed $h = 30$	45.2	20.0	41.9	60.6	42.4	41.9	50.0	34.4	60.6	25.0	29.0	63.6	45.5	69.4
TTI (Ours)	64.8	57.1	48.3	69.6	66.6	45.2	56.3	46.9	85.2	81.2	66.7	72.7	75.7	79.4

3 12B) itself, meaning that our training protocol implements a form of *self-improvement*. *TTI* also obtains the highest SR in 8 out of 13 domains, illustrating its efficacy.

TTI outperforms fixed-horizon via adaptive exploration. Table 3 also shows that our curriculum approach outperforms fixed $h = 10$ baseline by 5.7% and fixed $h = 30$ baseline by 19.6% in average accuracy. To better understand the use of interaction within a training rollout, we plot the average number of interaction steps on a held-out validation set with 78 tasks in Figure 5 (a). Note that the agent trained with $h = 10$ learns to continuously reduce the maximum number of steps it spends in a rollout, while $h = 30$ quickly drifts into aimless exploration and executes a larger number of steps pre-maturely in training, hindering performance. This aligns with our findings in Section 5.1 that simply running training at a longer horizon may not be sufficient for obtaining effective interaction scaling. In fact, we find that when training with *TTI*, the interaction length of the agent’s rollouts first decreases but then starts to increase as the maximum allowed horizon increases, indicating that an adaptive curriculum enables effective interaction scaling.

Figure 5 (d) shows that the task success rate also grows over time and correlates with the expanding horizon. While the average task success rates for *TTI* are better, we observe notable per-domain differences. Figure 5 (e) shows representative per-domain success rates. On domains like Allrecipes and Cambridge, *TTI* significantly outperforms fixed-horizon and zero-shot approaches, improving success rates by 31.4% and 15.6%, respectively, likely because these domains are highly information-dense and benefit from extended exploration enabled by adaptive interaction scaling. However, in domains like Amazon and GitHub, *TTI* underperforms the baselines. We notice that the base model already has strong knowledge about domain-specific terminologies (e.g., commit history, forks, stars) in these domains, resulting in high base performance. Inspecting the rollouts, we find that instead of using built-in filters and sorting, *TTI* can engage in exploration behaviors such as initiating Bing searches or consulting external sites. This exposes the agent to noisy or distracting information, reducing task success. We discuss these cases in Section 6.2.

Learning dynamics of TTI. To study how *TTI* enhances the “within-rollout” exploration capabilities of the agent, we measure the number of *GoBack* and *Bing* actions over the course of training. *GoBack* actions measure the number of retries the agent makes within an episode to get unstuck during exploration. *Bing* actions correspond to the number of times the agent attempts to seek information by moving to bing.com. As shown in Figure 5 (a, b, and d), the performance of *TTI* improves substantially as the number of *GoBack* and *Bing* actions and the trajectory length grow.

Also note that the trajectory length and the numbers of *GoBack* and *Bing* actions begin to increase with *TTI*, once the maximum allowed horizon length is increased as a part of the curriculum schedule (this regime is shown by the green shaded area in Figure 5). In contrast, these quantities continuously decrease over the course of training for the run with a lower number of maximum interaction steps ($h = 10$). We also find that the trajectory length shoots up substantially for the run with $h = 30$ and this correlates with worse performance. Finally, as shown in Figure 5 (c), as the agent’s trajectory grows longer with *TTI*, the number of tokens appearing in per-step reasoning actually becomes smaller. This implies that our agent is automatically learning to tradeoff interaction for per-step compute in order to attain higher performance, and perhaps prevents any issues with overthinking.

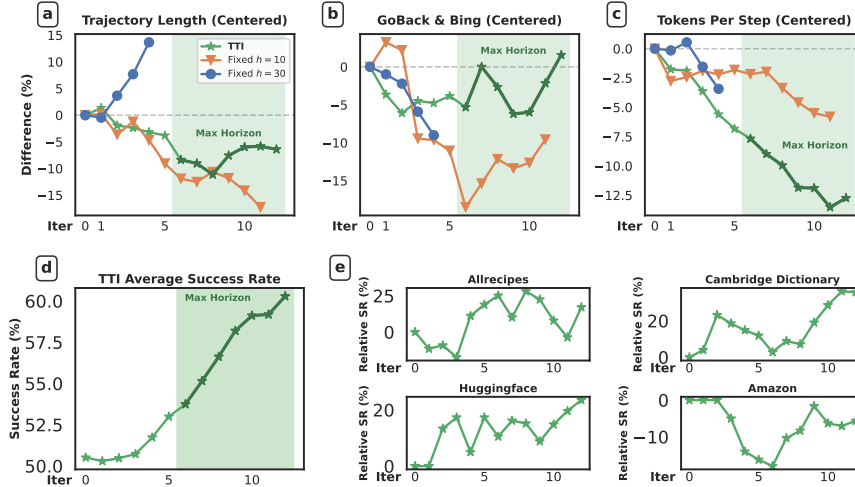


Figure 5: **Dynamics of TTI during training.** The green area represents the phase where the maximum allowed interaction horizon is the largest ($h = 30$), per our multiplicative schedule. All results are evaluated on a held-out subset of WebVoyager, not on the training tasks. **a:** Average trajectory length, i.e. the average number of steps taken in a trajectory normalized by the average length at the first iteration (iteration 0). **b:** Ratio of the sum of GoBack and Bing actions out of all actions normalized by the first iteration. **c:** The average number of tokens (CoT lengths) per action. **d:** Average task success rates for **TTI**. **e:** Per-domain success rates for **TTI**.

6.2 Case Studies: Strengths and Failure Modes

We conduct detailed case studies to analyze how **TTI** behaves across tasks and domains. These cases highlight both the strengths and remaining limitations of our approach.

Strength: Effective exploration in complex tasks (example visualized in Appendix F.4). For complex, exploratory tasks that require information retrieval, **TTI** trains the agent to extend its interaction horizon through search and backtracking, thus gathering and comparing information before making decisions. For instance, when tasked to find the baking temperature of an apple pie recipe with 4+ stars and 50+ reviews, our agent first selects a recipe but encounters a pop-up it cannot dismiss due to backend issues. It then tries another recipe but finds no baking temperature. Returning to the listing again, it correctly identifies one that meets all criteria. We also observe that such behaviors emerge progressively. In early training with shorter horizon, **TTI**-agent tends to stick to the first recipe it finds, keeps retrying it and saying *“I remember seeing one with 613 ratings earlier”* instead of seeking alternatives. Only after the training horizon becomes longer does it learn to explore and backtrack. This illustrates that when **TTI** runs a curriculum over interaction length, it teaches agents to adjust their horizon *within* a task and shift from exploitation to exploration. In contrast, training with a fixed short horizon can make it difficult to develop such exploratory behaviors.

Strength: Strategic exploitation in simple tasks (Appendix F.5). For simpler tasks with clear, deterministic paths (e.g., form filling or direct lookups), **TTI**-agent completes tasks efficiently without over-exploration. For example, when instructed to find the “top trending open-source project on machine learning” in GitHub, the agent goes directly to the Open-Source menu, selects the Trending tab, and performs search. This shows that **TTI** balances exploration and exploitation based on task context.

Despite these strengths, we also observe characteristic failure modes that point to areas for improvement and may partly explain the agent’s lower performance on domains like GitHub.

Failure mode: over-reliance on resets (Appendix F.6). When an action fails, our agent can reset the task by returning to the Bing search page rather than attempting domain-specific recovery actions (e.g., revisiting menus, refining filters). This suggests the agent treats search as a universal fallback. We also observe repeated resets within the same trajectory, indicating a lack of adaptive error recovery. While agents can extend horizons through both resetting and backtracking, the latter is often more natural. This highlights an area where **TTI** could improve by guiding exploration more systematically and enforcing structure, e.g., by using dense rewards [71, 68].

Failure mode: limited self-verification (Appendix F.7). We also observe that the agent can fail to verify its actions against the task goal. In one case, the agent identifies a 2021 GitHub repository for a task requiring one from 2022. While it explicitly acknowledges the mismatch, *“It was created in*

Table 4: **WebArena results**. For proprietary agents, we include the top 8 from the official leaderboard. Observe that TTI attains the best performance on an average, with especially better performance on the CMS domain.

	Method	Backbone	Average	Shopping	CMS	Reddit	GitLab	Maps
Proprietary-Based	IBM CUGA [92]	-	61.7	-	-	-	-	-
	OpenAI CUA [4]	-	58.1	-	-	-	-	-
	Jace AI [93]	-	57.1	-	-	-	-	-
	ScribeAgent [14]	GPT-4o + Qwen2.5 32B	53.0	45.8	37.9	73.7	59.7	56.3
	AgentSymbiotic [18]	Claude 3.5 + Llama 3.1 8B	48.5	48.7	41.2	63.2	47.2	57.8
	Learn-by-Interact [63]	Claude 3.5 Sonnet	48	-	-	-	-	-
	AgentOccam-Judge [33]	GPT-4	45.7	43.3	46.2	67.0	38.9	52.3
	WebPilot [34]	GPT-4o	37.2	36.9	24.7	65.1	39.4	33.9
Fully Open-Source (Self-Improvement)	Learn-by-Interact [63]	Codestral 22B	24.2	-	-	-	-	-
	AgentTrek [37]	Qwen2.5 32B	22.4	-	-	-	-	-
	AutoWebGLM [64]	ChatGLM3 6B	18.2	-	-	-	-	-
	NNetnav [94]	Llama 3.1 8B	7.2	7.4	4.2	0	0	28.5
	Zero-Shot Baseline	Gemma 3 12B	18.3	26.7	8.7	30.9	5.5	27.7
	Fixed $h = 10$	Gemma 3 12B	23.8	28.4	15.6	26.0	13.2	34.7
	Fixed $h = 30$	Gemma 3 12B	19.0	25.7	9.7	29.8	8.7	28.57
	TTI (Ours)	Gemma 3 12B	26.1	33.9	15.5	35.3	15.7	40.5

2021, not 2022, so it doesn’t meet the criteria”, it still submits it as the answer. This implies limited self-verification ability and could be mitigated by longer, more deliberate per-step reasoning. An important next step is to enrich **TTI** with the cognitive behaviors that enable per-step reasoning [91].

6.3 WebArena Results and Analysis

We further assess **TTI** on the full WebArena [2] (we only use the prompting-based verifier for training, but use the original benchmark evaluators for evaluation). As shown in Table 4 and Figure ?? (right), **TTI** obtains the highest performance among open-source agents trained entirely using a self-improvement procedure on self-generated data, without relying on proprietary models for task completion or distillation. While **TTI** improves over the zero-shot baseline by 7.8%, the gains are smaller than on WebVoyager, possibly because: (1) WebArena tasks are more complex, as reflected in lower accuracies even for proprietary models, leading to fewer successful rollouts per iteration and slower learning; (2) Agents occasionally attempt actions that are valid in real-world domains but invalid in WebArena counterparts (for example, search works reliably on Reddit but fails in WebArena’s Postmill due to environment bugs). More experiment details are in Appendix G.

