STRUCTURED DISTILLATION OF WEB AGENT CAPABILITIES ENABLES GENERALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Web agents powered by frontier models are increasingly able to navigate complex websites, but they rely heavily on proprietary APIs, which increases costs and hinders for local deployments. To solve this, smaller models can be hosted locally, but are often less capable. We take a step towards bridging this gap by introducing AGENT-AS-ANNOTATORS, an automatic framework for distilling the capabilities of frontier models to navigate websites into smaller models, which substantially improves their abilities to solve tasks on websites. The framework is designed to replicate how humans would collect trajectories for training web agents; but the entire pipeline is automated using modular LLM components based on frontier models. Using Gemini 3 Pro as the teacher model, we generate 3000 trajectories across six web environments from WebArena, which we filter and use to train a student (Qwen-3.5-9B) with only supervised finetuning (SFT). We find that the resulting agent can transfer beyond the environments used in its training. Notably, on WorkArena L1, success rises from 33.3% to 51.5%, and performance also improves across three additional benchmarks, each testing a different type of generalization. On WebArena, it achieves a success rate of 41.5% despite never seeing any of the benchmark tasks, which nearly doubles the best same-size agent (Go-Browse, 21.7%). Through ablations, we find that the model benefits from each of its components (personas, judge, hints) and reasoning traces. Our results show that structured trajectory synthesis from a single teacher can produce competitive yet locally deployable web agents. We release the code, data and family of models (ranging from 2 to 9B parameters) publicly.

1 INTRODUCTION

Frontier models are increasingly more capable at completing realistic web tasks (Wang et al., 2023; Yao et al., 2022b; Wei et al., 2022), ranging from forms filling and database querying to managing content across many applications. On a benchmark like WebArena (Zhou et al., 2023b), which tests an agent's ability to solve tasks on six realistic websites, the top frontier models can complete over 68% of the tasks (Lu et al., 2026). However, proprietary and large models are primarily accessible through expensive APIs, which can make them impractical and costly to use in daily tasks. Open-weight models with ∼9B parameters or less (Yang et al., 2024) are far more practical for local usage, but are less capable on specialized tasks like web navigation. On WebArena, a model like Qwen3.5-9B achieves 31% success rate (SR), substantially behind frontier models.

In order to close this gap, we examine the use of *agentic capability distillation* for web tasks, which we characterize as an approach for distilling the capabilities of a frontier model (i.e., the teacher) pertaining to completing web tasks onto smaller models that can be hosted locally (i.e., the student).

Recent works have explored distillation to improve smaller web agents using frontier models: InSTA (Trabucco et al., 2025) trains 1.7B models on internet scale data spanning 150K websites, whereas NNetNav (Murty et al., 2024) leverages retroactive labels and exploration to produce high quality trajectories for training Llama models (Dubey et al., 2024). Although these methods are shown to be effective, it is clear that their designs differ in ways that are difficult to compare systematically. This raises the need for a common framework that can define and isolate the role of each component in the broader pipeline.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

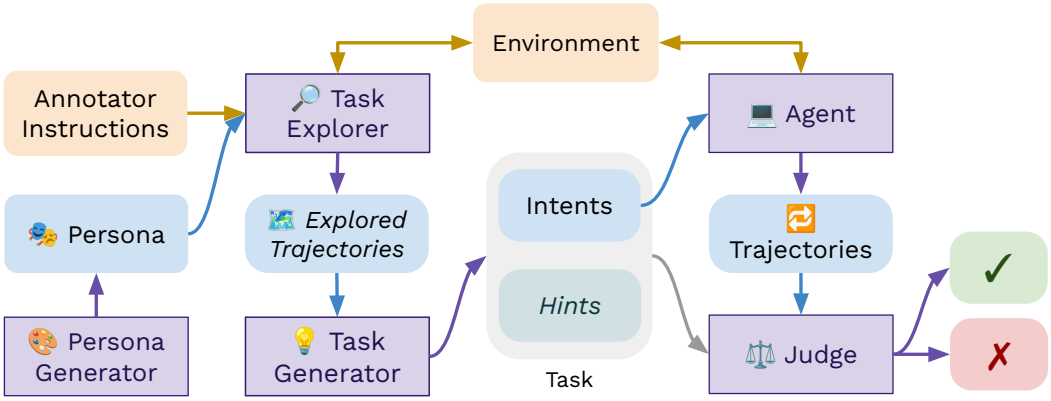

Figure 1: The AGENT-AS-ANNOTATORS pipeline replaces three human annotation roles with LLM modules. The Task Designer is replaced by a **Persona Generator** and **Task Generator** that synthesize task intents; the Annotator is replaced by an **Web Agent**; the Reviewer is replaced by a LLM-based **Judge**. Only successful trajectories are used to train the student.

To this end, we introduce AGENT-AS-ANNOTATORS, a framework that rethinks agentic capability distillation by drawing inspiration from how human contributors annotate tasks for web agent benchmarks. Using prior works (Zhou et al., 2023b; Deng et al., 2023; Lù et al., 2024) as inspiration, we find that human annotators played three distinct roles during the task collection process: a **Task Designer** first explored several web sandboxes in order to design tasks (with optional evaluation hints); an **Annotator** then attempts to complete the tasks, producing expert human trajectories; a **Reviewer** will then verify that the tasks are properly completed based on the original intent. Based on those roles, we design AGENT-AS-ANNOTATORS to replace each role with model-based modules: a Persona Generator and Task Generator in place of the Task Designer, a capable web agent teacher instead of Annotator, and a Judge with access to the evaluation hints in addition to the trajectories to replace Reviewer; we show the full pipeline in Figure 1. Naturally, we can express prior methods (Murty et al., 2024; Trabucco et al., 2025) as instantiations of our method simply by changing individual module (making it possible to keep the rest of the framework untouched); a comparison can be found in Table 2.

Equipped with our novel framework for studying agentic capabilities distillation, we instantiate AGENT-AS-ANNOTATORS using a Gemini 3 Pro teacher and Qwen-3.5 family as students. Using the strong teacher, we collect and release A3-SYNTH, a high quality dataset of 3000 tasks (collected across six WebArena environments) for training web agents. With 16,353 training examples after judge filtering, our dataset is the most size-efficient training dataset relative to prior works, yet enables the models to achieve stronger performance: a Qwen3.5-9B student trained on those tasks achieves 51.5% success rate (SR) on WorkArena L1, surpassing the 33.3% baseline. When evaluated on environments it has seen during training (i.e., WebArena), it achieves 41.5%, despite never seeing any of the tasks used in the evaluation split, establishing a new state-of-the-art for web agents under 27B parameters (Table 1).

We summarize our contributions as follows: we introduce (1) AGENT-AS-ANNOTATORS, a methodological framework for researchers to design and build efficient pipelines to distill frontier models's capabilities into smaller models, which enables systematic comparison (Section 3); (2) A3-SYNTH, a novel dataset of 3,000 web tasks collected inside WebArena's environments yet remains out-of-distribution with respect to its evaluation tasks (Section 4); (3) A3-Qwen family of web agents, which substantially surpass other web agents on several benchmarks, including both in-domain and unseen environments (Section 5).

2 RELATED WORK

Web agent benchmarks and methods. Evaluation of web agents started with simplified synthetic environments like World of Bits (Shi et al., 2017) and MiniWoB (Liu et al., 2018), which

later inspired benchmarks based on simulated online shopping platforms (Yao et al., 2022a). Later work attempt to achieve a higher degree of realism: Deng et al. (2023); Lù et al. (2024) focuses on human-collected trajectories from real websites, whereas to WebArena (Zhou et al., 2023b) designed 6 realistic self-hosted environments with automatic verifiers to determine success. VisualWebArena (Koh et al., 2024a) later extends WebArena to tasks requiring strong vision capabilities, whereas WorkArena (Drouin et al., 2024) and WorkArena++ (Boisvert et al., 2024) focused on real enterprise environments. OSWorld (Xie et al., 2024) and AndroidWorld (Rawles et al., 2024) broaden the categories of tasks to the full desktop environment and mobile devices, respectively. BrowserGym (de Chezelles et al., 2024) unifies several environment-based benchmarks under a single Gym-based interface (Brockman et al., 2016). The task creation process proposed in these benchmarks directly inspires our AGENT-AS-ANNOTATORS framework as they involve task design, annotation, and supervision roles.

Trajectory generation for web agents. Although past works (Deng et al., 2023; Lù et al., 2024; Xu et al., 2025) relied on human annotators to produce a large amount of trajectories, the labor cost can quickly become unsustainable, and the repetitive nature of the tasks can be demoralizing. Instead, several approaches explore ways to automatically collect trajectories with a frontier models. InSTA (Trabucco et al., 2025) collects tasks across 150K websites, which was filtered from the full internet following several criteria based on safety and relevance. On the other hand, NNetNav (Murty et al., 2024) retroactively labels exploratory trajectories to, which is subsequently filtered via an LLM judge. AgentTrek (Xu et al., 2024) use web tutorials to replay task with VLM verification, whereas Explorer (Pahuja et al., 2025) scales exploration-driven synthesis by iteratively refining exploration into task labels. Additionally, Go-Browse (Gandhi & Neubig, 2025) frames trajectory collection as a graph search over URLs, where the feasibility can be verified using a vision-enabled LLM. Although all these approaches are effective, they each differ in how their pipeline are designed. To validate our framework, we show in Table 2 that their modules can be mapped to AGENT-AS-ANNOTATORS modules, and we discuss the core differences in Section 3.2. Beyond methods that focus on pipeline design, several works explore finetuning on agent-trajectories, including FireAct (Chen et al., 2023), AgentTuning (Zeng et al., 2023), and WebRL (Qi et al., 2024); the latter generates new tasks from unsuccessful attempts.

Distillation and synthetic trajectories. Several works have shown the effectiveness of synthetic data for training models (Wang et al., 2022; Ge et al., 2024); Mukherjee et al. (2023); Hsieh et al. (2023); Gunasekar et al. (2023) show that stronger models can be effectively used to train weaker ones. Additionally, providing access to privileged information allows a base model to act as a teacher in a distillation setting (Penaloza et al., 2026). Moreover, Zhou et al. (2023a) found that the quality of the data can outweigh the benefits of a greater number of training examples. We extend insights from past works to the web agents setting, where our training data consists of trajectories stemming from interactions with web environments across many steps.

LLM-based judges and automatic evaluation. LLM judges have been shown to be effective for evaluating LLMs in several settings, from chatbots (Zheng et al., 2023) to long-form responses (Kim et al., 2023). Lambert et al. (2024) introduce a benchmark to evaluate the quality of automatic judges (i.e., reward models) that provide signals to train LLMs in a reinforcement learning (RL) setting. AgentRewardBench (Lù et al., 2025) provides a benchmark for assessing automatic evaluators (including LLM judges) to evaluate web agent trajectories, which Lin et al. (2025) extend to the computer-use scenario. Such benchmarks can be used to validate powerful judges for approaches relying on a reward model (Bai et al., 2024; Putta et al., 2024; He et al., 2024).

3 THE AGENT-AS-ANNOTATORS FRAMEWORK

In this section, we introduce the AGENT-AS-ANNOTATORS framework, which we use to study and design systems that enables the collection of high-quality trajectories from frontier models to train student models designed to run locally. We show how each human annotation role corresponds to a LLM-based module (Section 3.1), followed by an overview on how prior trajectory synthesis pipelines can be naturally expressed as instantiations of our framework (Section 3.2).

Table 1: Our Agent-as-annotator enables improvements on WebArena while transferring to 4 out-of-domain benchmarks, with the largest improvement on WorkArena L1 (from 33.3% to 51.5%), an enterprise interface never seen during training. Success rate (+absolute changes) (%); WA = WebArena, VWA = VisualWebArena, WoA = WorkArena (Drouin et al., 2024), W++ = WorkArena++ (Boisvert et al., 2024), WoB = MiniWoB. **Teacher model used to generate* A3-SYNTH.

Model	WA	VWA	WoA L1	W++ L2	WoB
Proprietary					
Gemini 3 Pro*	51.2	49.0	79.7	41.6	74.7
Gemini 3.1 Flash L.	42.3	35.0	58.5	21.1	74.1
Open-weight (base)					
Qwen3.5-27B	41.5	37.4	57.0	18.9	70.9
Qwen3.5-9B	31.0	28.5	33.3	2.2	63.2
Qwen3.5-4B	24.1	24.7	33.6	1.6	61.1
Qwen3.5-2B	3.1	5.3	4.2	0.0	11.8
A3 fine-tuned (ours; teacher: Gemini 3 Pro)					
A3-Qwen3.5-9B	**41.5** (+10.5)	**33.9** (+5.4)	**51.5** (+18.2)	**9.7** (+7.5)	**69.0** (+5.8)
A3-Qwen3.5-4B	35.2 (+11.1)	30.1 (+5.4)	44.8 (+11.2)	3.8 (+2.2)	66.9 (+5.8)
A3-Qwen3.5-2B	9.2 (+6.1)	7.6 (+2.2)	6.7 (+2.5)	0.0	38.6 (+26.8)

3.1 FROM HUMAN ROLES TO LLM MODULES

To create training and evaluation data for web agent tasks, we need to follow a structured human annotation process. We observe that this process involves three functional roles, present in varying degrees across benchmarks. First, a **Task Designer** needs to explore the web environment to gain inspiration (Zhou et al., 2023b). Then, they can start designing task intents (in natural language form) that are grounded in the actual environments, and provide criteria that enable proper evaluation. Given a task formed from a pair of intent and evaluation criteria, we need someone to complete the task in order to demonstrate how the task should be completed. In Mind2Web (Deng et al., 2023) and WebLINX (Lù et al., 2024), an **Annotator** receives the task intent and attempts to complete it in the same environment, producing a trajectory of actions interleaved with web states (screenshots, HTML pages, etc.). Once the trajectories are collected, a **Reviewer** can review the intents, trajectories and evaluation criteria to decide whether the annotator correctly solved the task, in which case the trajectory can be used as a positive sample for training models.