Further scaling. While **TTI** equips agents with the ability to adjust their interaction horizon during deployment, an open question remains: Can we further amplify performance by combining TTI with inference-time interaction scaling techniques such as re-checking as discussed in Section 4? To explore this, we apply the “check-again” strategy (Section 4) to intermediate **TTI** checkpoints. Due to the high evaluation cost associated with evaluating on full WebArena, we leverage the WebArena subset checkpoints obtained in Section 5.2. As shown in Figure 6, applying re-checking on top of **TTI** improves task success across various training stages. The benefits are more obvious in the early stage, when the agent has a stronger bias to terminate prematurely. As training progresses, **TTI** encourages longer interaction that naturally incorporate behaviors like re-checking, reducing the benefit of explicit re-checks.

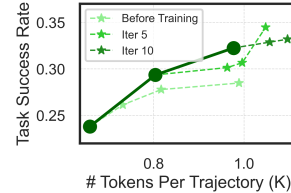


Figure 6: **We apply test-time re-checks to TTI checkpoints.**

7 Conclusion

In this work, we introduced interaction scaling as a new dimension of test-time scaling for interactive agents. Through empirical studies on web agents, we validate that interaction scaling enables agents to explore and adapt dynamically, significantly improving task performance. We hope that this work opens new directions in agentic reasoning and inspires broader applications beyond web navigation.

Limitations and future work. Our experiments are limited to web environments; extending this method to other domains like robotics or open-world games requires further exploration. Besides, scaling interaction steps increases computational costs during both inference and training. Although our adaptive scheduling helps, more efficient handling of long interactions is needed. In addition, our training relies on simple behavior cloning; future work could incorporate more advanced RL methods like PPO [95], GRPO [70] to improve performance. Lastly, due to high compute cost, we only ran the full benchmark once per setting, limiting the ability to quantify variance from policy, environment, and evaluator stochasticity. Future work should explore multiple runs or more robust evaluation.

Acknowledgments

We thank Jiayi Pan, Shanda Li, and Yuxiao Qu for feedback and informative discussions on an earlier version of this paper. This work was supported in part by the National Science Foundation grants IIS1705121, IIS1838017, IIS2046613, IIS2112471, the Office of Naval Research under N00014-24-1-2206, and funding from Meta, Morgan Stanley, Amazon, Google, and Scribe. We specially thank Scribe, The AGI company, and CMU FLAME Center (Orchard cluster) for providing a part of the GPU resources that supported this work. Any opinions, findings and recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of any of these funding agencies or employers.

References

- [1] Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models, 2024. URL <https://arxiv.org/abs/2401.13919>.
- [2] Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=oKn9c6ytLx>.
- [3] Claude. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku, 2024. URL <https://www.anthropic.com/news/3-5-models-and-computer-use>.
- [4] OpenAI. Introducing operator, 2025. URL <https://openai.com/index/introducing-operator/>.
- [5] Magnus Müller and Gregor Žunič. Browser use: Enable ai to control your browser, 2024. URL <https://github.com/browser-use/browser-use>.
- [6] Wenyi Hong, Weihan Wang, Qingsong Lv, Jiazheng Xu, Wenmeng Yu, Junhui Ji, Yan Wang, Zihan Wang, Yuxiao Dong, Ming Ding, and Jie Tang. Cogagent: A visual language model for gui agents, 2023.
- [7] Claude. Your code’s new collaborator, 2025. URL <https://www.anthropic.com/claude-code>.
- [8] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Rich Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. pi0.5: a vision-language-action model with open-world generalization. 2025. URL <https://api.semanticscholar.org/CorpusID:277993634>.
- [9] Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, and Sergey Levine. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. *ArXiv*, abs/2405.10292, 2024. URL <https://api.semanticscholar.org/CorpusID:269790773>.
- [10] Luyao Yuan, Zipeng Fu, Jingyue Shen, Lu Xu, Junhong Shen, and Song-Chun Zhu. Emergence of pragmatics from referential game between theory of mind agents, 2021. URL <https://arxiv.org/abs/2001.07752>.
- [11] Luyao Yuan, Dongruo Zhou, Junhong Shen, Jingdong Gao, Jeffrey L Chen, Quanquan Gu, Ying Nian Wu, and Song-Chun Zhu. Iterative teacher-aware learning. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 29231–29245. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/f48c04ffab49ff0e5d1176244fd6b65c-Paper.pdf.

- [12] Weixin Liang, Junhong Shen, Genghan Zhang, Ning Dong, Luke Zettlemoyer, and Lili Yu. Mixture-of-mamba: Enhancing multi-modal state-space models with modality-aware sparsity, 2025. URL <https://arxiv.org/abs/2501.16295>.
- [13] Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, Adrian Li-Bell, Danny Driess, Lachy Groom, Sergey Levine, and Chelsea Finn. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *ArXiv*, abs/2502.19417, 2025. URL <https://api.semanticscholar.org/CorpusID:276618098>.
- [14] Junhong Shen, Atishay Jain, Zedian Xiao, Ishan Amlekar, Mouad Hadji, Aaron Podolny, and Ameet Talwalkar. Scribeagent: Towards specialized web agents using production-scale workflow data. *ArXiv*, abs/2411.15004, 2024. URL <https://api.semanticscholar.org/CorpusID:274192657>.
- [15] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL <https://openreview.net/forum?id=kiYqb03wqw>.
- [16] Junhong Shen, Liam Li, Lucio M. Dery, Corey Staten, Mikhail Khodak, Graham Neubig, and Ameet Talwalkar. Cross-modal fine-tuning: align then refine. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.
- [17] Junhong Shen, Neil Tenenholtz, James Brian Hall, David Alvarez-Melis, and Nicolo Fusi. Tag-llm: Repurposing general-purpose llms for specialized domains, 2024.
- [18] Ruichen Zhang, Mufan Qiu, Zhen Tan, Mohan Zhang, Vincent Lu, Jie Peng, Kaidi Xu, Leandro Z. Agudelo, Peter Qian, and Tianlong Chen. Symbiotic cooperation for web agents: Harnessing complementary strengths of large and small llms. *ArXiv*, abs/2502.07942, 2025.
- [19] Yifei Zhou, Qianlan Yang, Kaixiang Lin, Min Bai, Xiong Zhou, Yu-Xiong Wang, Sergey Levine, and Erran L. Li. Proposer-agent-evaluator(pae): Autonomous skill discovery for foundation model internet agents. *ArXiv*, abs/2412.13194, 2024.
- [20] Hao Bai, Yifei Zhou, Mert Cemri, Jiayi Pan, Alane Suhr, Sergey Levine, and Aviral Kumar. Digirl: Training in-the-wild device-control agents with autonomous reinforcement learning. *ArXiv*, abs/2406.11896, 2024.
- [21] Hao Bai, Yifei Zhou, Erran L. Li, Sergey Levine, and Aviral Kumar. Digi-q: Learning q-value functions for training device-control agents. *ArXiv*, abs/2502.15760, 2025.
- [22] Junhong Shen and Lin F. Yang. Theoretically principled deep rl acceleration via nearest neighbor function approximation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35 (11):9558–9566, May 2021. doi: 10.1609/aaai.v35i11.17151. URL <https://ojs.aaai.org/index.php/AAAI/article/view/17151>.
- [23] Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning, 2025. URL <https://arxiv.org/abs/2411.02337>.
- [24] Claude. Claude takes research to new places, 2025. URL <https://www.anthropic.com/news/research>.
- [25] OpenAI. Introducing deep research, 2025. URL <https://openai.com/index/introducing-deep-research/>.
- [26] Google Gemini. Gemini deep research, 2025. URL <https://gemini.google/overview/deep-research/?hl=en>.
- [27] Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Fei-Fei Li, Hanna Hajishirzi, Luke S. Zettlemoyer, Percy Liang, Emmanuel J. Candes, and Tatsunori Hashimoto. s1: Simple test-time scaling. *ArXiv*, abs/2501.19393, 2025. URL <https://api.semanticscholar.org/CorpusID:276079693>.

- [28] Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for LLM problem-solving. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=VNckp7JEHn>.
- [29] Yinlam Chow, Guy Tennenholtz, Izzeddin Gur, Vincent Zhuang, Bo Dai, Sridhar Thiagarajan, Craig Boutilier, Rishabh Agarwal, Aviral Kumar, and Aleksandra Faust. Inference-aware fine-tuning for best-of-n sampling in large language models. *ArXiv*, abs/2412.15287, 2024. URL <https://api.semanticscholar.org/CorpusID:274965054>.
- [30] Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://openreview.net/forum?id=4FWAwZtd2n>.
- [31] Xiao Liu, Hanyu Lai, Hao Yu, Yifan Xu, Aohan Zeng, Zhengxiao Du, Peng Zhang, Yuxiao Dong, and Jie Tang. Webglm: Towards an efficient web-enhanced question answering system with human preferences, 2023.
- [32] Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Shane Gu, and Izzeddin Gur. Multimodal web navigation with instruction-finetuned foundation models, 2024. URL <https://arxiv.org/abs/2305.11854>.
- [33] Ke Yang, Yao Liu, Sapana Chaudhary, Rasool Fakoor, Pratik Chaudhari, George Karypis, and Huzefa Rangwala. Agentoccam: A simple yet strong baseline for llm-based web agents, 2024. URL <https://arxiv.org/abs/2410.13825>.
- [34] Yao Zhang, Zijian Ma, Yunpu Ma, Zhen Han, Yu Wu, and Volker Tresp. Webpilot: A versatile and autonomous multi-agent system for web task execution with strategic exploration, 2024. URL <https://arxiv.org/abs/2408.15978>.
- [35] Junhong Shen, Tanya Marwah, and Ameet Talwalkar. Ups: Towards foundation models for pde solving via cross-modal adaptation. *arXiv preprint arXiv:2403.07187*, 2024.
- [36] Junhong Shen, Kushal Tirumala, Michihiro Yasunaga, Ishan Misra, Luke Zettlemoyer, Lili Yu, and Chunting Zhou. Cat: Content-adaptive image tokenization, 2025. URL <https://arxiv.org/abs/2501.03120>.
- [37] Yiheng Xu, Dunjie Lu, Zhennan Shen, Junli Wang, Zekun Wang, Yuchen Mao, Caiming Xiong, and Tao Yu. Agenttrek: Agent trajectory synthesis via guiding replay with web tutorials. *ArXiv*, abs/2412.09605, 2024.
- [38] Yueqi Song, Frank Xu, Shuyan Zhou, and Graham Neubig. Beyond browsing: Api-based web agents, 2024. URL <https://arxiv.org/abs/2410.16464>.
- [39] Zongzhe Xu, Ritvik Gupta, Wenduo Cheng, Alexander Shen, Junhong Shen, Ameet Talwalkar, and Mikhail Khodak. Specialized foundation models struggle to beat supervised baselines, 2024. URL <https://arxiv.org/abs/2411.02796>.
- [40] Junhong Shen, Abdul Hannan Faruqi, Yifan Jiang, and Nima Maftoon. Mathematical reconstruction of patient-specific vascular networks based on clinical images and global optimization. *IEEE Access*, 9:20648–20661, 2021. doi: 10.1109/ACCESS.2021.3052501.
- [41] Boyuan Zheng, Michael Y. Fatemi, Xiaolong Jin, Zora Zhiruo Wang, Apurva Gandhi, Yueqi Song, Yu Gu, Jayanth Srinivasa, Gaowen Liu, Graham Neubig, and Yu Su. Skillweaver: Web agents can self-improve by discovering and honing skills. 2025. URL <https://api.semanticscholar.org/CorpusID:277634081>.
- [42] Jiayi Pan, Yichi Zhang, Nicholas Tomlin, Yifei Zhou, Sergey Levine, and Alane Suhr. Autonomous evaluation and refinement of digital agents. *arXiv preprint arXiv:2404.06474*, 2024.
- [43] Junhong Shen, Mikhail Khodak, and Ameet Talwalkar. Efficient architecture search for diverse tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

- [44] Renbo Tu, Nicholas Roberts, Mikhail Khodak, Junhong Shen, Frederic Sala, and Ameet Talwalkar. NAS-bench-360: Benchmarking neural architecture search on diverse tasks. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2022.
- [45] Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae, and Honglak Lee. Autoguide: Automated generation and selection of state-aware guidelines for large language model agents. *arXiv preprint arXiv:2403.08978*, 2024.
- [46] OpenAI. Gpt-4 technical report, 2024. URL <https://arxiv.org/abs/2303.08774>.
- [47] Anthropic. Introducing the next generation of claude, 2024. URL <https://www.anthropic.com/news/claude-3-family>.
- [48] Paloma Sodhi, S. R. K. Branavan, Yoav Artzi, and Ryan McDonald. Step: Stacked llm policies for web actions. In *Conference on Language Modeling (COLM)*, 2024. URL <https://arxiv.org/abs/2310.03720>.
- [49] Tamer Abuelsaad, Deepak Akkil, Prasenjit Dey, Ashish Jagmohan, Aditya Vempaty, and Ravi Kokku. Agent-e: From autonomous web navigation to foundational design principles in agentic systems. *ArXiv*, abs/2407.13032, 2024.
- [50] Wenduo Cheng, Junhong Shen, Mikhail Khodak, Jian Ma, and Ameet Talwalkar. L2g: Repurposing language models for genomics tasks. *bioRxiv*, 2024. doi: 10.1101/2024.12.09.627422. URL <https://www.biorxiv.org/content/early/2024/12/11/2024.12.09.627422>.
- [51] Lutfi Eren Erdogan, Nicholas Lee, Sehoon Kim, Suhong Moon, Hiroki Furuta, Gopala Anumanchipalli, Kurt Keutzer, and Amir Gholami. Plan-and-act: Improving planning of agents for long-horizon tasks, 2025.
- [52] Aviral Kumar, Vincent Zhuang, Rishabh Agarwal, Yi Su, John D. Co-Reyes, Avi Singh, Kate Baumli, Shariq Iqbal, Colton Bishop, Rebecca Roelofs, Lei M. Zhang, Kay McKinney, Disha Shrivastava, Cosmin Paduraru, George Tucker, Doina Precup, Feryal M. P. Behbahani, and Aleksandra Faust. Training language models to self-correct via reinforcement learning. *ArXiv*, abs/2409.12917, 2024.
- [53] Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language model agents. *arXiv preprint arXiv:2407.01476*, 2024.
- [54] Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *The Twelfth International Conference on Learning Representations*, 2024. URL <https://openreview.net/forum?id=Pc8AU1aF5e>.
- [55] Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. *arXiv preprint arXiv:2409.07429*, 2024.
- [56] Revanth Gangi Reddy, Sagnik Mukherjee, Jeonghwan Kim, Zhenhailong Wang, Dilek Hakkani-Tur, and Heng Ji. Infogent: An agent-based framework for web information aggregation. *ArXiv*, abs/2410.19054, 2024.
- [57] Shikhar Murty, Christopher Manning, Peter Shaw, Mandar Joshi, and Kenton Lee. Bagel: Bootstrapping agents by guiding exploration with language. *arXiv preprint arXiv:2403.08140*, 2024.
- [58] Shikhar Murty, Dzmitry Bahdanau, and Christopher D. Manning. Nnetscape navigator: Complex demonstrations for web agents without a demonstrator, 2024. URL <https://arxiv.org/abs/2410.02907>.
- [59] Brandon Trabucco, Gunnar Sigurdsson, Robinson Piramuthu, and Ruslan Salakhutdinov. Towards internet-scale training for agents. *ArXiv*, abs/2502.06776, 2025. URL <https://api.semanticscholar.org/CorpusID:276249229>.

- [60] Pranav Putta, Edmund Mills, Naman Garg, Sumeet Ramesh Motwani, Chelsea Finn, Divyansh Garg, and Rafael Rafailov. Agent q: Advanced reasoning and learning for autonomous ai agents. *ArXiv*, abs/2408.07199, 2024. URL <https://api.semanticscholar.org/CorpusID:271865516>.
- [61] Yifei Zhou, Andrea Zanette, Jiayi Pan, Sergey Levine, and Aviral Kumar. Archer: Training language model agents via hierarchical multi-turn rl. *ArXiv*, abs/2402.19446, 2024. URL <https://api.semanticscholar.org/CorpusID:268091206>.
- [62] Taiyi Wang, Zhihao Wu, Jianheng Liu, Jianye Hao, Jun Wang, and Kun Shao. Distrl: An asynchronous distributed reinforcement learning framework for on-device control agents. *ArXiv*, abs/2410.14803, 2024. URL <https://api.semanticscholar.org/CorpusID:273501605>.
- [63] Hongjin Su, Ruoxi Sun, Jinsung Yoon, Pengcheng Yin, Tao Yu, and Sercan Ö. Arik. Learn-by-interact: A data-centric framework for self-adaptive agents in realistic environments. *ArXiv*, abs/2501.10893, 2025.
- [64] Hanyu Lai, Xiao Liu, Iat Long Iong, Shuntian Yao, Yuxuan Chen, Pengbo Shen, Hao Yu, Hanchen Zhang, Xiaohan Zhang, Yuxiao Dong, and Jie Tang. 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, 2024.
- [65] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *ArXiv*, abs/2408.03314, 2024. URL <https://api.semanticscholar.org/CorpusID:271719990>.
- [66] Yangzhen Wu, Zhiqing Sun, Shanda Li, Sean Welleck, and Yiming Yang. Inference scaling laws: An empirical analysis of compute-optimal inference for problem-solving with language models. 2024. URL <https://api.semanticscholar.org/CorpusID:271601023>.
- [67] Karl Cobbe, Vineet Kosaraju, Mo Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *ArXiv*, abs/2110.14168, 2021. URL <https://api.semanticscholar.org/CorpusID:239998651>.
- [68] Amrith Rajagopal Setlur, Chirag Nagpal, Adam Fisch, Xinyang Geng, Jacob Eisenstein, Rishabh Agarwal, Alekh Agarwal, Jonathan Berant, and Aviral Kumar. Rewarding progress: Scaling automated process verifiers for llm reasoning. *ArXiv*, abs/2410.08146, 2024. URL <https://api.semanticscholar.org/CorpusID:273233462>.
- [69] Amrith Rajagopal Setlur, Nived Rajaraman, Sergey Levine, and Aviral Kumar. Scaling test-time compute without verification or rl is suboptimal. *ArXiv*, abs/2502.12118, 2025. URL <https://api.semanticscholar.org/CorpusID:276422443>.
- [70] DeepSeek-AI. Deepseek-rl: Incentivizing reasoning capability in llms via reinforcement learning. *ArXiv*, abs/2501.12948, 2025. URL <https://api.semanticscholar.org/CorpusID:275789950>.
- [71] Yuxiao Qu, Matthew Y. R. Yang, Amrith Rajagopal Setlur, Lewis Tunstall, Edward Beeching, Ruslan Salakhutdinov, and Aviral Kumar. Optimizing test-time compute via meta reinforcement fine-tuning. *ArXiv*, abs/2503.07572, 2025. URL <https://api.semanticscholar.org/CorpusID:276928248>.
- [72] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models, 2023. URL <https://arxiv.org/abs/2210.03629>.
- [73] Xiao Yu, Baolin Peng, Vineeth Vajipey, Hao Cheng, Michel Galley, Jianfeng Gao, and Zhou Yu. Exact: Teaching ai agents to explore with reflective-mcts and exploratory learning. *ArXiv*, abs/2410.02052, 2024. URL <https://api.semanticscholar.org/CorpusID:273098809>.

- [74] Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*, 2024.
- [75] Can Jin, Hongwu Peng, Qixin Zhang, Yujin Tang, Dimitris N. Metaxas, and Tong Che. Two heads are better than one: Test-time scaling of multi-agent collaborative reasoning. 2025. URL <https://api.semanticscholar.org/CorpusID:277781795>.
- [76] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Jun-Mei Song, Mingchuan Zhang, Y. K. Li, Yu Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *ArXiv*, abs/2402.03300, 2024.
- [77] Zora Zhiruo Wang, Apurva Gandhi, Graham Neubig, and Daniel Fried. Inducing programmatic skills for agentic tasks. 2025. URL <https://api.semanticscholar.org/CorpusID:277634286>.
- [78] Gemma Team. Gemma 3 technical report. *ArXiv*, abs/2503.19786, 2025. URL <https://api.semanticscholar.org/CorpusID:277313563>.
- [79] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.
- [80] Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M. Bayen, Sham M. Kakade, Igor Mordatch, and P. Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines. *ArXiv*, abs/1803.07246, 2018. URL <https://api.semanticscholar.org/CorpusID:4043645>.
- [81] Tingting Zhao, Hirotaka Hachiya, Gang Niu, and Masashi Sugiyama. Analysis and improvement of policy gradient estimation. *Neural networks : the official journal of the International Neural Network Society*, 26:118–29, 2011. URL <https://api.semanticscholar.org/CorpusID:2274728>.
- [82] Dotan Di Castro, Aviv Tamar, and Shie Mannor. Policy gradients with variance related risk criteria. *arXiv: Learning*, 2012. URL <https://api.semanticscholar.org/CorpusID:3109162>.
- [83] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *International Conference on Machine Learning*, 2009. URL <https://api.semanticscholar.org/CorpusID:873046>.
- [84] Sanmit Narvekar, Bei Peng, Matteo Leonetti, Jivko Sinapov, Matthew E. Taylor, and Peter Stone. Curriculum learning for reinforcement learning domains: A framework and survey. *ArXiv*, abs/2003.04960, 2020. URL <https://api.semanticscholar.org/CorpusID:212657666>.
- [85] Xin Wang, Yudong Chen, and Wenwu Zhu. A survey on curriculum learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:4555–4576, 2021. URL <https://api.semanticscholar.org/CorpusID:232362223>.
- [86] Tabet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher–student curriculum learning. *IEEE Transactions on Neural Networks and Learning Systems*, 31:3732–3740, 2017. URL <https://api.semanticscholar.org/CorpusID:8432394>.
- [87] Marcin Andrychowicz, Dwight Crow, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Joshua Tobin, P. Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Neural Information Processing Systems*, 2017. URL <https://api.semanticscholar.org/CorpusID:3532908>.
- [88] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*, 2023.