AGENT-AS-ANNOTATORS replaces the Task Design, Annotator and Reviewer with LLM modules, all of which can be powered by the same or several LLMs. We organize the process into two phases mirroring the division of labor described above.

Phase 1: Replacing task designer through structured task synthesis For this phase, the task annotator's job can be broken down into two modules: the prior of the person which will guide how they will approach the exploration (i.e., the **Persona**) and a **Task Generator**, which comes up with relevant tasks based on an exploration and the persona. We design the **Persona Generator** to yield diverse user profiles (background, expertise, goals) before using the structured profiles to generate a narrative description of each persona. This enables different usage patterns within the same environment; for example, *Alice* may be interested in hiking and sci-fi novels, whereas *Liam* is interested in indie music and street art (see C). Then, the **Task Generator** receives a persona, general instructions for annotators (based on WebArena), and access to web environments. The module explores the environment and stores observations in order to synthesize task intents and evaluation hints pairs, while staying grounded in the explored environment states; this avoids hallucinated tasks (for example buying items that may not exist on a shopping site, or accessing imaginary repositories on GitLab).

Phase 2: Replacing annotators and reviewers through trajectory collection and filtering Given only a task intent, a web **Agent** is tasked to solve a task in a controlled environment, which ultimately yields a trajectory. During the process, they cannot access the hints, exploration trajectories, nor the underlying persona. This separation enables the agent to genuinely attempt to solve the task without external information, which is reflected in its trajectory. Then, upon completion, the **Judge** evaluates

Table 2: Characteristics of past works with respect to the AGENT-AS-ANNOTATORS framework. AGENT-AS-ANNOTATORS is the only pipeline that instantiates all six modules. Task: *Ground.* = tasks defined before execution; *Retro.* = labeled post-hoc. Shaded cells = module absent. Individual module contributions are ablated in Appendix B.

Pipeline	Persona	Expl.	Task	Hints	Agent	Judge	Env.
NNetNav	Single-step	✓	Retro.	–	–	LLM	WA+Web
InSTA	–	✓	Ground.	–	✓	LLM	Web
AgentTrek	–	–	Tutorial	–	✓	VLM	Web
Go-Browse	–	✓	Ground.	–	✓	VLM	WA
AGENT-AS-ANNOTATORS	Two-step	✓	Ground.	✓	✓	LLM	WA

the agent by reviewing its trajectory alongside the personas and evaluation hints, enabling it to make a decision on whether the agent is successful; since the hints are directly grounded in the explorations, this step enables the judge to make a decision that connects directly to the reality of the environments. We use this success signal to filter the web trajectories keeping only those deemed successful to train the student (Figure 1).

3.2 PRIOR WORK AS INSTANTIATIONS OF AGENT-AS-ANNOTATORS

In Table 2, we identify five recent web trajectory synthesis methods containing components that correspond to modules in AGENT-AS-ANNOTATORS. Generally, a task is created from three possible approaches: through tutorials fetched online, retroactively labeling the exploration trajectories, and by generating tasks grounded on past explorations of the environments. Only grounded approaches can generate evaluation hints, and only AGENT-AS-ANNOTATORS currently leverage the use of hints for determining the success of trajectories.

- **InSTA** (Trabucco et al., 2025): achieves diversity by scaling to 150K websites (rather than using personas).
- **Go-Browse** (Gandhi & Neubig, 2025): uses graph search over URLs with automatic feasibility checking (through multimodal LLMs) and requires separate solver models.
- **NNetNav** (Murty et al., 2024): samples persona using a LLM, similar to our approach, through a single-step approach. It uses an agent to explore first to collect trajectories, then reassign labels post-hoc using a task labeling model; a later iteration of this work also explores recollecting trajectories from the relabeled tasks.

4 EXPERIMENTAL SETUP

In this section, we describe the A3-SYNTH dataset derived from AGENT-AS-ANNOTATORS (Section 4.1), the training procedure used to improve the local models (Section 4.2), and the evaluation and baselines (Section 4.3).

4.1 A3-SYNTH: HIGH-QUALITY TRAJECTORIES FOR TRAINING LOCAL MODELS

In order to collect a wide range of tasks across a realistic set of web environments, we apply AGENT-AS-ANNOTATORS on six self-hosted environments published by Zhou et al. (2023b): a Reddit-like forum, GitLab, an Amazon-style shopping website site, an e-commerce administration panel, a Wikipedia-like website, and OpenStreetMap (an open alternative to Google Maps). We generate 250 diverse personas using our pipeline and assign each of them to all six environments, yielding 1500 exploration trajectories. We then synthesize two task intents for each exploration (with randomly sampled steps), giving us a total of 3000 tasks. We call our novel dataset of high-quality trajectories A3-SYNTH, which we find to be effective for training small local models despite being substantially more data efficient than datasets from past works.

Teacher model. To instantitate each module, we use Gemini 3 Pro in low-thinking mode; using a single model as teacher allows us to simplify the experiments. By configuring the model to produce fewer reasoning traces, we can reduce the number of output tokens needed by smaller models to

produce an action, which further improves efficiency. We also find that lower thinking achieves comparable or even *higher* success rates on A3-SYNTH tasks across all six environments (Section 5.1). We evaluate alternatives (Gemini 3.1 Pro, Gemini 3 Flash) in Appendix A.1.

Judge. We also implement the judge module using the same model as the teacher, and we further augments past approaches such as Zheng et al. (2023) by providing the persona and evaluation hints. This allows the judge to make an informed decision, since the hints provide relevant information about the expected outcome and help resolve potential ambiguous outcomes (e.g., "the user should see a confirmation message on the settings page" or "the repository should contain a new file named X"). The Judge assesses whether the trajectory successfully completes the task (see Appendix C.4).

Data statistics. After filtering the trajectories through the judge, A3-SYNTH contains 2,322 successful trajectories, totaling 16,353 training examples (pairs of observations and actions), with, on average, 7.04 steps per trajectory and a mean response length of 1,920 characters. Each response contains reasoning trace summaries (averaging 1,021 characters).

4.2 TRAINING A3-QWEN FAMILY OF MODELS

Using our novel dataset, we employ well-established supervised finetuning (SFT) approaches to produce a highly capable web agent. First, the trajectories in A3-SYNTH are converted into standard chat format for supervised finetuning (Ouyang et al., 2022). For each step, we combine the task goal with interleaved observation of the web state (accessibility tree, screenshot) to produce a single user message. Then, the assistant response is formatted to contain the reasoning chain followed by the action (which is parsed into Playwright actions). We fine-tune multimodal models from the Qwen3.5 (Qwen Team, 2026) family ranging from 2B to 9B parameters. We train them for 2 epochs with a learning rate of 1×10^{-5}, a batch size of 32, and the maximum sequence length set to 8192. We apply cross-entropy loss only on the response tokens. The full hyperparameters can be found in Appendix C.6. The resulting family of web agents, A3-Qwen3.5, is released to enable the deployment of local models with substantially improved web navigation capabilities.

4.3 EVALUATION AND BASELINES

To assess the ability of our novel family of web agents, we evaluated them on several web navigation benchmarks using BrowserGym (de Chezelles et al., 2024). First, we evaluated them on the test split of **WebArena** (Zhou et al., 2023b), which consists of 381 tasks in the same six self-hosted web applications used for our training (i.e., in-domain environments), although the evaluation tasks were never seen during training. We further evaluate on **VisualWebArena** (Koh et al., 2024a) (449 test tasks), which extends WebArena to vision-based tasks, **WorkArena** (Drouin et al., 2024) (330 tasks) and **WorkArena++** (Boisvert et al., 2024) (185 test tasks) evaluate on ServiceNow, an enterprise platform completely different from any training environment, and **MiniWoB** (Liu et al., 2018) (625 tasks), which tests across a wide range of simple webpages. Apart from WebArena and parts of VisualWebArena, all environments are *fully out-of-distribution* (in the case of VisualWebArena, the tasks themselves require modalities, making them challenging in a different dimension). All models use a unified token budget of 65,536 tokens for fair comparison. To better understand the impact of our appraoches, we evaluate them against **proprietary** models (Gemini 3.1 Pro, Gemini 3.1 Flash Lite) and **open-weight baselines** from the Qwen family (Qwen Team, 2026; Bai et al., 2025) (Qwen3.5-27B/9B/4B/2B, Qwen3-VL-32B/8B-Thinking). Furthermore, we provide results for 16 additional commercial and research-oriented web agent results in Appendix A with details of the evaluation in Appendix D.2).

5 RESULTS

In this section, we present evaluation results across benchmarks (Section 5.1), and show several ablations of the modules corresponding to instantiations of prior works (Section 5.2).

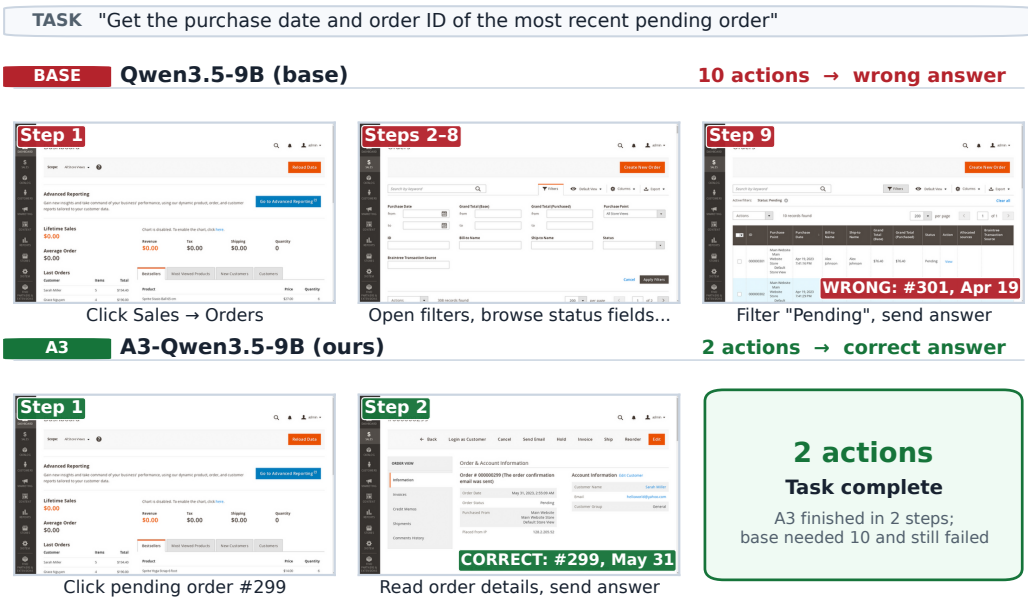

Figure 2: Base vs. fine-tuned model on a WebArena Shopping Admin task. The base Qwen3.5-9B (top, red) opens filters and browses fields for 10 actions before returning the wrong order. A3-Qwen3.5-9B (bottom, green) clicks the correct pending order from the dashboard and answers in 2 actions.

5.1 Evaluation Results

Overall, our family of web agents achieves substantially better performance compared to the base models. Our largest agent, A3-Qwen3.5-9B is finetuned from Qwen3.5-9B on A3-Synth, which enabled it to achieve 41.5% on WebArena's test split and transfer to several out-of-domain benchmarks (see Table 1). Compared to the Qwen3.5-27B, a 3× larger variant of the base model, A3-Qwen-3.5-9B achieves the same SR on WebArena (both 41.5%) and remains competitive on VisualWebArena (33.9% vs. 37.4%), which highlights the capability of the smaller model to learn beyond its regular post-training regime described by Qwen Team (2026). Moreover, compared to the current BrowserGym leaderboard, our approach surpasses most of the proprietary approaches (see Appendix A.5) on WebArena.

Transfer to unseen enterprise interfaces. Despite never seeing enterprise-related tasks or websites during training, A3-Qwen3.5-9B's performance on WorkArena reaches a SR of 51.5%, far ahead of Qwen3.5-9B's 33.3%. The platform behind WorkArena, ServiceNow, contains layouts, forms and navigation workflows that completely differ from WebArena, making it an ideal way to evaluate generalization. On the substantially more challenging WorkArena++ L2, finetuning Qwen3.5-9B on A3-Synth enables it to solve 9.7% of the tasks compared to 2.2% baseline.