- [89] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020. URL <https://api.semanticscholar.org/CorpusID:221191193>.
- [90] Yujia Qin, Yining Ye, Junjie Fang, Haoming Wang, Shihao Liang, Shizuo Tian, Junda Zhang, Jiahao Li, Yunxin Li, Shijue Huang, et al. Ui-tars: Pioneering automated gui interaction with native agents. *arXiv preprint arXiv:2501.12326*, 2025.
- [91] Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, nathan lile, and Noah D. Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *ArXiv*, abs/2503.01307, 2025. URL <https://api.semanticscholar.org/CorpusID:276741915>.
- [92] Sami Marreed, Alon Oved, Avi Yaeli, Segev Shlomov, Ido Levy, Aviad Sela, Asaf Adi, and Nir Mashkif. Towards enterprise-ready computer using generalist agent. *ArXiv*, abs/2503.01861, 2025. URL <https://api.semanticscholar.org/CorpusID:276775684>.
- [93] JaceAI. Awa 1.5 achieves breakthrough performance on webarena benchmark, 2024. URL <https://www.jace.ai/post/awa-1-5-achieves-breakthrough-performance-on-webarena-benchmark>.
- [94] Shikhar Murty, Dzmitry Bahdanau, and Christopher D. Manning. Nnetnav: Unsupervised learning of browser agents through environment interaction in the wild. 2024. URL <https://api.semanticscholar.org/CorpusID:273162280>.
- [95] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

A Broader Impact

This work contributes to the development of more adaptive and capable AI agents by introducing a new test-time scaling dimension focused on interaction rather than per-step reasoning alone. While this approach improves robustness and generalization in open-ended environments, it also raises important considerations. Increased agent autonomy can amplify both the benefits and risks of deployment in real-world systems. Moreover, agents capable of richer behaviors could be applied to sensitive domains (e.g., customer service, education, or automation workflows) where unintended actions could have large impacts. We encourage future work to consider ethical safeguards, interpretability tools, and human-in-the-loop designs when deploying interaction-scaled agents. Our experiments are conducted entirely in simulated environments, and we hope this work inspires further research on controllable and trustworthy agent behavior under realistic constraints.

B Observation Space Design

We use the screenshot accompanied with the web page’s accessibility tree as our main observation. We study two versions of accessibility tree. **Rich accessibility tree** is modified from the WebArena code and looks like:

```
[21]: RootWebArea 'Dashboard / Magento Admin' focused: True; [0]: link 'Magento Admin Panel'; [1]: link 'DASHBOARD'; [2]: link 'SALES'; [3]: link 'CATALOG'; [4]: link 'CUSTOMERS'; [5]: link 'MARKETING'; [6]: link 'CONTENT'; [7]: link 'REPORTS'; [8]: link 'STORES'; [22]: link 'SYSTEM'; [23]: link 'FIND PARTNERS & EXTENSIONS'; [24]: heading 'Dashboard'; [9]: link 'admin'; [10]: link "; [25]: StaticText 'Scope: "; [12]: button 'All Store Views' hasPopup: menu; [13]: link 'What is this?'; [14]: button 'Reload Data'...
```

Simple accessibility tree is modified from the PAE code and looks like:

```
[1]: "Dashboard"; [2]: "Sales"; [3]: "Catalog"; [4]: "Customers"; [5]: "Marketing"; [6]: "Content"; [7]: "Reports"; [8]: "Stores"; [9]: "admin"; [12]: <button> "All Store Views"; [13]: "What is this?"; [14]: <button> "Reload Data"; [15]: "Go to Advanced Reporting"; [16]: "here";...
```

Rich tree contains more details such as the HTML tag and attributes like `required`, `hasPopup` compared to simple tree. However, it is much longer than simple tree and hence harder to optimize due to the increased context length. As simple tree gives more steady training dynamics, we mainly use it for our experiments.

C Preliminary Test-Time Experiments on WebArena

C.1 General Prompts

General Prompt

Imagine you are an agent browsing the web, just like humans. Now you need to complete a task. In each iteration, you will receive an observation that includes the accessibility tree of the webpage and a screenshot of the current viewpoint. The accessibility tree contains information about the web elements and their properties. The screenshot will feature numerical labels placed in the TOP LEFT corner of web elements in the current viewpoint. Carefully analyze the webpage information to identify the numerical label corresponding to the web element that requires interaction, then follow the guidelines and choose one of the following actions:

1. Click a web element.
2. Delete existing content in a textbox and then type content.
3. Scroll up or down the whole window.
4. Go back, returning to the previous webpage.
5. Answer. This action should only be chosen when all questions in the task have been solved.

Correspondingly, action should STRICTLY follow the format specified by one of the following lines:

```
Click [numerical_label]  
Type [numerical_label] [content]
```

Scroll [up/down]
 GoBack
 ANSWER [content]
 Some examples are:
 Click [8]
 Type [22] [Boston]
 Scroll [down]
 ANSWER [06516]
 Key guidelines you MUST follow:
 * Action guidelines *
 - Use either screenshot or accessibility tree to obtain the numerical_label. Sometimes the accessibility tree captures more elements than the screenshot. It's safe to select these elements without scrolling
 - For text input, use Type action directly (no need to click first). All existing texts in the textbox will be deleted automatically before typing
 - Preserve text inside quotation marks exactly as provided by user
 - You must not repeat the same actions over and over again. If the same action doesn't work, try alternative approaches
 - Use ANSWER only after completing ALL task requirements
 - Wrap content for Type and ANSWER with square brackets '[]'
 - Do not add quotation marks for search queries
 * Web navigation hints *
 {hint}
 Your reply should strictly follow the format:
 Thought: Your reasoning trace. A good practice is to summarize information on the current web page that are relevant to the task goal, then generate a high-level plan that contains the sequence of actions you probably need to take
 Action: Based on this reasoning, identify the single most optimal action. You should output it in the format specified above (under "STRICTLY follow the format")
 After each action, you'll receive a new observation. Proceed until task completion. Now solve the following task.
 Task: {task_goal}
 Current URL: {url}
 Screenshot of current viewpoint: attached
 Accessibility tree of current viewpoint: {accessibility_tree}

Beyond the above CoT prompt, we also tried using a more complex prompt for the thought process. However, this does not lead to significant gain in downstream accuracy (see Table 5), but it could increase training and inference cost, so we did not use it in the end.

Complex Prompt

Thought: You must analyze the current webpage thoroughly to guide your decision-making. Show your reasoning through these steps:
 - Summarization: Begin by understanding the page context - identify what type of page you're on (search results, form, article, etc.) and how it relates to your objective. Summarize important information on the webpage that might be relevant to task completion. Especially when the task requires to return some answers to a specific question, you should note down intermediate information that helps generate the answer.
 - Planning: Generate a checklist of subtasks required for completion and cross-out the subtasks you've completed. Identify the next logical subtask.
 - Verification: Verify all information you've entered so far. Check that your inputs match requirements in terms of spelling and format (you should not change the user-specified information, even if there're grammar errors). Verify if any selections for dropdown items align with the task objective. Identify if there're necessary fields that have not been filled in. Note that if the last few steps are repeating the same action, there must be missing or incorrect information.

- Backtracking: If the task requires exploring multiple webpages (e.g., orders, posts, item pages, etc) to find out an answer, consider if you need to issue GoBack and return to the previous web page.
 - Candidate Generation: After all the above reasoning, list the most relevant possible actions, evaluate pros and cons of each action, and finally select the most effective action to progress task.
- Action: Choose ONE of the following action formats:
- Click [numerical_label] - Click a specific element
 - Type [numerical_label] [content] - Input text into a field
 - Scroll [up/down] - Navigate the page vertically
 - GoBack - Return to previous webpage
 - ANSWER [content] - Provide final answer when task is complete

C.2 WebArena Prompts

Below are the content replacing “{hint}” in the general prompt.

General Hint

- Always save progress through appropriate buttons (Save, Submit, Post, etc.)
- Always remember to interact with dropdown options after expanding
- Clear filters before setting new ones

Reddit

- Always save progress through appropriate buttons (Save, Submit, Post, etc.)
- Always remember to interact with dropdown options after expanding
- Pay attention to words like "latest", "newest", "hottest" in the task objective, which require clicking the dropdown menu and select "New" or "Top" with the correct time range
- When selecting a subforum, you can either browse the dropdown menu in the "Submit" page or navigate to "Forums" and check all subforums by clicking on "Next" to go over all pages. You must try to find a subforum that exactly matches your query. If there's no exact match, pick the most relevant one, ideally the subforum is about objects or locations contained in the given objective
- "Trending" means "hot"
- To find out all posts or replies from a user, click the user name and then click "Submissions" or "Comments"

CMS

- Always save progress through appropriate buttons (Save, Submit, Post, etc.)
- Always remember to interact with dropdown options after expanding
- Clear filters before setting new ones
- Use date format: month/day/year (e.g., 1/1/16, 12/31/24)
- When searching phone numbers, remove the country code
- When searching product name, use single but not plural form
- When the web page contains a table, aggregate the rows with the same item

Shopping

- Always save progress through appropriate buttons (Save, Submit, Post, etc.)
- Always remember to interact with dropdown options after expanding
- Sort items by price by clicking the dropdown menu and set descending/ascending direction
- When searching product name, use single but not plural form
- If the objective requires only finding an item, stop at the item page without adding to cart
- To find out the quality of a product, search the item, click on review, and inspect its review

- Click "Page Next" to iterate over all orders
- Since there's no way to filter order history, click "View Order" for every order within a date range and inspect individually. If the condition is not met, go back

GitLab

- Always save progress through appropriate buttons (Save, Submit, Post, etc.)
- Always remember to interact with dropdown options after expanding
- Clear filters before setting new ones
- When searching a repo in gitlab, type only the project name after "/" in the search box

Map

- Always remember to interact with dropdown options after expanding
- When searching for a place, remove prepositions like in/on/by/at. For example, use "starbucks, craig street" instead of "starbucks on craig street". Put the city name at the end
- When there is no results shown up after search, rephrase the address and try again
- To find direction between two points, after entering the from and to addresses, select the correct transportation (foot/bicycle/car) before clicking "Go"
- When the given location is not a geological address, use your knowledge to infer the address

C.3 CoT Experiments for Base Agent

To enable efficient rollout collection, we spin up multiple Docker containers on a single GPU according to the official WebArena repository. We use the vLLM [88] engine for inference and apply the following inference hyperparameters for most of our experiments.

- max_new_tokens: 1024
- max_attached_imgs: 4
- temperature: 1
- top_p: 0.95

We randomly subsample 62 test tasks for analysis purposes. Below are the results of zero-shot agent vs CoT prompting. "CoT" uses the "General Prompt" in Section C.1. "Complex CoT" uses the "Complex Prompt" in Section C.1. .

Table 5: Base agent results averaged over 3 runs on WebArena subset.

Prompt	Task SR (%)
Action Only	14.76
CoT	23.81
Complex CoT	23.33

C.4 Scaling Trade-off Experiments

"Check-again" for interaction scaling. After the agent outputs the task-stop signal, we append the following prompts to the observation to induce it to check again.

Check-Again Prompt

Important: You returned an answer in the last step. Let's pause, check the web page, and think again. If you still think the task is finished, double-check your answer, revise it if need, and return a final answer. If not, continue the task. Your output should still be in the same "Thought:...Action:..." format.

Table 6: Comparing different inference-time prompting strategies. Results averaged over 3 runs on WebArena subset. All methods are applied once.

Inference-Time Strategy	Task SR (%)
Baseline	23.81
Check-again	26.14
Budget-forcing	24.81
Best-of- n	25.03
Check-again + Budget-forcing	26.33
Check-again + Best-of- n	27.36

When applying multiple re-checks, we slightly vary the prompts such as ‘*Before you finalize the answer, re-evaluate it in terms of the current web page—what exactly supports or contradicts it?*’ or ‘*Why do I believe this answer is correct? What on the page justifies it? Could an alternative answer be better?*’ Please refer to the code base for the exact prompt used.

Per-step budget forcing. Following [71], we use the phrases below to induce longer per-step thinking. The phrases are different to ensure that the model does not run into the scenario of endless repeating a phrase.