Breakdown by scale and environments. When looking at the effect at the environment level (Table 5 in Appendix A.2), we observe that the largest improvement is on GitLab (from 30.7% to 49.5%) and Shopping Admin (from 26.9% to 44.9%), which involve complex form-filling and navigation workflows. At smaller student scales, our A3-Synth dataset provides substantial improvements on WebArena for A3-Qwen3.5-4B (24.1% to 35.2%) and A3-Qwen3.5-2B (3.1% to 9.2%) (Appendix A.4), confirming that the distilled data is useful across model capacities with diminishing returns at 2B. Figure 2 illustrates the difference between the base model and our finetuned counterpart: on a shopping admin task, the base model wanders through filters for 10 steps before returning to the wrong order, while A3-Qwen3.5-9B navigates directly to the correct order in 2 steps. Additional examples can be found in Appendix A.4.

Table 3: Ablations on WebArena (381 tasks). Changes are absolute relative to the first row in each block. The full-scale, 600-trajectory, and 368-trajectory blocks use different trajectory counts; compare rows only within each block.

Ablation	Trajs	Personas	Hints	Reasoning traces	Judge filtering	SR (%)	Change (% absolute)
Full pipeline	2,322	✓	✓	✓	✓	41.5	–
1,430 trajectories	1,430	✓	✓	✓	✓	40.2	−1.3
No Judge filtering	2,999	✓	–	✓	–	37.0	−4.5
No reasoning traces	2,322	✓	✓	–	✓	33.6	−7.9
600 trajectories	600	✓	✓	✓	✓	37.8	–
No hints	600	✓	–	✓	✓	35.4	−2.4
368 trajectories	368	✓	✓	✓	✓	34.1	–
No personas	368	–	✓	✓	✓	25.2	−8.9

Quality of Teacher model across model tiers Using a strong teacher can affect the final performance of the distilled web agent. Gemini 3 Pro achieves 69–85% success rate across environments on A3-SYNTH tasks, which is substantially above Gemini 3 Flash (17–53%). Despite yielding fewer trajectories due to the lower trajectories, Gemini 3 Flash's provides more trajectories steps due to its longer trajectories (22,707 vs. 16,353 for Pro). We note that a similar pattern appears in InSTA (Trabucco et al., 2025), where their 1.7B student is trained on high-quality data, allowing it to outperform its 235B teacher. We compare teacher models by their success on A3-SYNTH Table 4, which guided our decision to use Gemini 3.0 Pro as the primary teacher to generate our dataset.

5.2 ABLATIONS

In table 3, we present the ablations across different modules as well as the effect of data size; we present the results for WebArena's test split (details in Appendix B). We observe that retaining the negative examples flagged by the judge results in a reduced SR, from 41.5% to 37.0%, despite 40% more training data. We further observe that removing reasoning traces reduces SR to 33.6%, indicating the importance of the summarized reasoning traces. Removing the hints that judges receive lowers SR from 37.8% to 35.4% (using 600 trajectories subset for controlled training), while removing personas (while keeping everything else fixed) reduces 34.1% to 25.2% (using the 368-trajectories variants of the training). When reducing the data, we find that the SR scales with more data at a logarithmic scale; increasing trajectory count from 285 to 715 yield a 5% improvement in SR, while the last 892 trajectories only resulted in a 1.3% improvement. Moreover, other ablations following the style of NNetNav and InSTA also show performance drop, which highlights the value of the AGENT-AS-ANNOTATORS framework and the the versatility of A3-SYNTH as a training dataset. We show supplementary results in Table 17 and supporting evidence is shown in Figure 8.

6 DISCUSSION

Generalizing to unseen environments or modalities. An important distinction between AGENT-AS-ANNOTATORS and prior approaches is that we only use six web environments, which is substantially smaller in scope compared to InSTA's 150,000 websites (Trabucco et al., 2025). Despite the small number of environments, we observe strikingly strong generalization capabilities: our flagship finetuned web agent substantially exceed the ability of the original Qwen3.5-9B on unseen enterprise platforms (WorkArena L1, 33.3% → 51.5%), longer-horizon enterprise workflows (WorkArena++ L2, 2.2% → 9.7%), simplified HTML mini-tasks (MiniWoB, 63.2% → 69.0%), and shows an ability to adapt to tasks containing image-based instructions (VisualWebArena, 28.5% → 33.9%). We hypothesize that our persona-driven approach allows a higher diversity of tasks, allowing the model to learn to generalize on a limited number of environments, which is a major departure from InSTA-style approach of exploring a large number of websites without personas.

Future directions. Several extensions could be built on top of our AGENT-AS-ANNOTATORS framework, A3-SYNTH dataset and family of models. By combining our highly environment-efficient

approach with broader environment coverage (leveraging environments from existing benchmarks or open-source projects), our framework lays the foundation for future works to scale our approach while maximizing the usage of each environment. Moreover, the modular nature of AGENT-AS-ANNOTATORS makes it straightforward to change the LLMs inside each module, allowing greater specialization (e.g. using a specialized judge and a dedicated web agent as teacher instead of general-purpose LLMs). Our hint-based grounding can further improve the accuracy of the judge, resulting in better reward signals; combined with our high quality tasks and the sandbox nature of the environments, future work can explore more advanced post-training approaches such as RLVR (Lambert et al., 2025). Finally, A3-SYNTH can be used as part of a mid-training pipeline to train stronger web agents in future works.

7 CONCLUSION

Our motivation to close the gap between frontier LLMs and small open-weight models for web agent tasks lead us to design the AGENT-AS-ANNOTATORS framework. Inspired by the roles humans play when annotating web agent benchmarks, the framework provides a structured way to generate high-quality tasks and trajectories using frontier LLMs as teacher and judge models. Through the process of building the A3-SYNTH dataset, we found that data quality matters more than quantity: 2,322 trajectories produced by a strong teacher and verified by a strong judge enabled us to train a strong Qwen-3.5-9B-derived web agent that surpasses all existing of the same size across five benchmarks, and transfers to enterprise platforms, image-based tasks, and long-horizon workflows that were never seen during training. Through ablations experiments on WebArena, we find that each module contributes towards the final performance of the model. We release the A3-SYNTH dataset, the checkpoints for the family of A3-Qwen web agents, and source code to support future work on making web agents broadly accessible.

REPRODUCIBILITY STATEMENT

Our experiments used AgentLab and BrowserGym for evaluation. The hyperparameters and training details can be found in Section 4.2 and Appendix C.6. We release (1) the full A3-SYNTH trajectory dataset with every teacher configurations, (2) the full source code to reproduce our experiments and framework, (3) the web agent checkpoint (A3-Qwen3.5-2B/4B/9B), and (4) the scripts and configurations used for training and evaluation.

ETHICS STATEMENT

Distilling web agent capabilities into small open-weight models lowers the barrier to deploying autonomous web agents, which can increase the risk for spam and scraping. We will release our work in a license that forbids the use of our models for such purpose. Morever, we generated all trajectories on self-hosted WebArena environments, so we did not generate training trajectories on production websites intended for real users. We encourage the community to develop appropriate safeguards before deploying our agents in production.

AI USE STATEMENT

Gemini models generated synthetic tasks and trajectories and evaluated task success, as described in Section 4 and Appendix C.4. Coding Agents were used to help prepare figures, tables, search for additional relevant papers, and draft the initial version of the paper. It was also used for developing data processing, evaluation code, and experimental analysis. The author wrote the entirety of the final manuscript and verified figures, results and references.

REFERENCES

Moises Andrade, Joonhyuk Cha, Brandon Ho, V. Srihari, Karmesh Yadav, and Z. Kira. Let's think in two steps: Mitigating agreement bias in mllms with self-grounded verification. *ArXiv*, abs/2507.11662, 2025. URL https://arxiv.org/abs/2507.11662.

Hao Bai, Yifei Zhou, M. 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. URL https://arxiv.org/abs/2406.11896.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Léo Boisvert, Megh Thakkar, Maxime Gasse, Massimo Caccia, Thibault Le Sellier de Chezelles, Quentin Cappart, Nicolas Chapados, Alexandre Lacoste, and Alexandre Drouin. Workarena++: Towards compositional planning and reasoning-based common knowledge work tasks. *ArXiv*, abs/2407.05291, 2024. URL https://arxiv.org/abs/2407.05291.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *ArXiv*, abs/1606.01540, 2016. URL https://arxiv.org/abs/1606.01540.

Baian Chen, Chang Shu, Ehsan Shareghi, Nigel Collier, Karthik Narasimhan, and Shunyu Yao. Fireact: Toward language agent fine-tuning. *ArXiv*, abs/2310.05915, 2023. URL https://arxiv.org/abs/2310.05915.

Thibault Le Sellier de Chezelles, Maxime Gasse, Alexandre Lacoste, Alexandre Drouin, Massimo Caccia, Léo Boisvert, Megh Thakkar, Tom Marty, Rim Assouel, Sahar Omidi Shayegan, Lawrence Jang, Xing Han Lù, Ori Yoran, Dehan Kong, Frank F. Xu, Siva Reddy, Quentin Cappart, Graham Neubig, Ruslan Salakhutdinov, and Nicolas Chapados. The browsergym ecosystem for web agent research. *ArXiv*, abs/2412.05467, 2024. URL https://arxiv.org/abs/2412.05467.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *ArXiv*, abs/2306.06070, 2023. URL https://arxiv.org/abs/2306.06070.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, I. Laradji, Manuel Del Verme, Tom Marty, Léo Boisvert, Megh Thakkar, Quentin Cappart, David Vázquez, Nicolas Chapados, and Alexandre Lacoste. Workarena: How capable are web agents at solving common knowledge work tasks? *ArXiv*, abs/2403.07718, 2024. URL https://arxiv.org/abs/2403.07718.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, A. Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony S. Hartshorn, Aobo Yang, Archi Mitra, A. Sravankumar, A. Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aur'elien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, B. Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, C. Touret, Chunyang Wu, C. Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, D. Song, D. Pintz, D. Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab A. AlBadawy, E. Lobanova, Emily Dinan, E. Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, G. Nail, G. Mialon, Guanglong Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, J. V. D. Linde, J. Billock, Jenny Hong, Jenya Lee, J. Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, J. Johnstun, Joshua Saxe, Ju-Qing Jia, Kalyan Vasuden Alwala, K. Upasani, Kate Plawiak, Keqian Li, Kenneth Heafield, Kevin R. Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuen ley Chiu, Kunal Bhalla, Lauren Rantala-Yeary, L. Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat,

Luke de Oliveira, M. Muzzi, Ma hesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Mathew Oldham, Mathieu Rita, Maya Pavlova, M. Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Niko lay Bashlykov, Nikolay Bogoychev, Niladri S. Chatterji, Olivier Duchenne, Onur cCelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasić, Peter Weng, Prajjwal Bhargava, P. Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Q. He, Qingxiao Dong, Ragavan Srinivasan, R. Ganapathy, Ramon Calderer, Ricardo Sil veira Cabral, Robert Stojnic, R. Raileanu, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ron nie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, S. Hosseini, Sa hana Chennabasappa, Sanjay Singh, Sean Bell, S. Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, S. Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, S. Collot, Suchin Gururangan, S. Borodinsky, Tamar Herman, T. Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whit ney Meers, X. Martinet, Xiaodong Wang, X. Tan, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yiqian Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zhengxu Yan, Zhengxing Chen, Zoe Papakipos, Aaditya K. Singh, Aaron Grattafiori, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adi Gangidi, Adolfo Victoria, Ahuva Goldstand, A. Menon, Ajay Sharma, Alex Boesenberg, Alex Vaughan, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Anam Yunus, Andrei Lupu, Andres Alvarado, A. Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Franco, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, B. Leonhardi, Po-Yao (Bernie) Huang, Beth Loyd, Beto de Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, B. Ni, Braden Hancock, Bram Wasti, Brandon Spence, B. Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Damon Civin, D. Beaty, Daniel Kreymer, Shang-Wen Li, Danny Wyatt, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Erik Brinkman, E. Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, F. Kreuk, Feng Tian, Firat Ozgenel, Francesco Caggioni, Francisco (Paco) Guzmán, Frank J. Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Govind Thattai, Grant Herman, Grigory Sizov, Guangyi Zhang, G. Lakshminarayanan, Hamid Shojanazeri, Han Zou, Hannah Wang, Han Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Igor Molybog, Igor Tufanov, Irina-Elena Veliche, Itai Gat, J. Weissman, James Geboski, James Kohli, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, J. Chan, Jenny Zhen, J. Reizenstein, J. Teboul, J. Zhong, Jian Jin, Jingyi Yang, Joe Cummings, J. Carvill, Jon Shepard, J. McPhie, J. Torres, Josh Ginsburg, Junjie Wang, Kaixing(Kai) Wu, U. KamHou, Karan Saxena, Karthik Prasad, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, K. Veeraraghavan, Kelly Michelena, Keqian Li, Kun Huang, Kunal Chawla, Kushal Lakhotia, Kyle Huang, Lailin Chen, Lakshya Garg, A. Lavender, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, M. Bhatt, M. Tsimpoukelli, Martynas Mankus, Matan Hasson, M. Lennie, Matthias Reso, M. Groshev, M. Naumov, Maya Lathi, Meghan Keneally, M. Seltzer, Michal Valko, Michelle Restrepo, M. Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, M. Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Mun ish Bansal, N. Santhanam, Natascha Parks, Natasha White, Navy ata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikolay Pavlovich Laptev, Ning Dong, Ning Zhang, Norman Cheng, Oleg Chernoguz, O. Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pe dro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollár, Polina Zvyagina, Prashant Ratanchandani, P. Yuvraj, Qian Liang, Rachad Alao, R. Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, R. Maheswari, Russ Howes, Ruty Rinott, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, S. Sidorov, S. Pan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Zha, S. Shankar, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, S. Gupta, Sung-Bae Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Kohler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy

Chou, Tzook Shaked, Varun Vontimitta, Victoria O Ajayi, Victoria Montanez, Vijai Mohan, Vinay Kumar, Vishal Mangla, Vlad Ionescu, V. Poenaru, Vlad T. Mihailescu, V. Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, W. Bouaziz, W. Constable, Xia Tang, Xiaofang Wang, Xiaojian Wu, Xiaolan Wang, Xide Xia, Xilun Wu, Xinbo Gao, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu Wang, Yuchen Hao, Yundi Qian, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, and Zhiwei Zhao. The llama 3 herd of models. 2024. URL https://arxiv.org/abs/2407.21783.