- First time: Wait, let me think deeper.
- Second time: But let me double-check.
- Third time: But hold on.

Per-step best-of- n . We tried both selecting by log likelihood and majority voting, with the latter showing slightly better results.

Additional results for combined scaling. Beyond evaluating each scaling method separately, we also tried combining methods along different axes.

D Online Filtered BC on WebArena

We use the following hyperparameters to obtain the training curves in Table 4. During training, the vision_tower of Gemma 3 is kept frozen because it is frozen during pretraining. Other hyperparameters can be found in our code in the supplementary material.

- num_iteration: 10
- actor_epochs: 1 # number of epochs to update the actor
- rollout_size: 512
- num_update_sample_per_iteration: 512
- lr: 1e-6
- optimizer: AdamW
- scheduler: WarmupCosineLR
- batch_size: 4
- grad_accum_steps: 2
- eval_horizon: 30

E TTI Implementation

We provide the pseudocode in Algorithm 1. For the replay buffer, to encourage the agent to learn from more recent examples, we assign weights based on recency when sampling rollouts to update the agent: for the k -th trajectory added to the buffer, its weight is $\frac{k}{|D|}$.

Algorithm 1 TTI: Filtered Behavior Cloning with Interaction Scheduling

```
1: Input: Agent policy  $\pi_\theta$ , Evaluator  $\mathcal{R}$ , Environment  $\mathcal{E}$ , Learning rate  $\alpha$ , Replay buffer  $\mathcal{D}$ ,  
   Interaction scheduler hyperparameters  $h_{\min}$ ,  $h_{\max}$   
2: Initialize policy  $\pi_\theta$  from pretrained model  
3: Initialize replay buffer  $\mathcal{D} \leftarrow \{\}$   
4: for each episode  $i$  do  
5:   Set interaction horizon  $h_i \leftarrow \text{get\_schedule}(i, h_{\min}, h_{\max})$   
6:   for each rollout to collect do  
7:     Initialize environment:  $s_0 \sim \mathcal{E}$   
8:     for each  $t$  in  $[1, h_i]$  do  
9:       Observe current state  $s_t$   
10:      Predict action  $\hat{a}_t \leftarrow \pi_\theta(s_t)$   
11:      Execute action  $\hat{a}_t$  in environment  
12:      Observe next state  $s_{t+1}$   
13:      if episode done then  
14:        Compute reward  $r_t \leftarrow \mathcal{R}(s_t, \hat{a}_t)$   
15:      else  
16:         $r_t \leftarrow 0$   
17:      end if  
18:      Store transition:  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, \hat{a}_t, r_t, s_{t+1})\}$   
19:    end for  
20:  end for  
21:  for sample successful trajectory in  $\mathcal{D}$  do  
22:    for  $t = 1$  to  $h_{\text{stop}}$  do  
23:      Accumulate loss:  $L(\theta) \leftarrow L(\theta) + \text{CrossEntropy}(\pi_\theta(s_t), \hat{a}_t)$   
24:    end for  
25:  end for  
26:  Update policy:  $\theta \leftarrow \theta - \alpha \nabla_\theta L(\theta)$   
27: end for
```

F WebVoyager Experiments

F.1 Task Generator & Evaluator Prompt

Task Generator Prompt

You are a website exploration assistant tasked with discovering potential tasks on websites. These tasks should be similar to a user-specified task and aim to complete some high-level goals such as booking restaurants in a website. Your goal is to freely explore websites and propose tasks similar to a given set of examples. For each iteration, you'll receive:

- An observation with the webpage's accessibility tree

- A screenshot showing numerical labels in the TOP LEFT corner of web elements

You will then generate possible tasks while exploring the website. You should imagine tasks that are likely proposed by a most likely user of this website. You'll be given a set of examples for reference, but you must not output tasks that are the same as the given examples. The generated tasks must be realistic and at least require 3 steps to complete. It cannot be too simple.

Response Format and Available Actions

Your reply for each iteration must strictly follow this format:

Thought: Analyze the current webpage thoroughly to guide your exploration. Examine the webpage's structure, content, and interactive elements to identify potential tasks that users might perform on this site. Decide whether you want to keep exploring or output some tasks

Tasks: If you think you are ready to generate some tasks, output them in the following format (note that different tasks are separated with double semicolons): GENERATE [task1;answer1;;task2;answer2]

Action: Then, to continue with your exploration, choose ONE of the following action formats:

- Click [numerical_label] - Click a specific element
- Type [numerical_label] [content] - Input text into a field
- Scroll [up/down] - Navigate the page vertically
- GoBack - Return to previous webpage

Examples:

Click [8]

Type [22] [Boston]

Scroll [down]

GENERATE [Find the company's phone number;(555) 123-4567;;Locate the price of the basic subscription plan;\$19.99/month]

Your final output should look like:

Thought: ...

Tasks: GENERATE [...] (this is optional, only generate when you are confident)

Action: ...

Critical Guidelines

Action Rules

- Use either screenshot or accessibility tree to obtain the numerical_label
 - For text input, use Type action directly (no need to click first)
 - Ensure proposed tasks are diverse and demonstrate different aspects of the website. The tasks must have diverse difficulty and require different number of steps (3-20) to complete.
 - Tasks should be clear, specific, achievable, and self-contained. It cannot be too general, e.g., related to 'any post', 'any product', 'any place'. It must not depend on any context or actions that you have performed, i.e., you must assume zero prior knowledge when someone wants to complete the task
 - Your task should be objective and unambiguous. The carry-out of the task should NOT BE DEPENDENT on the user's personal information such as the CURRENT TIME OR LOCATION
 - Your tasks should be able to be evaluated OBJECTIVELY. That is, by looking at the last three screenshots and the answer provided by an agent, it should be possible to tell without ambiguity whether the task was completed successfully or not
 - Answers should be precise (e.g., exact prices, specific information, exact text)
 - You should output both operational tasks (the goal is to complete some steps) and information retrieval tasks (the goal is to find some answer to return)
 - You must refer to the examples given and mimic the complexity and task structure. See how these tasks are self-contained and realistic
 - Your proposed task cannot be a single action like click, type! Tasks like 'Determine the number of uses for that term' is unacceptable because it is ambiguous as a stand-alone task; 'Uncheck Use system value' is unacceptable because it is not a complete task; 'Locate the total revenue for the last month' is unacceptable because 'last month' is ambiguous;
- After each action, you'll receive a new observation. Continue exploring and generating tasks.

Here're some examples: {example}

Current URL: {url}

Screenshot of current viewpoint: attached

Accessibility tree of current viewpoint: {accessibility_tree}

Evaluator Prompt

You are an expert in evaluating the performance of a web navigation agent. The agent is designed to help a human user navigate a website to complete a task. Your goal is to decide whether the agent's execution is successful or not.

As an evaluator, you will be presented with three primary components to assist you in your role:

1. Web Task Instruction: This is a clear and specific directive provided in natural language, detailing the online activity to be carried out.
2. Result Response: This is a textual response obtained after the execution of the web task. It serves as textual result in response to the instruction.

3. Result Screenshots: This is a visual representation of the screen showing the result or intermediate state of performing a web task. It serves as visual proof of the actions taken in response to the instruction.

- You **SHOULD NOT** make assumptions based on information not presented in the screenshot when comparing it to the instructions.
- Your primary responsibility is to conduct a thorough assessment of the web task instruction against the outcome depicted in the screenshot and in the response, evaluating whether the actions taken align with the given instructions.
- NOTE that the instruction may involve more than one task, for example, locating the garage and summarizing the review. Failing to complete either task, such as not providing a summary, should be considered unsuccessful.
- NOTE that the screenshot is authentic, but the response provided by LLM is generated at the end of web browsing, and there may be discrepancies between the text and the screenshots.
- Note that if the content in the Result response is not mentioned on or different from the screenshot, mark it as not success.
- NOTE that the task may be impossible to complete, in which case the agent should indicate this in the response. **CAREFULLY VERIFY THE SCREENSHOT TO DETERMINE IF THE TASK IS IMPOSSIBLE TO COMPLETE.** Be aware that the agent may fail because of its incorrect actions, please do not mark it as impossible if the agent fails because of its incorrect actions.

You should explicitly consider the following criterion:

- Whether the claims in the response can be verified by the screenshot. E.g. if the response claims the distance between two places, the screenshot should show the direction. **YOU SHOULD EXPECT THAT THERE IS A HIGH CHANCE THAT THE AGENT WILL MAKE UP AN ANSWER NOT VERIFIED BY THE SCREENSHOT.**
- Whether the agent completes **EXACTLY** what the task asks for. E.g. if the task asks to find a specific place, the agent should not find a similar place.

In your responses:

You should first provide thoughts **EXPLICITLY VERIFY ALL THREE CRITERION** and then provide a definitive verdict on whether the task has been successfully accomplished, either as 'SUCCESS' or 'NOT SUCCESS'.

A task is 'SUCCESS' only when all of the criteria are met. If any of the criteria are not met, the task should be considered 'NOT SUCCESS'.

F.2 Agent Prompt

WebVoayager

Imagine you are a robot browsing the web, just like humans. Now you need to complete a task. In each iteration, you will receive an observation that includes the accessibility tree of the webpage and a screenshot of the current viewpoint. The accessibility tree contains information about the web elements and their properties. The screenshot will feature numerical labels placed in the TOP LEFT corner of web elements in the current viewpoint. Carefully analyze the webpage information to identify the numerical label corresponding to the web element that requires interaction, then follow the guidelines and choose one of the following actions:

1. Click a web element.
2. Delete existing content in a textbox and then type content.
3. Scroll up or down the whole window.
4. Go back, returning to the previous webpage.
5. Navigate to Bing's homepage.
6. Answer. This action should only be chosen when all questions in the task have been solved. Correspondingly, action should **STRICTLY** follow the format specified by one of the following lines:

Click [numerical_label]

Type [numerical_label] [content]

Scroll [up/down]

GoBack

Bing
 ANSWER [content]
 Some examples are:
 Click [8]
 Type [22] [Boston]
 Scroll [down]
 Bing
 ANSWER [06516]
 Key guidelines you MUST follow:
 * Action guidelines *
 1. The predicted action should be based on elements as long as it's accessibility tree OR screenshot. Sometimes, accessibility tree or screenshot captures more elements than the other, but it's fine to use either one.
 2. To input text for search bars, no need to click textbox first, directly type content. After typing, the system automatically hits 'ENTER' key.
 3. When a complex task involves multiple questions or steps, select 'ANSWER' only at the very end, after addressing all of these questions or steps. Double check the formatting requirements in the task when ANSWER. Always think twice before using 'ANSWER' action!!!
 4. When specifying the content for 'Type' and 'ANSWER' actions, be sure to wrap the content with '[]'.
 5. Use 'GoBack' to return to the previous state, use it when you find the previous action incorrect.
 6. When you see a pop-up page, you should immediately 'GoBack' to the previous page.
 7. Use 'Bing' when you need to navigate to a different website or search for new information.
 Your reply should strictly follow the format:
 Thought: Your reasoning trace. A good practice is to follow this format:
 - Observation summary: where are you at now? list all elements that are related to the task goal. e.g. if you're trying to filter something out, list all filters visible.
 - Planning: what sequence of actions do you need take to achieve the task goal? give a high-level overview of the steps you need to take.
 - Possible actions: to achieve that plan, what are potential actions you need to do immediately and what's their effect? List at least 3 actions and analyze each of them.
 Action: Based on this reasoning, identify the single most optimal action. You should output it in the format specified above ("...STRICTLY follow the format...").
 After you issue an action, the user will execute it and provide a new observation. Now solve the following task.
 Task: {task_goal}
 Current URL: {url}
 Screenshot of current viewpoint: attached
 Accessibility tree of current viewpoint: {accessibility_tree}

F.3 Experiment Details

We use the following hyperparameters to obtain the WebVoyager results.

- num_iteration: 10
- actor_epochs: 1 # number of epochs to update the actor
- rollout_size: 512
- num_update_sample_per_iteration: 512
- lr: 1e-5
- optimizer: AdamW
- scheduler: WarmupCosineLR
- batch_size: 4
- grad_accum_steps: 2

- eval_horizon: 30 # note that train horizon is different for different methods, but evaluation horizon is kept the same

E.4 Case Studies: Strengths 1

Task: Locate a recipe for an American apple pie on Allrecipes with a rating of at least 4 stars and more than 50 reviews. Note the maximum temperature mentioned in the Directions.

Fully trained agent explores:

Thought:
- Observation summary: I am on the Allrecipes homepage. I need to find a recipe for American apple pie with a rating of at least 4 stars and more than 50 reviews. The search bar [2] is available, so I should use it to search for "American apple pie".
- Planning: First, I need to search for "American apple pie" using the search bar. Then, I need to browse the results and look for recipes with a rating of at least 4 stars and more than 50 reviews. Once I find a suitable recipe, I need to note the maximum temperature mentioned in the Directions.
- Possible actions:
1. Click [2]: This would focus the search bar, allowing me to type in the search term.
2. Type [2] [American apple pie]: This would directly enter the search term into the search bar and submit the search.
3. Scroll [down]: This would scroll down the page, but I need to search first.
I should start by searching for "American apple pie".
- Actions:
Type [2] [American apple pie]

Thought:
- Observation summary: I'm now on the search results page for "American apple pie". There are several recipes displayed, some with ratings and reviews. I need to find a recipe with a rating of at least 4 stars and more than 50 reviews.
- Planning: I need to scan the recipe results for those that meet the criteria. I will look for a star rating of 4 or higher and a review count greater than 50. Once I find a suitable recipe, I'll click on it to view the details and note the maximum temperature.
- Possible actions:
1. Scroll [down]: This will allow me to see more recipes.
2. Click [7]: This is a "Clear search" button, which isn't what I need right now.
3. Click on a specific recipe: This would take me to the recipe details page. I need to examine the recipes and their ratings/reviews first.
I should start by scrolling down to see more results.
- Actions:
Scroll [down]

Thought:
- Observation summary: I am still on the search results page for "American apple pie". I've scrolled down, and I see more recipes. I need to find a recipe with a rating of at least 4 stars and more than 50 reviews.
- Planning: I need to carefully examine each recipe, looking for a rating of 4 or higher and a review count greater than 50. The screenshot and accessibility tree show ratings are displayed and recipes have titles.
- Possible actions:
1. Scroll [down]: Scroll further down the page.
2. Click on a specific recipe: If I find a suitable recipe, click on it to view the details.
3. Click [24]: This seems to be a "Save Recipe" button, which I don't need right now.
Let's look at the first few recipes more closely. The first recipe "American Apple Pie" has a rating and reviews. Let's see if it meets the criteria.
- Actions:
Click [11]

Thought:
- Observation summary: I am now on the recipe details page for "All American Apple Pie". The rating is 4.4 stars, and it has 497 reviews. This meets the criteria of at least 4 stars and more than 50 reviews. I need to find the maximum temperature mentioned in the Directions.
- Planning: I will now scan the "Directions" section for the maximum temperature. I will carefully read through the directions to find the highest temperature mentioned.
- Possible actions:
1. Scroll [down]: Scroll down the page to find the "Directions" section.
2. Click on a specific button: There are various buttons, but none of them are relevant to finding the temperature.
3. Read the directions: This is the most logical action to find the temperature.
I will start by scrolling down to find the "Directions" section.
- Actions:
Scroll [down]

Thought:
- Observation summary: I've scrolled down the recipe page, and I see a pop-up asking me to send the recipe to myself. This is interfering with my ability to find the directions.
- Planning: I need to dismiss the pop-up before I can find the directions. The pop-up has a "Close" button, which I should click.
- Possible actions:
1. Click [22]: This looks like the "Close" button on the pop-up.
2. Scroll [down]: This won't work because the pop-up is blocking the way.
3. Click [17]: This seems to be part of the pop-up, so clicking it wouldn't be helpful.
I should click the "Close" button to dismiss the pop-up.
- Actions:
Click [22]

Thought:
- Observation summary: The previous action [Click [22]] was invalid, and I'm still seeing the "Would You Like to Send This Recipe to Yourself?" pop-up. My previous click attempt didn't work.
- Planning: I need to try a different approach to close the pop-up. Maybe the "Close" button isn't the best option. I'll try clicking the "YES, PLEASE" button instead, hoping it will dismiss the pop-up.
- Possible actions:
1. Click [23]: This seems to be the "YES, PLEASE" button on the pop-up.
2. Click [22]: Try clicking the "Close" button again, just in case.
3. Scroll [down]: This won't work because the pop-up is blocking the way.
Let's try clicking "YES, PLEASE".
- Actions:
Click [23]

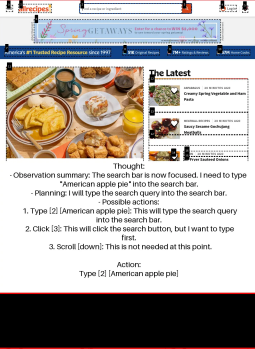
Thought:
- Observation summary: I'm back on the Allrecipes recipe page for "All American Apple Pie". The pop-up has disappeared. "Would You Like to Send This Recipe to Yourself?" This is exactly what I was trying to avoid.
- Planning: The pop-up is persistent and interfering with my progress. I need to try a different recipe. I'll go back and try a different recipe.
- Possible actions:
1. Go back: Return to the previous page to try a different recipe.
2. Click [22]: Close the pop-up (unavailable).
3. Click [23] "YES, PLEASE" (unavailable).
Going back is the best option.
- Actions:
Go back

Thought:
- Observation summary: I am back on the search results page for "American apple pie". This time, I'm determined to avoid the recipe with the persistent pop-up. I'll choose a different recipe.
- Planning: I need to scan the search results for a recipe with a rating of at least 4 stars and more than 50 reviews, but without the annoying pop-up. I will look for a recipe that seems promising and click on it.
- Possible actions:
1. Click on a specific recipe: This would take me to the recipe details page. I need to examine the recipes and their ratings/reviews first.
2. Scroll [down]: This will allow me to see more recipes.
3. Click [7]: This is the "Clear search" button, which isn't needed right now.
Let's try clicking on the recipe labeled "Southern Living American Apple Pie." It appears to have a rating and more than 50 reviews.
- Actions:
Click [9]

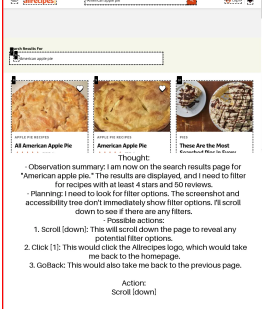
Early-stage agent prefers exploitation:



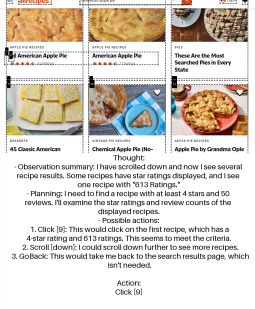
Thought:
- Observation summary: I am on the Allrecipes homepage. The task is to find an American apple pie recipe with a rating of at least 4 stars and more than 50 reviews. I need to search for "American apple pie" first.
- Planning: I need to use the search bar to search for "American apple pie." Then, I will need to filter the results by rating and reviews. Finally, I will note the maximum temperature mentioned in the recipe's directions.
- Possible actions:
1. Click [2]: This would click the search bar, allowing me to type in the search query.
2. Scroll [down]: This would scroll down the page, but it's not necessary yet.
3. GoBack: This would take me back to the previous page, which is not what I want.
Action:
Click [2]



Thought:
- Observation summary: The search bar is now focused. I need to type "American apple pie" into the search bar.
- Planning: I will type the search query into the search bar.
- Possible actions:
1. Type [2] (American apple pie): This will type the search query into the search bar.
2. Click [3]: This will click the search button, but I want to type first.
3. Scroll [down]: This is not needed at this point.
Action:
Type [2] (American apple pie)



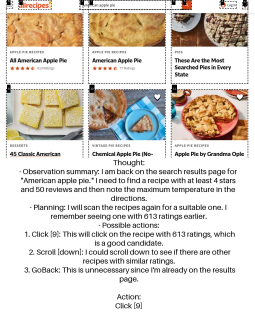
Thought:
- Observation summary: I am now on the search results page for "American apple pie." The results are displayed, and I need to filter for recipes with at least 4 stars and 50 reviews.
- Planning: I need to look for filter options. The screenshot and accessibility tree don't immediately show filter options. I'll scroll down to see if there are any filters.
- Possible actions:
1. Scroll [down]: This will scroll down the page to reveal any potential filter options.
2. Click [1]: This would click the Allrecipes logo, which would take me back to the homepage.
3. GoBack: This would take me back to the previous page.
Action:
Scroll [down]



Thought:
- Observation summary: I have scrolled down and now I see several recipe results. Some recipes have star ratings displayed, and I see one recipe with "11.3 ratings."
- Planning: I need to find a recipe with at least 4 stars and 50 reviews. I'll examine the star ratings and review counts of the displayed recipes.
- Possible actions:
1. Click [9]: This would click on the first recipe, which has a 4 star rating and 11.3 ratings. This seems to meet the criteria.
2. Scroll [down]: I could scroll down further to see more recipes.
3. GoBack: This would take me back to the search results page, which isn't needed.
Action:
Click [9]



Thought:
- Observation summary: I seem to have been redirected to a different page, a gallery of American dessert recipes. This is not what I expected. I think I clicked on something unexpectedly. I need to go back to the previous search results page.
- Planning: I need to use the "GoBack" action to return to the search results page for "American apple pie" recipes.
- Possible actions:
1. GoBack: This will navigate back to the previous page.
2. Click [1]: This is the Allrecipes logo, which might take me to the homepage.
3. Scroll [down]: This is not relevant now.
Action:
GoBack



Thought:
- Observation summary: I am back on the search results page for "American apple pie." I need to find a recipe with at least 4 stars and 50 reviews and then note the maximum temperature in the directions.
- Planning: I will scan the recipes again for a suitable one. I remember seeing one with 11.3 ratings earlier.
- Possible actions:
1. Click [9]: This will click on the recipe with 11.3 ratings, which is a good candidate.
2. Scroll [down]: I could scroll down to see if there are other recipes with similar ratings.
3. GoBack: This is unnecessary since I'm already on the results page.
Action:
Click [9]



Thought:
- Observation summary: I seem to be stuck on a gallery of American dessert recipes again. I keep getting redirected when I try to click on a recipe. The accessibility tree shows a search bar and a "Log in" button, but no recipe details.
- Planning: I suspect there's an issue with how the site is handling the recipe links. I need to go back to the search results and try a different approach.
- Possible actions:
1. GoBack: This will take me back to the search results page.
2. Click [1]: This would take me to the Allrecipes homepage.
3. Scroll [down]: This is not helpful at this point.
Action:
GoBack

E.5 Case Studies: Strengths 2

Task: Identify the latest top-trending open-source project in the category of ‘Machine Learning’ on GitHub, and check the number of stars it has received.