Apurva Gandhi and Graham Neubig. Go-browse: Training web agents with structured exploration. *ArXiv*, abs/2506.03533, 2025. URL https://arxiv.org/abs/2506.03533.

Tao Ge, Xin Chan, Xiaoyang Wang, Dian Yu, Haitao Mi, and Dong Yu. Scaling synthetic data creation with 1,000,000,000 personas. *ArXiv*, abs/2406.20094, 2024. URL https://arxiv.org/abs/2406.20094.

Suriya Gunasekar, Yi Zhang, J. Aneja, C. C. T. Mendes, A. Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, A. Salim, S. Shah, Harkirat Singh Behl, Xin Wang, Sébastien Bubeck, Ronen Eldan, A. Kalai, Y. Lee, and Yuan-Fang Li. Textbooks are all you need. *ArXiv*, abs/2306.11644, 2023. URL https://arxiv.org/abs/2306.11644.

Yuyu Guo, Wenjie Yang, Siyuan Yang, Ziyang Liu, Cheng Chen, Yuan Wei, Yun Hu, Yangru Huang, Guo Hao, Dongsheng Yuan, Jianming Wang, Xin Chen, Hang Yu, Lei Lei, and Peng Di. Opagent: Operator agent for web navigation. *ArXiv*, abs/2602.13559, 2026. URL https://arxiv.org/abs/2602.13559.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Hongming Zhang, Tianqing Fang, Zhenzhong Lan, and Dong Yu. Openwebvoyager: Building multimodal web agents via iterative real-world exploration, feedback and optimization. pp. 27545–27564, 2024. URL https://arxiv.org/abs/2410.19609.

Cheng-Yu Hsieh, Chun-Liang Li, Chih-Kuan Yeh, Hootan Nakhost, Yasuhisa Fujii, Alexander J. Ratner, Ranjay Krishna, Chen-Yu Lee, and Tomas Pfister. Distilling step-by-step! outperforming larger language models with less training data and smaller model sizes. *ArXiv*, abs/2305.02301, 2023. URL https://arxiv.org/abs/2305.02301.

Seungone Kim, Jamin Shin, Yejin Cho, Joel Jang, S. Longpre, Hwaran Lee, Sangdoo Yun, Seongjin Shin, Sungdong Kim, James Thorne, and Minjoon Seo. Prometheus: Inducing fine-grained evaluation capability in language models. *ArXiv*, abs/2310.08491, 2023. URL https://arxiv.org/abs/2310.08491.

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

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

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, Noah A. Smith, and Hannaneh Hajishirzi. Rewardbench: Evaluating reward models for language modeling, 2024. URL https://arxiv.org/abs/2403.13787.

Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V. Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, Yuling Gu, Saumya Malik, Victoria Graf, Jena D. Hwang, Jiangjiang Yang, Ronan Le Bras, Oyvind Tafjord, Chris Wilhelm, Luca Soldaini, Noah A. Smith, Yizhong Wang, Pradeep Dasigi, and Hannaneh Hajishirzi. Tulu 3: Pushing frontiers in open language model post-training, 2025. URL https://arxiv.org/abs/2411.15124.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. A diversity-promoting objective function for neural conversation models. In Kevin Knight, Ani Nenkova, and Owen

Rambow (eds.), *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 110–119, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-1014. URL https://aclanthology.org/N16-1014/.

Haojia Lin, Xiaoyu Tan, Yulei Qin, Zihan Xu, Yuchen Shi, Zongyi Li, Gang Li, Shaofei Cai, Siqi Cai, Chaoyou Fu, Ke Li, and Xing Sun. Cuarewardbench: A benchmark for evaluating reward models on computer-using agent, 2025. URL https://arxiv.org/abs/2510.18596.

E. Liu, Kelvin Guu, Panupong Pasupat, Tianlin Shi, and Percy Liang. Reinforcement learning on web interfaces using workflow-guided exploration. *ArXiv*, abs/1802.08802, 2018. URL https://arxiv.org/abs/1802.08802.

Dunjie Lu, Shuai Bai, Tianyi Bai, Sicheng Fan, Chang Gao, Jian Guan, Feng Hu, Mianqiu Huang, Xingyang Huang, Yizhen Jiang, Yuheng Jing, Dehui Kong, Ning Li, Dayiheng Liu, Shixuan Liu, Zheng Liu, Que Shen, Bowen Wang, Junli Wang, Chencan Wu, Rui Xie, Tianbao Xie, Zhihui Xie, Haiyang Xu, An Yang, Tao Yu, Wenzhen Yuan, Xi Zhang, Zhenru Zhang, Mingkang Zhu, Zhaoqing Zhu, Yizhong Cao, Kai Dang, Binyuan Hui, Kaixin Li, Junyang Lin, Haiquan Wang, Zekun Wang, Yiheng Xu, Fan Yan, Mengqi Yuan, Danyang Zhang, Jiajun Zhang, Zhipeng Zhang, Fan Zhou, and Fan Zhou. Qwen-cua: Native computer use for (almost) everything, 2026. URL https://arxiv.org/abs/2608.02352.

Xing Han Lù, Zdeněk Kasner, and Siva Reddy. Weblinx: Real-world website navigation with multi-turn dialogue. pp. 33007–33056, 2024. URL https://arxiv.org/abs/2402.05930.

Xing Han Lù, Amirhossein Kazemnejad, Nicholas Meade, Arkil Patel, Dongchan Shin, Alejandra Zambrano, Karolina Stańczak, Peter Shaw, Christopher Pal, and Siva Reddy. Agentrewardbench: Evaluating automatic evaluations of web agent trajectories. *ArXiv*, abs/2504.08942, 2025. URL https://arxiv.org/abs/2504.08942.

Subhabrata Mukherjee, Arindam Mitra, Ganesh Jawahar, Sahaj Agarwal, Hamid Palangi, and A. Awadallah. Orca: Progressive learning from complex explanation traces of gpt-4. *ArXiv*, abs/2306.02707, 2023. URL https://arxiv.org/abs/2306.02707.

Shikhar Murty, Dzmitry Bahdanau, and Christopher D. Manning. Nnetnav: Unsupervised learning of browser agents through environment interaction in the wild. 2024. URL https://arxiv.org/abs/2410.02907.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, S. Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke E. Miller, M. Simens, Amanda Askell, Peter Welinder, P. Christiano, Jan Leike, and Ryan J. Lowe. Training language models to follow instructions with human feedback. *ArXiv*, abs/2203.02155, 2022. URL https://arxiv.org/abs/2203.02155.

Vardaan Pahuja, Yadong Lu, Corby Rosset, Boyu Gou, Arindam Mitra, Spencer Whitehead, Yu Su, and Ahmed Awadallah. Explorer: Scaling exploration-driven web trajectory synthesis for multimodal web agents. In *Annual Meeting of the Association for Computational Linguistics*, pp. 6300–6323, 2025. URL https://arxiv.org/abs/2502.11357.

Emiliano Penaloza, Dheeraj Vattikonda, Nicolas Gontier, Alexandre Lacoste, Laurent Charlin, and Massimo Caccia. Privileged information distillation for language models. *ArXiv*, abs/2602.04942, 2026. URL https://arxiv.org/abs/2602.04942.

Pranav Putta, Edmund Mills, Naman Garg, S. 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://arxiv.org/abs/2408.07199.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Xinyue Yang, Jiadai Sun, Yu Yang, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. *ArXiv*, abs/2411.02337, 2024. URL https://arxiv.org/abs/2411.02337.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Christopher Rawles, Sarah Clinckemaillie, Yifan Chang, J. Waltz, G. Lau, Marybeth Fair, Alice Li, Will Bishop, Wei Li, Folawiyo Campbell-Ajala, Daniel Toyama, Robert Berry, Divya Tyamagundlu, Timothy P. Lillicrap, and O. Riva. Androidworld: A dynamic benchmarking environment for autonomous agents. *ArXiv*, abs/2405.14573, 2024. URL https://arxiv.org/abs/2405.14573.

Gabriel Sarch, Snigdha Saha, Naitik Khandelwal, Ayush Jain, Michael J. Tarr, Aviral Kumar, and Katerina Fragkiadaki. Grounded reinforcement learning for visual reasoning. *ArXiv*, abs/2505.23678, 2025. URL https://arxiv.org/abs/2505.23678.

Junhong Shen, Hao Bai, Lunjun Zhang, Yifei Zhou, Amrith Rajagopal Setlur, Shengbang Tong, Diego Caples, Nan Jiang, Tong Zhang, Ameet Talwalkar, and Aviral Kumar. Thinking vs. doing: Agents that reason by scaling test-time interaction. *ArXiv*, abs/2506.07976, 2025. URL https://arxiv.org/abs/2506.07976.

Tianlin Shi, A. Karpathy, Linxi (Jim) Fan, J. Hernández, and Percy Liang. World of bits: An open-domain platform for web-based agents. pp. 3135–3144, 2017. URL https://www.semanticscholar.org/paper/298a55ddc9777e39c5bad92a750827e1cae98ac1.

Brandon Trabucco, Gunnar Sigurdsson, Robinson Piramuthu, and Ruslan Salakhutdinov. Towards internet-scale training for agents. *ArXiv*, abs/2502.06776, 2025. URL https://arxiv.org/abs/2502.06776.

Lei Wang, Chengbang Ma, Xueyang Feng, Zeyu Zhang, Hao ran Yang, Jingsen Zhang, Zhi-Yang Chen, Jiakai Tang, Xu Chen, Yankai Lin, Wayne Xin Zhao, Zhewei Wei, and Ji rong Wen. A survey on large language model based autonomous agents. *Frontiers of Computer Science*, 18, 2023. URL https://arxiv.org/abs/2308.11432.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions. pp. 13484–13508, 2022. URL https://arxiv.org/abs/2212.10560.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, F. Xia, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903, 2022. URL https://arxiv.org/abs/2201.11903.

Zhepei Wei, Wenlin Yao, Yao Liu, Weizhi Zhang, Qin Lu, Liang Qiu, Changlong Yu, Puyang Xu, Chao Zhang, Bing Yin, Hyokun Yun, and Lihong Li. Webagent-r1: Training web agents via end-to-end multi-turn reinforcement learning. pp. 7909–7928, 2025. URL https://arxiv.org/abs/2505.16421.

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

Kevin Xu, Yeganeh Kordi, Tanay Nayak, Adi Asija, Yizhong Wang, Kate Sanders, Adam Byerly, Jingyu Zhang, Benjamin Van Durme, and Daniel Khashabi. Turkingbench: A challenge benchmark for web agents. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 3694–3710, 2025.

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. URL https://arxiv.org/abs/2412.09605.

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

Qwen An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxin Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yi-Chao Zhang, Yunyang Wan, Yuqi Liu, Zeyu Cui, Zhenru Zhang, Zihan Qiu, Shanghaoran Quan, and Zekun Wang. Qwen2.5 technical report. *ArXiv*, abs/2412.15115, 2024. URL https://arxiv.org/abs/2412.15115.

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

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *ArXiv*, abs/2210.03629, 2022b. URL https://arxiv.org/abs/2210.03629.

Aohan Zeng, Mingdao Liu, Rui Lu, Bowen Wang, Xiao Liu, Yuxiao Dong, and Jie Tang. Agenttuning: Enabling generalized agent abilities for llms. *ArXiv*, abs/2310.12823, 2023. URL https://arxiv.org/abs/2310.12823.

Yanli Zhao, A. Gu, R. Varma, Liangchen Luo, Chien chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, Alban Desmaison, Can Balioglu, Bernard Nguyen, Geeta Chauhan, Yuchen Hao, and Shen Li. Pytorch fsdp: Experiences on scaling fully sharded data parallel. *Proc. VLDB Endow.*, 16:3848–3860, 2023. URL https://arxiv.org/abs/2304.11277.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, E. Xing, Haotong Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging llm-as-a-judge with mt-bench and chatbot arena. *ArXiv*, abs/2306.05685, 2023. URL https://arxiv.org/abs/2306.05685.

Chunting Zhou, Pengfei Liu, Puxin Xu, Srini Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, L. Yu, Susan Zhang, Gargi Ghosh, M. Lewis, Luke Zettlemoyer, and Omer Levy. Lima: Less is more for alignment. *ArXiv*, abs/2305.11206, 2023a. URL https://arxiv.org/abs/2305.11206.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents. *ArXiv*, abs/2307.13854, 2023b. URL https://arxiv.org/abs/2307.13854.

A Extended Results and Analysis

A.1 A3-Synth Teacher Model Results

In Table 4, we report the success rates reported by the judge module (which is the same model as the agent, thus self-reporting) and observation-action pairs retrieved from successful trajectories, which we use for training. We test several teachers, which all try to solve 500 tasks for each website in WebArena (for a total of 3000).

Among the teachers, Gemini 3 Pro with reduced thinking has the highest success rate on all six sites. Gemini 3 Flash yields a greater number of training examples but lower task success.

When controlling for the number of total trajectories, we found that training an earlier variant of our final model, Qwen3-VL-8B-Thinking, using Gemini 3 Pro as teacher achieves 36.2% success rate on the final WebArena test set, whereas Gemini 3 Flash as a teacher only achieves 24.9%. This motivates our final choice of using Gemini 3 Pro as the teacher. We switch to a stronger base model later, but avoid rerunning the experiment due to the elevated costs.

Table 4: Gemini 3 Pro (reduced thinking) achieves the highest success rates across all environments, though Flash produces more total training examples (22,707 vs. 16,353) from lower-quality trajectories. Extended teacher model comparison on A3-SYNTH: success rate (%) by site and total training examples produced.

Teacher	Reddit	GitLab	Shopping	Shop Adm.	Wikipedia	Map	Examples
Gemini 3 Pro							
Pro (reduced thinking)	**69.0**	**70.6**	**80.8**	**80.8**	**85.4**	**78.0**	**16,353**
Pro	66.0	64.2	68.8	78.2	78.8	74.4	15,351
Gemini 3.1 Pro							
3.1 Pro (reduced thinking)	58.4	67.6	61.4	63.8	51.0	45.4	14,859
Gemini 3 Flash							
Flash (high thinking)	45.8	33.0	39.4	39.4	35.6	17.0	22,707
Flash (reduced thinking)	45.8	35.6	40.2	38.4	33.6	16.4	10,943

A.2 PER-BENCHMARK DETAILED RESULTS

We report in table 1 and figure 3 detailed results across both in-domain and out-of-domain benchmarks, focusing on BrowserGym-defined test splits for WebArena (381 tasks), VisualWebArena (449), and WorkArena++ L2 (185), and full task sets for WorkArena L1 (330 episodes) and MiniWoB (625) (de Chezelles et al., 2024). We identify the full-benchmark results whenever available in the table captions, and tables 5–9 give the breakdown per site or per domain for each benchmark.

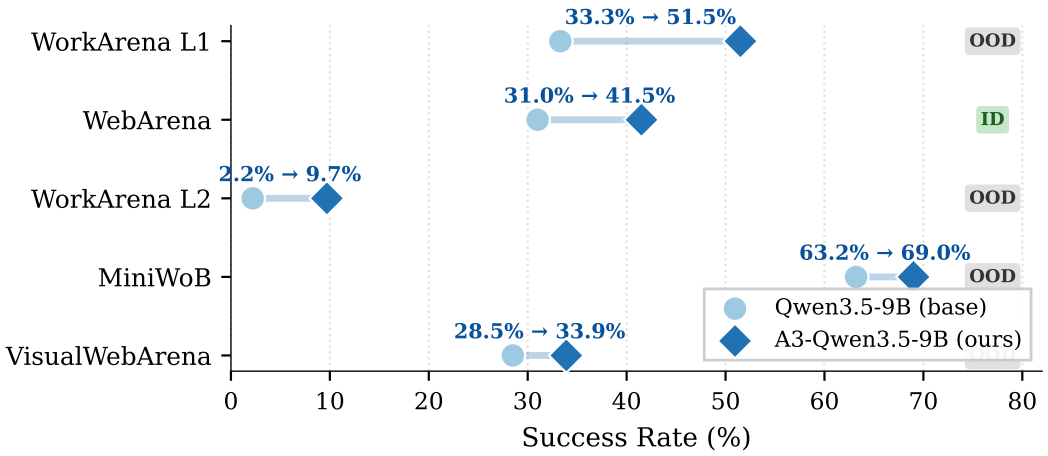

Figure 3: Cross-benchmark success rates for Qwen3.5-9B before and after fine-tuning on A3-SYNTH Pro (reduced thinking) trajectories. The largest gain is on WorkArena L1 (from 33.3% to 51.5%), an enterprise interface never seen during training. All BrowserGym benchmarks except WebArena are fully out-of-distribution.

Table 5: Fine-tuned Qwen3.5-9B (41.5%) matches the 3× larger Qwen3.5-27B on WebArena, with the largest gains on GitLab (from 30.7% to 49.5%) and Shopping Admin (from 26.9% to 44.9%). Per-site success rates and absolute changes (%); **bold** = best open-weight ≤9B.

Model	Reddit	GitLab	Shopping	Shop Admin	Map	All
Proprietary frontier models						
Gemini 3.1 Pro	79.2	57.4	43.3	55.1	42.2	53.8
Gemini 3 Pro	75.0	56.4	34.4	53.8	45.3	51.2
Gemini 3.1 Flash Lite	68.8	48.5	32.2	41.0	28.1	42.3
Open-weight models (base)						
Qwen3.5-27B	56.2	44.6	35.6	46.2	28.1	41.5
Qwen3.5-9B	54.2	30.7	30.0	26.9	20.3	31.0
Qwen3.5-4B	33.3	23.8	24.4	28.2	12.5	24.1
Qwen3.5-2B	4.2	2.0	7.8	1.3	0.0	3.1
A3 fine-tuned (ours)						
A3-Qwen3.5-9B	**56.2** +2.0	**49.5** +18.8	**33.3** +3.3	**44.9** +18.0	**25.0** +4.7	**41.5** +10.5
A3-Qwen3.5-4B	50.0 +16.7	39.6 +15.8	32.2 +7.8	30.0 +1.8	26.4 +13.9	35.2 +11.1
A3-Qwen3.5-2B	6.2 +2.0	6.9 +4.9	14.4 +6.6	7.5 +6.2	11.3 +11.3	9.2 +6.1

Table 6: VisualWebArena results. Qwen3.5-9B and A3-Qwen3.5-9B are evaluated on the full 910 tasks (test + train); other models use the test split only (449 tasks, marked †). Success rates and absolute changes (%) by site. Cross-site tasks are grouped by starting site.

Model	Classifieds	Shopping	Reddit	All
Proprietary frontier models				
Gemini 3.1 Pro	47.4	51.9	39.0	47.9
Gemini 3 Pro	51.7	49.8	44.0	49.0
Gemini 3 Flash	37.9	40.8	35.0	38.8
Gemini 3.1 Flash Lite	31.9	36.9	34.0	35.0
Open-weight models (base)				
Qwen3.5-27B	37.9	39.9	31.0	37.4
Qwen3.5-9B†	22.4	32.6	20.0	26.2
Qwen3-VL-8B-Thinking	25.0	30.0	16.0	25.6
Qwen3.5-4B	29.8	27.6	16.5	24.7
Qwen3.5-2B	5.3	6.1	5.1	5.3
A3 fine-tuned (ours)				
A3-Qwen3.5-9B†	**35.8** +13.4	**35.2** +2.6	**34.1** +14.1	**33.7** +7.5
A3-Qwen3.5-4B	28.1 -1.7	34.6 +7.0	26.6 +10.1	**30.1** +5.4
A3-Qwen3.5-2B	7.9 +2.6	8.3 +2.2	6.3 +1.2	**7.6** +2.2

Table 7: A3 fine-tuning produces the largest gain on WorkArena L1 (from 33.3% to 51.5%), with the biggest per-category jump on Order tasks (from 57.8% to 96.7%). Success rates and absolute changes (%) by task category; Crt = Create (50), Flt = Filter (60), Srt = Sort (60), Ord = Order (90), Cht = Chart (40), Nav = Navigation (30).

Model	Create	Filter	Sort	Order	Chart	Navigate	All
Proprietary frontier models							
Gemini 3.1 Pro	70.0	33.3	86.7	98.9	90.0	100	79.4
Gemini 3 Pro	72.0	35.0	88.3	100	82.5	100	79.7
Gemini 3.1 Flash Lite	52.0	31.7	10.0	94.4	70.0	96.7	58.5
Open-weight models (base)							
Qwen3.5-27B	60.0	13.3	11.7	95.6	70.0	96.7	57.0
Qwen3-VL-8B-Thinking	40.0	1.7	10.0	83.3	57.5	60.0	43.3
Qwen3.5-4B	38.0	1.7	11.7	45.6	60.0	63.3	33.6
Qwen3.5-9B	32.0	1.7	13.3	57.8	42.5	53.3	33.3
Qwen3.5-2B	0.0	0.0	11.7	1.1	12.5	3.3	4.2
A3 fine-tuned (ours)							
A3-Qwen3.5-9B	**46.0** +14.0	1.7	**20.0** +6.7	**96.7** +38.9	**65.0** +22.5	**70.0** +16.7	**51.5** +18.2
A3-Qwen3.5-4B	40.0 +2.0	0.0 -1.7	11.7	92.2 +46.6	57.5 -2.5	50.0 -13.3	**44.8** +11.2
A3-Qwen3.5-2B	0.0	0.0	0.0 -11.7	4.4 +3.3	32.5 +20.0	16.7 +13.4	**6.7** +2.5

Table 8: A3 fine-tuning yields a 3.7× relative improvement on WorkArena++ L2 (2.9% → 10.6%), with gains concentrated on Filter tasks. Dashboard tasks (n=116) are 0% for all models. Success rates and absolute changes (%) by category on the full benchmark (341 tasks); Dsh = Dashboard (116), Flt = Filter (75), Nav = Navigate (26), Inf = Infeasible (52), Oth = Other (72).

Model	Dashboard	Filter	Navigate	Infeasible	Other	All
Proprietary frontier models						
Gemini 3.1 Pro	0.0	76.2	53.8	53.8	29.8	40.0
Gemini 3 Pro	0.0	81.0	53.8	50.0	40.0	41.6
Gemini 3.1 Flash Lite	0.0	69.0	30.8	11.5	6.4	21.1
Open-weight models (base)						
Qwen3.5-27B	0.0	31.0	15.4	50.0	10.6	18.9
Qwen3.5-9B	0.0	5.3	11.5	5.8	0.0	2.9
Qwen3-VL-8B-Thinking	0.0	0.0	0.0	7.7	0.0	1.1
Qwen3.5-4B	0.0	4.8	7.7	0.0	0.0	1.6
Qwen3.5-2B	0.0	0.0	0.0	0.0	0.0	0.0
A3 fine-tuned (ours)						
A3-Qwen3.5-9B	0.0	**34.7** +29.4	**26.9** +15.4	0.0	4.2 +4.2	**10.6** +7.6
A3-Qwen3.5-4B	0.0	7.1 +2.3	30.8 +23.1	0.0	0.0	**3.8** +2.2
A3-Qwen3.5-2B	0.0	0.0	0.0	0.0	0.0	0.0

Table 9: A3 fine-tuning improves MiniWoB from 63.2% to 69.0%, with Type rising from 61.0% to 72.4% and Workflow from 77.6% to 88.2%. Success rates and absolute changes (%) by category; Clk = Click (160), Typ = Type (123), Drg = Drag (55), Wkf = Workflow (85), Wdg = Widget (59), Cmp = Comprehend (52), Rsn = Reason (91).

Model	Click	Type	Drag	Wkfl	Widget	Compr.	Reason	All
Proprietary frontier models								
Gemini 3.1 Pro	87.5	84.6	16.4	98.8	83.1	71.2	64.8	77.1
Gemini 3 Pro	83.8	83.2	16.4	91.8	81.7	70.9	63.5	74.7
Gemini 3.1 Flash Lite	85.0	79.7	7.3	95.3	79.7	73.1	64.8	74.1
Open-weight models (base)								
Qwen3.5-9B	80.0	61.0	1.8	77.6	55.9	61.5	65.9	63.2
Qwen3.5-4B	78.8	52.8	5.5	74.1	61.0	57.7	64.8	61.1
Qwen3.5-2B	18.8	4.0	0.0	15.7	0.0	5.7	11.8	11.8
A3 fine-tuned (ours)								
A3-Qwen3.5-9B	**81.2** +1.2	**72.4** +11.4	**5.5** +3.7	**88.2** +10.6	**62.7** +6.8	**63.5** +2.0	**70.3** +4.4	**69.0** +5.8
A3-Qwen3.5-4B	82.5 +3.7	68.8 +16.0	1.8 -3.7	88.2 +14.1	63.3 +3.3	63.6 +3.6	60.0 -4.7	**66.9** +5.8
A3-Qwen3.5-2B	44.1 +25.3	20.0 +16.0	0.0	30.0 +14.3	60.0 +60.0	31.4 +25.7	40.0 +28.2	**38.6** +26.8

A.3 ONLINE-MIND2WEB

In addition to the primary benchmarks described in the main paper, we also explored how well our proposed web agents would perform on the open internet. Therefore, we evaluate on Online-Mind2Web, using the May 2026 snapshot of 300 tasks across 136 live websites; we note that none of the websites were used for training (Xue et al., 2025; Deng et al., 2023). Each model receives accessibility trees and screenshots, and respect a 30-step limit. They were evaluated using the recommended WebJudge-7B developed by the author specifically for the benchmark. We count failed or missing trajectories as failure (reward of 0). Given the costs of running on the open internet, we focus on fewer and lightweight comparisons: Gemini 3.1 Flash Lite as the light proprietary alternative to Gemini Pro; Qwen3.5-9B and A3-Qwen3.5-9B as the respective base and finetuned variants.