Thought:
Observation summary: I am on GitHub's homepage. I need to find the latest top-trending open-source project in the 'Machine Learning' category. The screenshot shows a search bar, navigation buttons including 'Open Source', and buttons for signing up. Planning: First, I need to explore the 'Open Source' section to see the trending projects. Then I'll need to filter projects by category.

Possible actions:

1. Click [6]: This will take me to the open source section, which seems like a good starting point.
2. Type [9]: This will open the search bar, allowing me to search for 'Machine Learning' projects.
3. Scroll [down]: This will scroll down the page, but I don't need to do it yet, since I want to go directly to the 'Open Source' section first.

Action:
Click [6]

Thought:
Observation summary: I am now on the 'Open Source' page. The page displays a list of open-source projects. I need to find the latest top-trending project in the 'Machine Learning' category. The screenshot shows a search bar and a 'Trending' tab. Planning: I need to navigate to the 'Trending' tab and then search for 'Machine Learning' projects.

Possible actions:

1. Click [10]: This will navigate me to the 'Trending' tab.
2. Type [14]: This will open the search bar, allowing me to search for 'Machine Learning' projects.
3. Scroll [down]: This will scroll down the page, but I want to go directly to the 'Trending' tab first.

Action:
Click [10]

Thought:
Observation summary: I am on the 'Trending' page. The page displays a list of trending repositories. I need to find the latest top-trending open-source project in the 'Machine Learning' category. The screenshot shows a search bar and several repository listings. The accessibility tree provides more details about each repository, including its name and the number of stars it has received. Planning: I need to search for repositories related to 'Machine Learning' within the trending projects. I can start by visually scanning the repository descriptions.

Possible actions:

1. Type [7]: This will open the search bar, allowing me to search for 'Machine Learning' projects. This seems like the most efficient way to find projects by category.
2. Scroll [down]: This will scroll down the page, but I want to use the search bar first.
3. Click [28]: This will navigate me to the first repository listed, which appears to be related to AI.

Action:
Type [7]

Thought:
Observation summary: The previous action [Type [7]] was invalid. I need to revise it. I'm still on the 'Trending' page, ready to search. The screenshot shows the search bar and some repository listings. The accessibility tree also lists the repositories and their stars. Planning: I need to be more careful in searching for 'Machine Learning' projects. I'll start by visually scanning the repository descriptions for mentions of 'Machine Learning' or related keywords.

Possible actions:

1. Type [7]: This will open the search bar again. I need to search for 'Machine Learning' or a related keyword.
2. Scroll [down]: This will scroll down the page, but I'll try searching first.
3. Click [28]: This would click on the first repository listed, which appears to be 'GoogleCloudPlatform / kubectrl'. I'm not sure if it's related to 'Machine Learning' yet.

Action:
Type [7]

Thought:
Observation summary: I'm on the 'Trending' page. I need to search for 'Machine Learning' projects. The screenshot shows the search bar, but the accessibility tree doesn't seem to be responding to it correctly. Planning: I'll try typing 'Machine Learning' directly into the search bar.

Possible actions:

1. Type [7] 'Machine Learning': This will directly type 'Machine Learning' into the search bar.
2. Click [11]: This will navigate me to the 'Explore' page, but I'm not ready to leave yet.
3. Scroll [down]: This might reveal more repositories, but I'll try typing first.

Action:
Type [7] 'Machine Learning'

Thought:
Observation summary: I'm on the search results page for 'Machine Learning' repositories. The page displays a list of repositories related to machine learning. I need to find the latest top-trending repository based on the number of stars. The screenshot shows 'wepeMachineLearning' as a top result. Planning: I need to examine the repositories and their star counts to determine the latest top-trending project. The accessibility tree lists several repositories and their star counts.

Possible actions:

1. Click [33]: This will navigate me to the 'wepeMachineLearning' repository. This seems like a good starting point since it's listed first.
2. Scroll [down]: This will scroll down the page, but I want to examine the top results first.
3. Click [45]: This will navigate me to the 'JosephMills/awesome-machine-learning' repository.

Action:
Click [33]

Thought:
Observation summary: I'm now viewing the 'wepeMachineLearning' repository. The page displays the repository's code, issues, pull requests, and other details. The accessibility tree shows information about the repository's contributors and commits. Planning: I need to verify that this is the top-trending project and check its star count. The description mentions 'Machine Learning and Deep Learning'. The tree lists '5,46 stars'.

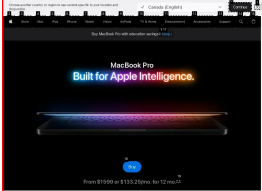
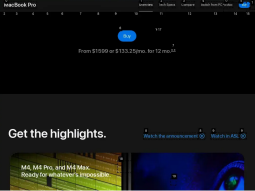
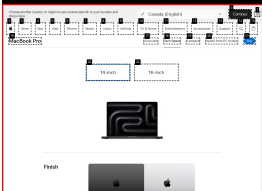
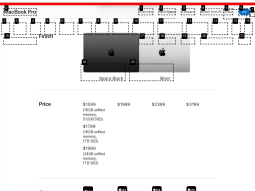
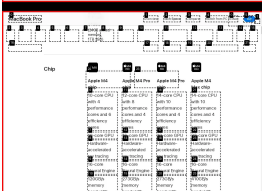
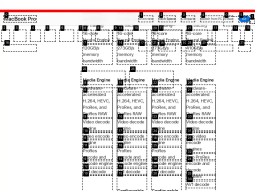
Possible actions:

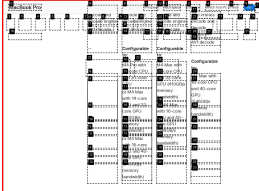
1. Answer: I've found the top-trending project ('wepeMachineLearning') and confirmed its star count (5,46). I can now provide the answer.
2. Scroll [down]: I could scroll down, but I've already found the relevant information.

Action:
ANSWER [5,46 stars]

E.6 Case Studies: Fail Modes 1

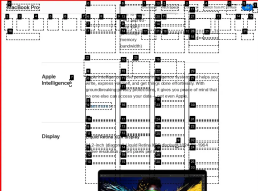
Task: On Apple's website, how many different types of keyboards are available when customizing your 14-inch MacBook Pro?

 Mother's Day It's not too late to find the perfect gift for Mom.	 Mac If you can dream it, Mac can do it.
Thought: - Observation summary: I am on the Apple homepage. The accessibility tree shows options for Mac, iPad, iPhone, Watch, AirPods, etc. The screenshot shows a large banner image and a "Shop" button. The task asks about the number of different types of keyboards available when customizing a 14-inch MacBook Pro. - Planning: First, I need to navigate to the Mac section of the website. Then, I need to find the MacBook Pro and customize it. Finally, I can count the different keyboard types. Possible actions: 1. Click [4] (Mac): This will take me to the Mac section of the website, which is the first step in finding the MacBook Pro. 2. Click [12] (Accessories): This might lead me to a page with keyboard options, but I need to first find the MacBook Pro. 3. Click [17] (Shop): This will take me to the shop, but I need to first find the MacBook Pro. Actions: Click [4]	Thought: - Observation summary: I am now on the Mac section of the Apple website. I can see different Mac models listed, including MacBook Air, MacBook Pro, Mac, Mac mini, Mac Studio, and Mac Pro. The screenshot also shows a "Shop" button and a banner image. - Planning: I need to find the MacBook Pro and start the customization process. Possible actions: 1. Click [17] (MacBook Pro): This will take me to the MacBook Pro page, which is what I need. 2. Click [16] (MacBook Air New): This would take me to the MacBook Air page, which is not what I need. 3. Click [27] (Shop): This will take me to the shop, but I need to first find the MacBook Pro. Actions: Click [17]
 MacBook Pro Built for Apple Intelligence.	 Get the highlights.
Thought: - Observation summary: I am on the MacBook Pro product page. I can see the "Buy" button, a price, and images of the MacBook Pro. - Planning: Now I need to start the customization process, which should include selecting the keyboard type. Possible actions: 1. Click [16] (Buy): This will take me to the buying page, but I need to customize the MacBook Pro first. 2. Scroll [down]: This might reveal more customization options. 3. Click [17] (Shop): This will take me to the shop, but I need to first customize the MacBook Pro. Actions: Scroll [down]	Thought: - Observation summary: I'm still on the MacBook Pro product page, further down than before. I can see more images and text about the MacBook Pro. - Planning: I need to find the keyboard customization options. Based on the previous steps and the accessibility tree, I should look for a "Tech Specs" or "Compare" option. Possible actions: 1. Click [2] (Tech Specs): This should take me to a page with detailed specifications, which might include keyboard details. 2. Click [3] (Compare): This might show a comparison between different models, which could reveal keyboard differences. 3. Scroll [down]: This might reveal more information, but I'm hoping for a more direct option. Actions: Click [2]
 MacBook Pro	 MacBook Pro
Thought: - Observation summary: I am now on the MacBook Pro technical specifications page. I can see a table comparing the 14-inch and 16-inch models. - Planning: I need to examine the specifications for the keyboards. I should look for a section that lists the different keyboard options. Possible actions: 1. Scroll [down]: To see if the keyboard specifications are further down the page. 2. Click [16] (Tech Specs): This seems redundant since I'm already on the tech specs page. 3. Click [16] (Compare): This might show a comparison of the keyboard features. Actions: Scroll [down]	Thought: - Observation summary: I'm on the MacBook Pro tech specs page, and it looks like I'm looking at a pricing table. I need to find the keyboard types. - Planning: I need to carefully review the specifications table to identify the different keyboard options. I'm looking for a "Keyboard" section or a comparison of keyboard features. Possible actions: 1. Scroll [down]: To see if the keyboard specifications are further down the page. 2. Click [17] (Shop): This would take me away from the comparison table. 3. Click [2] (Tech Specs): This is redundant as I'm already on the tech specs page. Actions: Scroll [down]
 MacBook Pro	 MacBook Pro
Thought: - Observation summary: I'm still scrolling down the MacBook Pro technical specifications page. I need to look for information about the keyboards themselves, not just the performance specs. - Planning: I need to refine my search on the specifications page. I'm looking for a specific section about the keyboard types available. Possible actions: 1. Scroll [down]: To see if the keyboard specifications are further down the page. 2. Click [17] (Shop): This would take me away from the comparison table. 3. Click [2] (Tech Specs): This is redundant as I'm already on the tech specs page. Actions: Scroll [down]	Thought: - Observation summary: I've scrolled further down the technical specifications page and am now looking at the "Configure" section. I can see various configurations for the MacBook Pro. - Planning: I need to examine the available keyboard configurations. The page shows options for "16-core unified memory, 512GB SSD" and other configurations. I'm looking for a clear indication of the keyboard types. Possible actions: 1. Scroll [down]: To see if there are more configuration options. 2. Click [17] (Shop): This would take me away from the configuration options. 3. Click [2] (Tech Specs): This is redundant as I'm already on the tech specs page. Actions: Scroll [down]



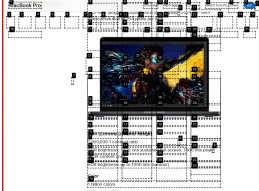
Thought:

- Observation summary: I'm still on the tech specs page, looking at the configuration options. I can see the M4 Pro and M4 Max configurations listed.
- Planning: I need to focus on identifying the different keyboard types. The page doesn't explicitly state the keyboard types, but mentions "configurable" options.
- Possible actions:
 - 1. Scroll (down): To see if more configuration details are available.
 - 2. Click [17] (Shop): This would take me away from the configuration options.
 - 3. Click [2] (Tech Specs): This is redundant as I'm already on the tech specs page.
- Actions:
 - Scroll (down)



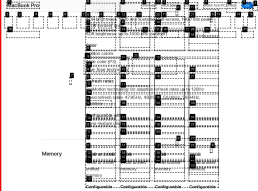
Thought:

- Observation summary: I'm on the MacBook Pro specifications page. The page displays various specifications like "Apple Intelligence," "Display," and "Keyboard."
- Planning: I need to count the different keyboard types. I see a "Keyboard" section and a "Display" section, but I need to look for different types within those categories.
- Possible actions:
 - 1. Scroll (down): To see if more keyboard options are listed.
 - 2. Click [17] (Shop): This would take me away from the specs page.
 - 3. Click [2] (Tech Specs): This is redundant as I'm already on the tech specs page.
- Actions:
 - Scroll (down)



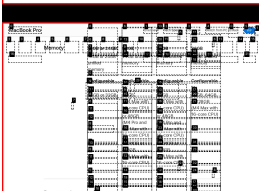
Thought:

- Observation summary: I'm on the MacBook Pro specifications page. I'm seeing details about the display, including "Liquid Retina XDR" and "1 billion colors." The page mentions "sustained brightness up to 1000 nits outdoors."
- Planning: I need to determine the number of different keyboard types. The page doesn't clearly state the keyboard types, but it does mention display specs.
- Possible actions:
 - 1. Scroll (down): To see if the keyboard information is further down the page.
 - 2. Click [17] (Shop): This would take me to a different page.
 - 3. Click [2] (Tech Specs): This would just bring me back to this same page.
- Actions:
 - Scroll (down)



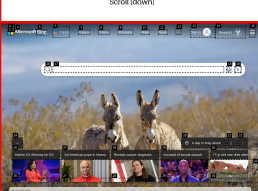
Thought:

- Observation summary: I am on the MacBook Pro specs page. I've been scrolling down, looking for information about the keyboards. I've seen the display specs, and it seems the page is focused more on the display than the keyboard.
- Planning: I need to refine my approach. The page might not directly list the keyboard types, but it mentions "configurable" options. I'll keep scrolling, but I'm not getting much information about the keyboard.
- Possible actions:
 - 1. Scroll (down): Continue scrolling, though I'm skeptical about finding more information.
 - 2. Click [17] (Shop): This would take me to a different page, which might have more options.
 - 3. Click [2] (Tech Specs): This would just bring me back to this same page.
- Actions:
 - Scroll (down)



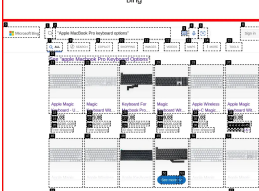
Thought:

- Observation summary: I'm still on the MacBook Pro specifications page, scrolling down. I've found the "Memory" section. I've been looking for keyboard information, but it's hard to find.
- Planning: The task asks about keyboard types. I'm not finding a dedicated keyboard section. I'm going to try searching for "keyboard" on the Apple website.
- Possible actions:
 - 1. Bing: Use Bing to search for "Apple MacBook Pro keyboard options."
 - 2. Click [4] (Switch from PC to Mac): This seems irrelevant to finding the keyboard information.
 - 3. Scroll (down): Continue scrolling, but I'm not optimistic about finding more information.
- Actions:
 - Bing



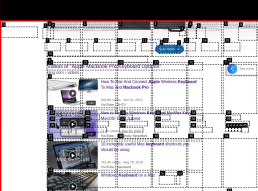
Thought:

- Observation summary: I'm on the Bing search page. I need to search for "Apple MacBook Pro keyboard options."
- Planning: I need to enter the search query into the search bar and submit the search.
- Possible actions:
 - 1. Type [17] "Apple MacBook Pro keyboard options": This will enter the search query into the search bar.
 - 2. Click [17] (Search icon): This will submit the search.
 - 3. Click [23] (Feedback): This is irrelevant to performing the search.
- Actions:
 - Type [17] "Apple MacBook Pro keyboard options"



Thought:

- Observation summary: I'm on the Bing search results page for "Apple MacBook Pro keyboard options." The page lists several different keyboard options, including the Apple Magic Keyboard, Keyboard for MacBook Pro, and Apple Wireless USB-C Magic Keyboard.
- Planning: Now I need to count the number of different keyboard types that are listed. I will list them and then count them.
- Possible actions:
 - 1. Scroll (down): To see if there are more keyboard options listed.
 - 2. Click [17] (Shop): This would take me to a shop, but I just want to count the options.
 - 3. Click [52] (See more): This might show more options, but I need to see what's already available first.
- Actions:
 - Scroll (down)



Thought:

- Observation summary: I'm still on the Bing search results page for "Apple MacBook Pro keyboard options." I've identified the following keyboard types:
 - 1. Apple Magic Keyboard (USB-C)
 - 2. Apple Wireless USB-C Magic Keyboard With Touch ID, US English
 - 3. Apple Keyboard for MacBook Pro 15" 2018 & 2019 A1990
 - 4. Keyboard for MacBook Pro 16" 2019 A2141
 - 5. Apple Magic Keyboard - White - US English
 - 6. Apple Magic Wireless Keyboard, White (MKG3LL/A)
- Planning: I've listed all the different keyboard types. Now I can count them to answer the question.
- Possible actions:
 - 1. Click [4] (Switch from PC to Mac): This is irrelevant to the task.
 - 2. Scroll (down): Unnecessary, as I've already listed all the options.
 - 3. Click [6] (See more): Unnecessary, as I've already listed all the options.
- Actions:
 - ANSWER [6]

Task: Which university maintains and manages ArXiv. Accessing the university’s website from ArXiv, how many undergraduate students are currently at the university.

arXiv

We are so thankful for our members - arXiv would not be possible without their support, arXiv members are universities, research labs, libraries, and consortia that contribute funding to arXiv to help advance the cause of open science.

User feedback on arXiv comes from several sources:

- Consortia
- Observation summary: I'm new on the "Our Members" page. The page lists various consortia and contributors. Cornell University is still visible.
- Planning: I need to confirm Cornell University's role. The page mentions "Cornell DH and HGF Consortium (Germany)". This seems to be consortium involving Cornell. It could find more information about the consortium.
- Possible actions:
 - Click [1]: This takes me back to the previous page.
 - Click [18]: This is the link for "Cornell DH and HGF Consortium Germany".
 - Click [16]: This is the link for "Canadian Research Knowledge Network Canada".
- Action: Click [1]

Cornell University

Our Members 2023 - 2024

We are so thankful for our members - arXiv would not be possible without their support, arXiv members are universities, research labs, libraries, and consortia that contribute funding to arXiv to help advance the cause of open science.

User feedback on arXiv comes from several sources:

- Consortia
- Observation summary: I'm on the Bing's homepage.
- Planning: How I need to find the number of undergraduate students at Cornell University. The page doesn't seem to directly provide this information. I need to check other sources.
- Possible actions:
 - Click [1]: This takes me back to the "Our Members" page.
 - Bing IP: Bing search engine to search Cornell University undergraduate student population.
- Action: Bing

"Cornell University undergraduate student population"

There are 26,284 students including 16,051 undergraduate and 10,233 graduate students at Cornell University by the academic year 2023-2024. By ethnicity status, there are 26,384 full-time and 630 part-time students with gender distribution of 12,342 male and 13,922 female students.

Observation summary: I'm on the Bing search results page for "Cornell University undergraduate student population". The first result from univstats.com provides the answer.

Planning: The first result states: There are 26,284 graduate students including 16,051 undergraduate and 10,233 graduate students at Cornell University for the academic year 2023-2024. By attending status, there are 26,384 full-time and 630 part-time students with gender distribution of 12,342 male and 13,922 female students. *The undergraduate student population is 16,051.

Possible actions:

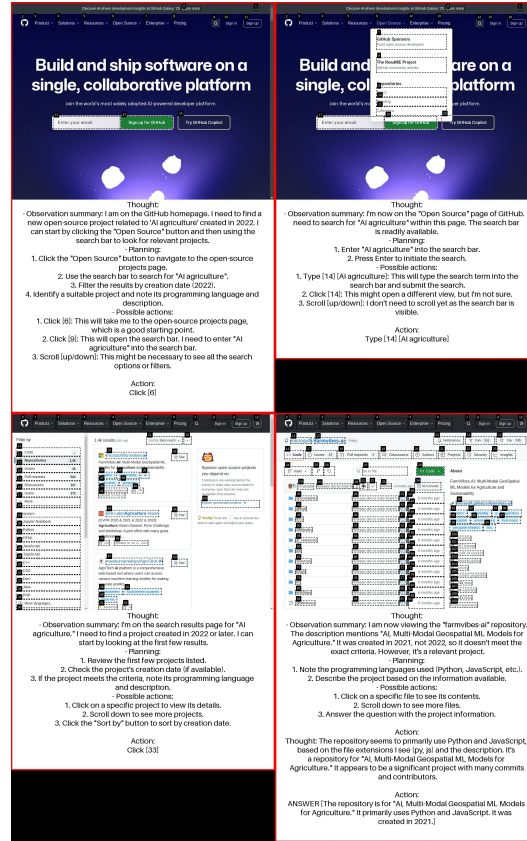
- Click [17]: This goes to the univstats.com page.
- Click [16]: This goes to the Cornell University Student Population and Demographics page.

Answer: I have the information I need to answer the question.

Action: ANSWER [16,051]

E.7 Case Studies: Fail Modes 2

Task: Identify a new open-source project on GitHub related to 'AI agriculture' that created in 2022, and note its main programming language and description.



G Full WebArena Experiment Details

We use the following hyperparameters to obtain the full WebArena results.

- num_iteration: 10
- actor_epochs: 1 # number of epochs to update the actor
- rollout_size: 512
- num_update_sample_per_iteration: 512
- lr: 1e-6
- optimizer: AdamW
- scheduler: WarmupCosineLR
- batch_size: 4
- grad_accum_steps: 2
- eval_horizon: 30

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [NA].
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes]" is generally preferable to "[No]", it is perfectly acceptable to answer "[No]" provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No]" or "[NA]" is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist",**
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers.**

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: Our contribution is to propose a new axis of test-time scaling for agent settings and design effective methods for leveraging the benefits during online training. These are reflected in the abstract and introduction.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: See the Conclusion section.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: There is no theoretical result in this paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We provide all training and inference configurations, hyperparameters, LLM prompts in the main text and appendix. We also include the code in the supplementary materials.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We use open-source benchmarks and release our code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [\[Yes\]](#)

Justification: Yes, see the appendices.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [\[Yes\]](#)

Justification: We report numbers averaged three trials.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [\[Yes\]](#)

Justification: Yes, we specify the GPUs we used.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.

- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines?>

Answer: [Yes]

Justification: The paper follows the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: See Broader Impact section.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: We do not use data or train models with high risk of misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.

- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The datasets are cited.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: The paper does not release new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: The paper does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: We discussed how we prompt and train LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.