Table 10: Online-Mind2Web results using WebJudge-7B. Fine-tuning increases success from 46.3% to 53.7% across 300 live-website tasks.

Model	Success rate
Qwen3.5-9B	46.3% (139/300)
A3-Qwen3.5-9B	**53.7% (161/300)**
Gemini 3.1 Flash Lite	62.7% (188/300)

A.4 QUALITATIVE COMPARISON: BASE VS. FINE-TUNED

In this section we present several scenarios (inside figures) that compare successful trajectories from our A3-Qwen3.5-9B with Qwen3.5-9B (non-finetuned) baseline, which fails on the same tasks. We note that those were selected to specifically analyze the cases where A3 succeeds over the baseline. We note that there are also instances where the base model succeeds but our finetuned model does not; this is not the focus of this section.

In Figure 2, we show a shopping admin task where A3 finds the correct order in two actions. However, the base model selected the incorrect order ID after ten steps. In subsequent figures we also find similar outcomes on different benchmarks: figure 4 shows A3 succeed on a hardware ordering task on ServiceNow, figure 5 highlights A3 successfully identifying the warranty expiration date of a laptop, Figure 6 finds that A3 successfully writes a comment on a Reddit-style forum, and Figure 7 shows that A3 properly selects and submit the requested time; in each scenario the base model fails at a certain stage of the task.

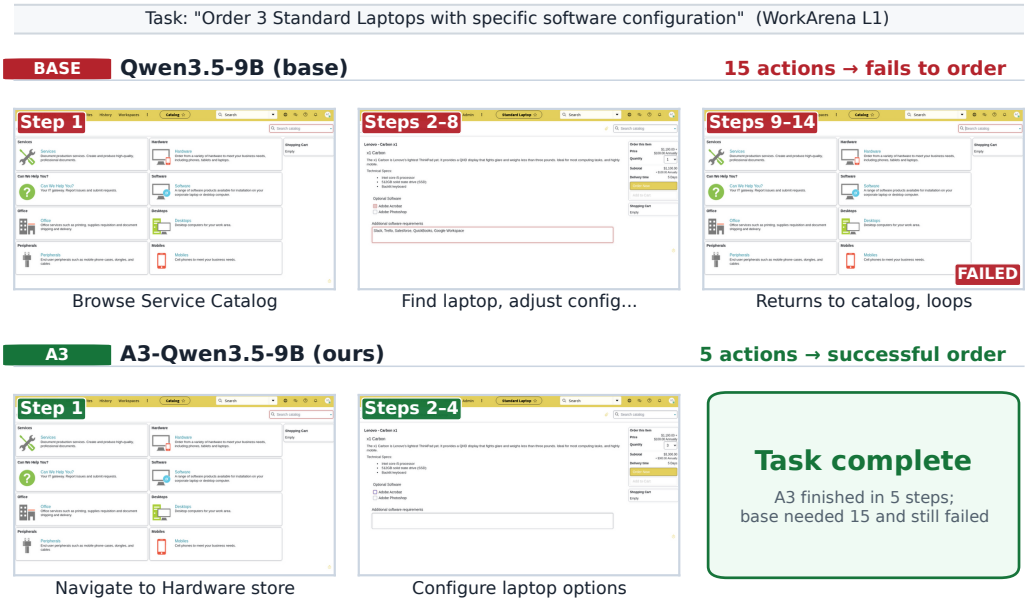

Figure 4: Base vs. A3 model on WorkArena L1 (order Standard Laptop). The base model (top, red) finds the product page but fails to configure and submit the order within 15 actions. The A3 model (bottom, green) efficiently fills in all required fields and places the order in 5 actions.

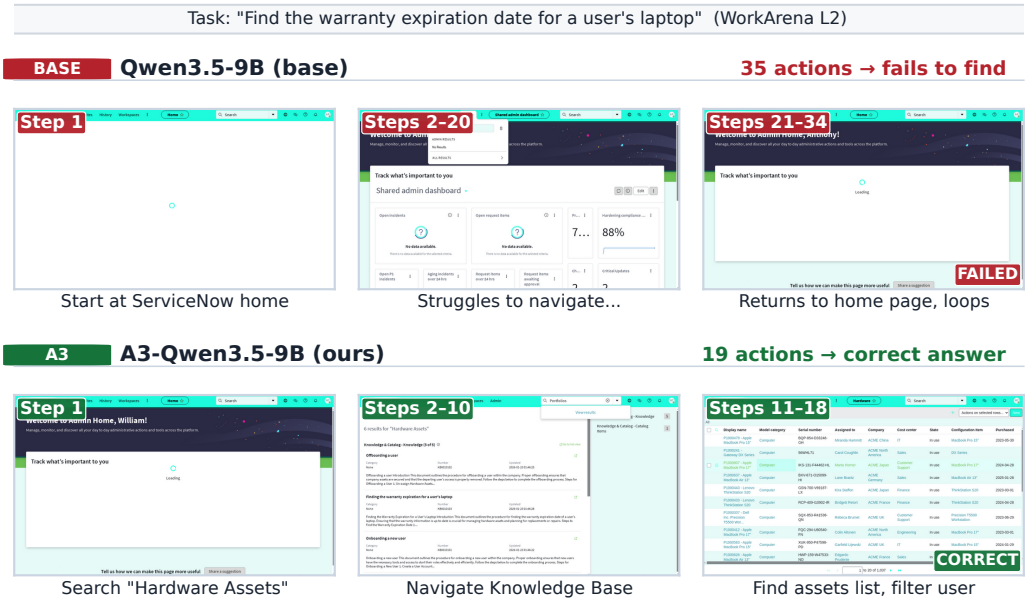

Figure 5: Base vs. A3 model on WorkArena L2 (find warranty expiration date). The base model (top, red) fails to navigate to the hardware assets module within 35 actions. The A3 model (bottom, green) navigates through the knowledge base, finds the assets list, filters by user, and reports the correct warranty date.

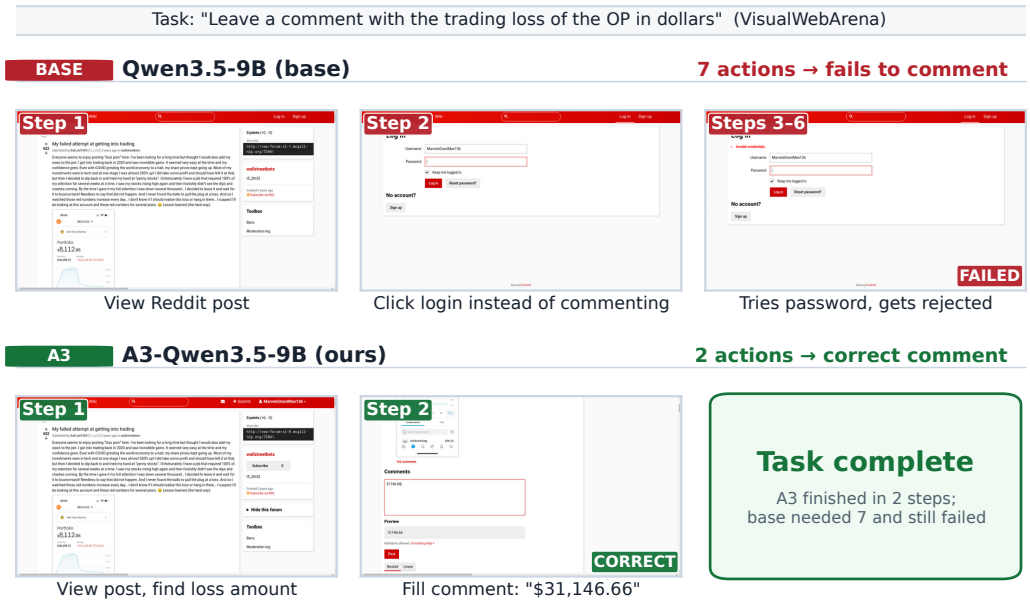

Figure 6: Base vs. A3 model on VisualWebArena task 378 (Reddit comment). The base model (top, red) clicks login and fails authentication for 7 actions. The A3 model (bottom, green) directly types the trading loss amount into the comment box and submits in 2 actions.

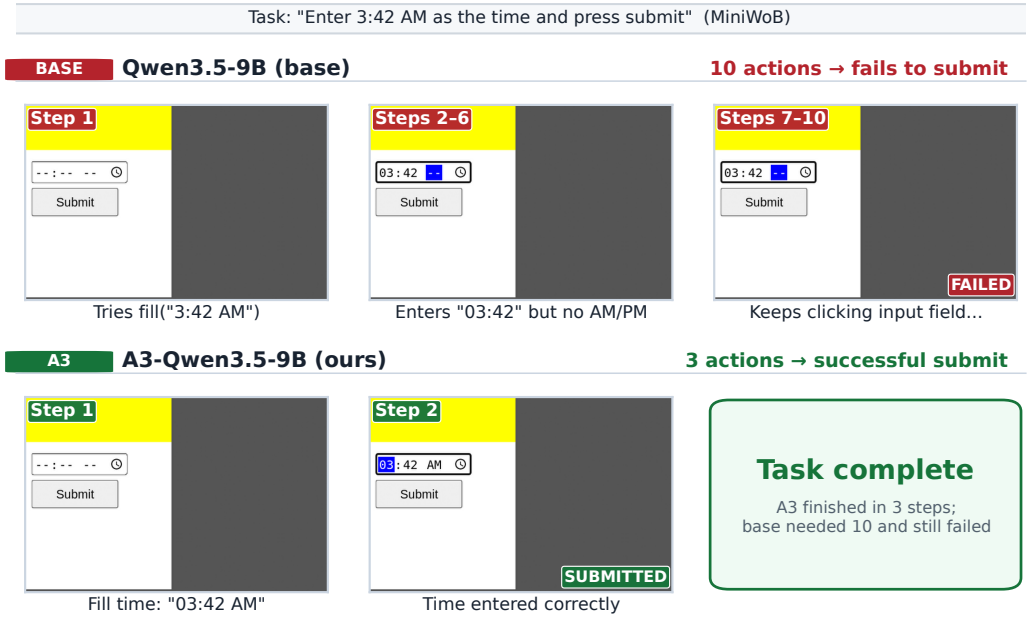

Figure 7: Base vs. A3 model on MiniWoB (enter time). The base model (top, red) repeatedly clicks the input field for 10 actions without entering the time correctly. The A3 model (bottom, green) fills in "03:42 AM" in the correct format and submits in 3 actions.

Scale Interactions We show in Table 5 that our fine-tuning helps improve the 9B, 4B, and 2B variants of Qwen3.5 from 31.0% to 41.5%, 24.1% to 35.2%, and 3.1% to 9.2%, respectively on WebArena. Our fine-tuned 9B web agent matches the 27B variant of Qwen3.5, which highlights the efficiency of our web agent given a similar level of capabilities.

A.5 Comparison with BrowserGym Ecosystem Results

In table 11, we compare A3 with the trajectories[1] publicly released in the AgentLab project (de Chezelles et al., 2024). They use the same *GenericAgent* harness, which includes accessibility tree observations, screenshots, actions, and evaluate with the same script-based or automatic evaluators released in the BrowserGym framework. We use the same test splits, but WorkArena L2 is restricted to the 147 tasks shared across benchmark versions since they do not report the full split consistently.

Table 11: Comparison with BrowserGym ecosystem results using the same agent harness and evaluation protocol. A3-Qwen3.5-9B exceeds last-generation frontier models (GPT-4o and Claude 3.5 Sonnet) and up-to-date open models on WebArena, while approaching last-generation frontier models on WorkArena L1. Success rate (%). † = 147/185 task overlap due to WorkArena version mismatch from the prior work.

Model	Params	WA	VWA	L1	L2	WoB
BrowserGym ecosystem (same harness)						
Claude 3.5 Sonnet	–	36.0	22.0	56.4	38.8†	69.8
o1-mini	–	29.9	–	56.7	14.3†	67.8
GPT-4o	–	31.5	26.3	45.5	7.5†	63.8
Llama 3.1 405B	405B	22.6	–	43.3	8.9†	64.6
Llama 3.1 70B	70B	17.1	–	27.9	3.4†	57.6
GPT-4o-mini	–	13.6	18.0	27.0	2.0†	56.6
This work (same harness)						
Gemini 3.1 Pro	–	53.8	47.9	79.4	40.0	77.1
Qwen3.5-27B	27B	41.5	37.4	57.0	18.9	70.9
A3-Qwen3.5-9B	**9B**	**41.5**	**33.9**	**51.5**	**9.7**	**69.0**
Qwen3.5-9B (base)	9B	31.0	28.5	33.3	2.2	63.2

A.6 Comparison with Published Leaderboard Results

Table 12 compares A3 with proprietary, open-weight, and sub-10B models from the WebArena/VisualWebArena[2] and BrowserGym[3] leaderboards.

Uncertainty and paired significance. In table 13, we report the 95% confidence intervals computed using the Wilson method alongside the two-sided McNemar tests by pairing outcomes based on task and random seed; all results achieve $p < 10^{-4}$, which provides the statistical evidence that our method shows significant improvement for VisualWebArena and MiniWoB.

For MiniWoB, we use all 625 episodes to compute SRs and intervals. However, we collapsed repeated task-and-seed pairs to 590 pairs for testing. For other benchmarks, we use the displayed n for both.

Comparison against open models on WebArena leaderboard Moreover, A3 reaches 42.1% on the full WebArena split (of 812) whereas Go-Browse reaches 21.7% (Gandhi & Neubig, 2025) on the official WebArena leaderboard. WebRL (Qi et al., 2024) and WebAgent-R1 (Wei et al., 2025) are excluded because they use WebArena-Lite, a human-verified subset of 165 tasks rather than the full 812-task benchmark.

[1] https://huggingface.co/datasets/agentlabtraces/agentlabtraces

[2] https://docs.google.com/spreadsheets/d/1M801lEpBbKSNwP-vDBkC_pF7LdyGU1f_ufZb_NWNBZQ

[3] https://huggingface.co/spaces/ServiceNow/browsergym-leaderboard

Table 12: A3-Qwen3.5-9B compared against official leaderboard results across all five BrowserGym benchmarks. For each benchmark, we show the top proprietary, open-weight, and sub-10B models. Our 9B SFT model is the best open-weight SFT result on full WebArena (nearly 2× Go-Browse) and exceeds GPT-4o on 4/5 benchmarks under the same GenericAgent protocol. Success rate (%); ‡ = custom agent pipeline. A3-Qwen3.5-9B and Qwen3.5-9B numbers are on the full benchmark (812 WA, 910 VWA, 330 L1, 341 L2, 625 WoB).

Harness	Model	Params	WA	VWA	L1	L2	WoB
Proprietary SOTA (from official leaderboards)							
Custom‡	OpAgent (Guo et al., 2026)	–	71.6	–	–	–	–
Custom‡	SGV (Andrade et al., 2025)	–	–	54.0	–	–	–
GenericAgent	GPT-5 (de Chezelles et al., 2024)	–	–	–	79.1	69.4	71.5
GenericAgent	Claude 4 Sonnet (de Chezelles et al., 2024)	–	–	–	63.3	40.4	70.7
Open-weight (≥10B, from leaderboards)							
Custom	TTI (Shen et al., 2025)	12B	26.1	–	–	–	–
Custom	Llama-3-70B + Search (Koh et al., 2024b)	70B	–	16.7	–	–	–
GenericAgent	GPT-oss-120B (de Chezelles et al., 2024)	120B	–	–	50.9	11.5	66.4
Open-weight (<10B)							
Custom	Go-Browse (Gandhi & Neubig, 2025)	7B	21.7	–	–	–	–
Custom	NNetNav (Murty et al., 2024)	8B	16.3	–	–	–	–
Custom	ViGoRL (Sarch et al., 2025)	7B	–	11.2	–	–	–
This work (GenericAgent)							
GenericAgent	**A3-Qwen3.5-9B**	**9B**	**42.1**	**33.7**	**51.5**	**10.6**	**69.0**
GenericAgent	Qwen3.5-9B (base)	9B	30.2	26.2	33.3	2.9	63.2

Table 13: Full-benchmark uncertainty and paired significance for Qwen3.5-9B before and after fine-tuning. Unlike Table 1, this table combines the official train and test task partitions for WebArena, VisualWebArena, and WorkArena++ L2. Success rates, confidence interval endpoints, and absolute changes are in %.

Benchmark	n	Base (95% CI)	A3 (95% CI)	Δ (% absolute)	Exact p
WebArena	812	30.2 [27.1, 33.4]	42.1 [38.8, 45.5]	+11.9	4.14×10^{-14}
VisualWebArena	910	26.2 [23.4, 29.1]	33.7 [30.7, 36.9]	+7.6	2.62×10^{-6}
WorkArena L1	330	33.3 [28.5, 38.6]	51.5 [46.1, 56.9]	+18.2	2.72×10^{-11}
WorkArena++ L2	341	2.9 [1.6, 5.3]	10.6 [7.7, 14.3]	+7.6	1.29×10^{-5}
MiniWoB	625	63.2 [59.3, 66.9]	69.0 [65.2, 72.5]	+5.8	7.07×10^{-6}

B Ablation Studies

Judge Filtering Ablation We train Qwen3.5-9B on all valid trajectories from our Gemini 3 Pro (reduced thinking) teacher, allowing us to obtain 37.0% on 381 WebArena tasks. When using all 2,322 trajectories classified as successful by our judge (Table 14), we obtain 41.5%, showing that filtering improves finetuning performance despite using less data to train the models.

Table 14: Judge filtering ablation on WebArena (381 tasks). Removing the Judge and training on all trajectories (including failed ones) reduces overall success rate from 41.5% to 37.0% despite 40% more training data. Per-site breakdowns show the largest drops on Map (26.4% to 18.9%), Shopping (34.1% to 27.3%), and GitLab (53.3% to 46.7%).

Training Data	Trajs	All	Rdt	Git	Shop	ShAdm	Map
Judge-filtered (A3-Qwen3.5-9B)	2,322	**41.5**	60.0	**53.3**	**34.1**	**44.9**	**26.4**
Unfiltered (no Judge)	2,999	37.0	**62.2**	46.7	27.3	42.3	18.9
Change (% absolute; no Judge)		−4.5	+2.2	(−6.6)	(−6.8)	(−2.6)	(−7.5)

Table 15: Data scaling on WebArena (381 tasks). Performance increases with training data, from 32.0% at 285 trajectories to 41.5% at 2,322, with diminishing returns at larger scales.

Variant	Trajs	Steps	All	Rdt	Git	Shop	ShAdm	Map
285-traj	285	2,036	32.0	53.3	39.1	28.4	33.3	15.1
715-traj	715	5,060	37.0	57.8	46.7	27.3	42.3	22.6
1430-traj	1,430	9,999	40.2	**62.2**	46.7	35.2	46.2	20.8
Full (A3-Qwen3.5-9B)	2,322	16,353	**41.5**	60.0	**53.3**	35.2	46.2	**26.4**

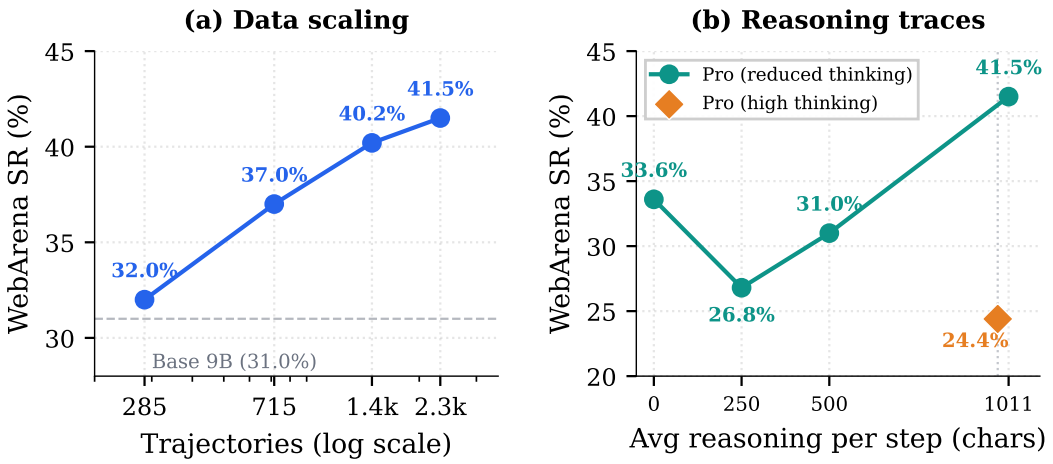

Figure 8: (a) WebArena SR increases with training data from 32.0% (285 trajectories) to 41.5% (2,322), with diminishing returns. (b) Reasoning trace ablation: removing traces lowers success from 41.5% to 33.6%, while truncating them lowers success to 31.0% at 500 chars and 26.8% at 250, suggesting incoherent mid-thought cutoffs degrade training signal more than the absence of reasoning.

Reasoning Trace Ablation We find that removing reasoning from `<thought>` and `<think>` blocks of tokens, or, alternatively, truncating each block to 500 or 250 characters, results in the success rate to fall from 41.5% to 33.6% (no reasoning) and to 31.0% or 26.8% with 500- or 250-character truncation (Figure 8); the lower performance can be attributed to the partial reasoning adding more noise than removing reasoning completely.

Teacher thinking budget. However, training Qwen3.5-9B on Gemini 3 Pro in high thinking mode results in a success rate of 24.4% on the WebArena test split, which is substantially lower than the 41.5% with reduced thinking; which highlights that a larger thinking budget does not improve smaller student.

Persona Ablation To understand the effect of adding persona, we train Qwen3.5-9B on two 368-trajectory datasets, one with, and another without persona, from Gemini 3 Pro (reduced thinking), while keeping other settings the same. We find that a persona-based approach reaches 34.1% (130/381), versus 25.2% (96/381) with generic prompts (we note the success rates differ since this is a controlled ablation on a smaller subset). Since the trajectories differ in length when removing personas, our datasets respectively contain 2,639 and 3,004 trajectory steps used for training.

Hints Ablation To understand the effect of removing hints from the judge's decision, we train Qwen3.5-9B on 600 trajectories of both variants: 4,261 training examples (trajectory steps) with hints results in a SR of 37.8%, whereas without hints, we obtain 4,195 examples that can be used to finetune Qwen3.5-9B to achieve 35.4% (Table 16). We further found that removing hints results in 632 of 2,968 judge success decisions to change (corresponding to 21.3% of the total.

Table 16: Hints ablation on WebArena (381 tasks). Both variants use 600 trajectories; removing hints from the Judge reduces downstream student performance from 37.8% to 35.4%, with the largest per-site drops on Shopping Admin (41.0% to 34.6%) and GitLab (44.6% to 39.1%).

Training Data	Trajs	All	Rdt	Git	Shop	ShAdm	Map
Full pipeline (A3-Qwen3.5-9B)	2,322	**41.5**	**60.0**	**53.3**	34.1	**44.9**	**26.4**
With hints (subsampled)	600	37.8	60.0	44.6	30.7	41.0	24.5
No hints	600	35.4	57.8	39.1	**35.2**	34.6	22.6
Change (% absolute; no hints)		−2.4	(−2.2)	(−5.5)	+4.5	(−6.4)	(−1.9)
Change (% absolute; vs. full)		−6.1	(−2.2)	(−14.2)	+1.1	(−10.3)	(−3.8)

Retroactive Labeling Ablation We further run a retroactive labeling-based approach to understand if prior works, such as NnetNav, might provide a strong alternative to our approach. We design a NNetNav-style ablation that processes 1,200 Gemini 3 Pro exploration episodes using the publicly available NNetNav pipeline[4]. We use Gemini 3.1 Pro as a drop-in replacement to produces state summaries, reward scores, task labels, regenerated actions, and terminal responses, following Gemini 3 Pro's discontinuation. As a result, 2,133 explorative trajectories were processed to obtain 14,857 trajectory steps, which we used to train Qwen3.5-9B. The resulting web agent reaches 20.2% SR on WebArena test (77/381). This result shows that despite using the same exploration teacher as A3, exploration-based training with retroactively labeling does not improve the baseline results.

Pipeline Comparisons In Table 17, we show Qwen3.5-9B trained on 5 different settings, using the full pipeline and with several targeted ablations, which we evaluate on the 381 tasks from WebArena's test split. Our result shows that each component in the pipeline is needed to produce a high quality dataset that we can use to train local web agents. Since rerunning the full pipeline would require a substantially higher budget than the targeted ablations that we have shown in the other scenarios, our Go-Browse results instead leverages the full 37,435 released training examples published from the original work [5], as their pipeline differ in significant manner that are harder to isolate without changing the nature of their pipelines, which would make it a different task.

Table 17: Experiments grouped by training data construction. Only the two 368-trajectory rows isolate a pipeline component; their training-example counts differ because their trajectories have different lengths.

Training data	Construction	Generation model(s)	Training examples	SR (%)
A3 full	Full pipeline	Gemini 3 Pro	16,353	41.5
A3 subset	Full pipeline, 368 trajectories	Gemini 3 Pro	2,639	34.1
No personas (InSTA-style)	Generic task prompts, 368 trajectories; hints retained	Gemini 3 Pro	3,004	25.2
NNetNav-style	Released NNetNav processing of 1,200 exploration episodes	Gemini 3 Pro exploration; Gemini 3.1 Pro processing	14,857	20.2
Go-Browse data	Released successful examples in native format	Original Go-Browse pipeline	37,435	23.6

C Framework and Implementation Details

C.1 Vocabulary Mapping Across Pipelines

To establish a common vocabulary, in Table 18, we show how each prior work's components map to each of our Agent-as-Annotators modules, and further identify components that are not present (we note that task generation may precede execution, or it may relabel explorations afterward).

[4] https://github.com/MurtyShikhar/NNetnav

[5] https://huggingface.co/datasets/apurvaga/go-browse-wa

Table 18: Vocabulary mapping across trajectory synthesis pipelines. "–" indicates the module is absent.

A3 Module	InSTA	NNetNav	Explorer	Go-Browse
Persona Gen.	–	Persona DB	–	–
Exploration	Agent	LLM Explorer	Task Proposer	NavExplorer
Task Gen.	Task Proposer	Task Labeler	Refiner + Summ.	PageExplorer
Hints	–	–	–	–
Agent	Agent	–	–	Feas.Check + Solvers
Judge	LLM Judge	ORM	Task Verifier	VLM-as-judge

C.2 Persona Generation

During the persona generation stage, we use the teacher model to generate 250 personas, each with a name, professional skills, interests; in a second pass, we produce a freeform description of their background and activity pattern on the web, allowing a more natural and diverse description of the personas. We show examples below:

1. **Alice Chen** (Data Scientist): Skills in Python Programming, Data Analysis, Machine Learning. Interests: Robotics, Hiking, Sci-Fi Novels. Alice specializes in transforming raw data into actionable intelligence; she spends weekends building custom robotics projects.
2. **Liam O'Connor** (Senior Graphic Designer): Skills in Graphic Design, Adobe Creative Suite, Typography. Interests: Analog Photography, Indie Music, Street Art. Liam merges digital precision with physical-world textures, shooting exclusively on film and collecting vinyl records.
3. **Dr. Fatima Al-Rashidi** (Biomedical Researcher): Skills in Bioinformatics, Statistical Analysis, Laboratory Techniques. Interests: Scientific Illustration, Mountaineering, Calligraphy. Fatima bridges computational biology with hands-on lab work, frequently presenting at international conferences.

Persona-conditioned Exploration Given a persona, we can use the teacher inside the AgentLab and BrowserGym framework to explore the websites, each producing 6 explorations (one for each website), for a total of 1500 persona-conditioned explorations, enabling a high level of diversity (see Table 19).

C.3 Task Generation and Annotator Instructions

Our Task Generator module is designed to receive a persona, an exploration trajectory, and instructions adapted from WebArena (Zhou et al., 2023b). The instructions provided for the annotators calls for creative, multi-step task intents, which can expressed as templates with replaceable variables, allowing multiple concrete tasks per template. Since our method is fully automatic, we do not replace with any variable, but leave this option open for future works that may be interested in further increasing the size of the dataset without generating new intents.

Our *annotator instructions* are provided to the Task Generator model in its system prompt alongside the assigned persona and an exploration trajectory:

1. **Abstract and high-level.** The intent should require multiple actions to complete, not merely one or two steps. For example, instead of "click the science subreddit," annotators are encouraged to produce intents like "post a greeting message on science subreddit," which requires navigation, form-filling, and submission.
2. **Creative.** Common tasks such as account creation are discouraged. Instead, annotators should add constraints (e.g., "create a Reddit account identical to my GitLab one") to produce unique intents.
3. **Template-based with variables.** Intents should be formulated as templates with replaceable elements marked as variables (e.g., {{section_name}}, {{topic}}). Each template is instantiated with multiple variable assignments, producing diverse concrete tasks from a single template. For example, "Browse the {{section_name}} section to find a post containing {{topic}}" can be instantiated as "Browse the General section to find a post containing urban exploration" or "Browse the Off-Topic section to find a post containing analog photography."

Example Synthesized Tasks We present below 5 sample tasks from A3-SYNTH (one for each site), which illustrates the diversity of our synthesized intents, evaluation hints, and personas used to condition the task generation.

Reddit (persona: Liam O'Connor).

- **Intent:** "Browse the General section to find a post containing urban exploration and reply with a comment emphasizing the value of texture and grit."
- **Hint:** "The agent must navigate to the specified section of the forum, identify a post that contains content related to urban exploration (e.g., by checking titles or images), and successfully submit a reply that explicitly mentions the design principle. Success is measured by the presence of the user's comment on the post page."

GitLab (persona: Alice Chen).

- **Intent:** "Create a new public repository named 'ml-data-pipeline' with a README file, then add an issue titled 'Set up CI/CD pipeline' with the label 'enhancement'."
- **Hint:** "The agent should create the repository with the specified name, ensure it is public, and contains a README. Then navigate to the Issues section, create a new issue with the exact title, and apply the 'enhancement' label. Success requires both the repository and issue to be visible."

Shopping (persona: Dr. Fatima Al-Rashidi).

- **Intent:** "Find a laboratory notebook suitable for research documentation, add it to the shopping cart, and proceed to the checkout page."
- **Hint:** "The agent should search for laboratory or research notebooks, select an appropriate product, add it to the cart, and navigate to the checkout page. The cart should contain at least one item and the agent should be on the checkout page at the end."

Shopping Admin (persona: Alice Chen).

- **Intent:** "Navigate to the product catalog and change the price of the most expensive product in the 'Electronics' category to $99.99."
- **Hint:** "The agent must access the product catalog, filter or navigate to the Electronics category, identify the most expensive product, edit its price to 99.99, and save the changes. The product's price should be updated to $99.99 upon completion."

Map (persona: Liam O'Connor).

- **Intent:** "Search for street art galleries near the city center and get directions from the nearest parking lot to the top-rated gallery."
- **Hint:** "The agent should search for street art or art galleries, identify one near the center with good ratings, then search for a nearby parking lot and request directions between the two locations. Success requires the directions to be displayed on the map."

C.4 JUDGE EVALUATION PROTOCOL

Our judge module is used to evaluate each trajectory in order to determine whether the web agent successfully completed the requested task. Our Judge module is designed based on the judge from AgentRewardBench (Lù et al., 2025). We implement it as a LLM judge that receives the full interaction and is requested to produce parseable outputs (for evaluation).

Judge input. For each trajectory, our Judge module receives: (1) a system prompt defining its role as judge for web tasks, (2) the task intent, (3) the list of actions with URLs and reasoning chains for each step, (4) the last accessibility tree recorded, and (5) the last screenshot from the trajectory.

Success evaluation. The Judge answers whether the sequence of actions successfully achieved the goal, choosing from Successful, Unsuccessful, or In Progress. It returns its reasoning in `<reasoning>` and its decision in `<answer>`.

Hint integration. We further append the hints to the task goal in order to provide the Judge with a structured criteria for determining success.

Parsing. We extract the decision from `<answer>` and retain only trajectories labeled Successful.

C.5 Training Data Statistics

Trajectories that are considered successful by the judge are converted into observation-action examples used for training, which we report in Table 4 with counts.

Coverage and instruction diversity. To understand how much our dataset covers, we analyzed 16,353 steps from all 2,322 successful trajectories (Table 19); we compute the Distinct-n metric, which is the ratio of unique to total instruction n-grams (Li et al., 2016), allowing us to estimate the diversity of the data. Reading from the results, we find that there are over 37,467 unique 4-grams, indicating a high level of diversity in the tasks.

Table 19: Website-state coverage, instruction diversity, and the most frequently accessed functional pages in the A3-Qwen3.5-9B training data. Home pages and landing pages are omitted from panel (c).

(a) Website-state coverage

Site	URLs	Paths
Shopping	711	231
Map	572	145
GitLab	541	369
Wikipedia	494	465
Reddit	387	255
Shopping Admin	171	171
All sites	**2,876**	**1,636**

(b) Instruction diversity

n-gram	Unique	Total	Distinct-n
Unigrams	6,257	57,975	0.11
Bigrams	22,496	55,653	0.40
Trigrams	33,206	53,331	0.62
4-grams	37,467	51,009	0.73

(c) Most frequently accessed functional pages

#	Page	Site	Steps	Share (%)
1	`/directions`	Map	928	5.7
2	`/catalogsearch/result`	Shopping	535	3.3
3	`/search`	Map	469	2.9
4	New-product form (`/admin/catalog/product/new`)	Shopping Admin	345	2.1
5	New-customer form (`/admin/customer/index/new`)	Shopping Admin	271	1.7
6	Wishlist (`/wishlist/index/index`)	Shopping	263	1.6
7	`/search`	Reddit	240	1.5
8	Customer grid (`/admin/customer/index`)	Shopping Admin	227	1.4
9	`/submit` (new post)	Reddit	213	1.3
10	`/catalogsearch/result/index`	Shopping	186	1.1

Data format. Each conversation contains a system prompt, task intent, observations from the browser (URL, accessibility tree, screenshot), the reasoning chain from the model, and the action predicted by the model. We use XML-style blocks like `<thought>`, `<think>`, and `<action>` to enable easier parsing for recording the thoughts and execute the final action.

Action distribution. Analyzing the training data (16,353 steps) generated from Gemini 3.0 Pro (reduced thinking), we find the following distribution: `click` (65.3%), `fill` (23.4%), `send_msg_to_user` (4.1%), `select_option` (2.5%), `scroll` (1.5%), `keyboard_press` (1.5%), `hover` (1.0%), `goto` (0.4%), `go_back` (0.3%), and other actions (0.2%). This highlights a high dependency on several tasks, with other actions like press and visiting websites to be very rare.

Response statistics. Each model response averages 1,920 characters. We found that <thought> blocks achieve a median 996 characters whereas<think> has a median of 459 characters when present. The presence of two types of tokens is due to Gemini using thoughts instead of thinking, but at the same time, can also produce think tokens, resulting in two segments of thinking traces (thought being the summary, whereas think is part of the completion).

C.6 HYPERPARAMETERS

Table 20 lists the training hyperparameters for A3-Qwen3.5-9B.

Table 20: Training hyperparameters for A3-Qwen3.5-9B.

Hyperparameter	**Value**
Base model	Qwen/Qwen3.5-9B
Training data	Pro (reduced thinking) (16,353 examples)
Learning rate	1×10^{-5}
LR schedule	Cosine with warmup
Warmup ratio	0.03
Weight decay	0.0
Max gradient norm	0.3
Epochs	2
Batch size per GPU	1
Gradient accumulation steps	4
Effective batch size	$8 \times 4 \times 1 = 32$
Max sequence length	8,192 tokens
Precision	bfloat16
Attention	Flash Attention 2
Parallelism	FSDP (4–8 GPUs)
Optimizer	AdamW
Checkpoint saving	Every 200 steps
Random seed	0

Infrastructure. We use HuggingFace Transformers and TRL with FSDP (Zhao et al., 2023). Eight H100 offers an optimal configuration to train the 9B-sized models, whereas a single A6000 can be used for inference, albeit at a slower rate than commercial APIs.

Loss function. We apply the causal cross-entropy loss only to assistant tokens, leaving system and user tokens masked.

D BENCHMARK DETAILS

D.1 A3-SYNTH DETAILS

Task categories. Our tasks are designed to span a wide range of web interaction patterns: content creation (posting, commenting, editing), information retrieval, filtering or comparing items, navigation (finding specific pages, following links), data management (creating records, updating fields), and transaction-based workflows (adding to cart, submitting forms). We believe that a persona-based approach enables the generation to cover diverse use cases within each website.

Relationship to WebArena Test Tasks. Although A3-SYNTH tasks are produced for the same six web environments used in the WebArena test set, we note that the synthesized task intents are fully distinct from the 381 test tasks authored by humans. We specifically did not seed the task generator with tasks from WebArena to avoid bias. Instead, we rely on the general annotation guidelines that do not mention specific solution patterns. Due to the environment overall, we consider WebArena as an "in-domain" benchmark, but "out-of-domain" with respect to task instructions (thus it is not IID with respect to the tasks). The consistent improvements on fully out-of-distribution benchmarks

like WorkArena, WorkArena++, MiniWoB and Online-Mind2Web confirm that the learned skills generalize beyond environment-specific knowledge, and the improvements on VisualWebArena highlight the model can also generalize to more vision-heavy tasks.

D.2 Evaluation Benchmark Summary

In table 21, we summarize the key characteristics of all benchmarks used in our evaluation.

Table 21: Summary of evaluation benchmarks.

Benchmark	Tasks	Steps	Domain	Relation to Training
WebArena	812	30	Self-hosted web apps	In-domain (same env)
VisualWebArena	910†	30	Visual web tasks	OOD (different tasks)
WorkArena L1	330	15	ServiceNow (basic)	OOD (different env)
WorkArena++ L2	185	50	ServiceNow (complex)	OOD (different env)
MiniWoB	625	15	Synthetic micro-tasks	OOD (different env)
Online-Mind2Web	300	30	Live websites	OOD (different env)