WebArbiter: A Principle-Guided Reasoning Process Reward Model for Web Agents

Anonymous authors
Paper under double-blind review

Abstract

Web agents hold great potential for automating complex computer tasks, yet their interactions involve long horizons, multi-step decisions, and actions that can be irreversible. In such settings, outcome-based supervision is sparse and delayed, often rewarding incorrect trajectories and failing to support inference-time scaling. This motivates the use of Process Reward Models (WebPRMs) for web navigation, but existing approaches remain limited: scalar WebPRMs collapse progress into coarse, weakly grounded signals, while checklist-based WebPRMs rely on brittle template matching that fails under layout or semantic changes and often mislabels superficially correct actions as successful, providing little insight or interpretability. To address these challenges, we introduce **WebArbiter**, a reasoning-first, principle-inducing WebPRM that formulates reward modeling as text generation, producing structured justifications that conclude with a preference verdict and identify the action most conducive to task completion under the current context. Training follows a two-stage pipeline: reasoning distillation equips the model with coherent principle-guided reasoning, and reinforcement learning corrects teacher biases by directly aligning verdicts with correctness, enabling stronger generalization. To support systematic evaluation, we release WebPRMBench, a comprehensive benchmark spanning four diverse web environments with rich tasks and high-quality preference annotations. On WebPRMBench, WebArbiter-7B outperforms the strongest baseline, Gemini Flash, by 10.9%. In reward-guided trajectory search on WebArena-Lite, it surpasses the best prior WebPRM by up to 7.2%, underscoring its robustness and practical value in real-world complex web tasks.

1 Introduction

Large Language Models (LLMs) (Achiam et al., 2023; Guo et al., 2025a) have demonstrated impressive capabilities in planning (Huang et al., 2024; Zhang et al., 2025a), decision-making (Li et al.,

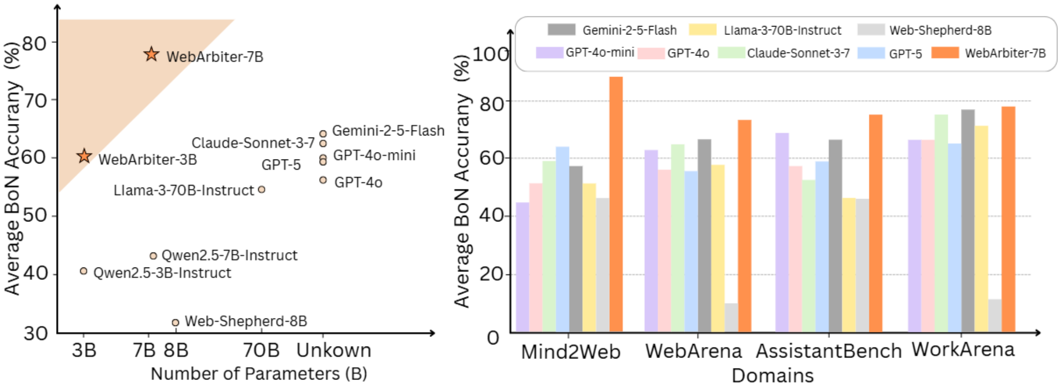


Figure 1: Performance comparison on WebPRMBench. **Left:** *Average Best-of-N Acc* vs. model size, showing superior efficiency despite smaller scale. **Right:** Domain-wise *Avg BoN Acc*, where WebArbiter achieves the best results across all environments, confirming robustness and scalability.

2024), and complex task execution (Xi et al., 2024; Zhang et al., 2025b). Extending these abilities with browser access enables LLM agents to perform complex web tasks similar to humans (OpenAI, 2025a; Anthropic, 2024a; Adept, 2022). However, web interactions involve long horizons, multi-step decisions, and actions that can be irreversible. For example, submitting an incorrect form may not be recoverable. This requires agents to make reliable decisions throughout the interaction process, rather than relying solely on final outcomes. Traditional Outcome Reward Models (ORMs) are ill-suited: they provide only sparse and delayed feedback, may misclassify incorrect trajectories as successes, and cannot guide inference-time strategies, such as reward-guided search.

Recent studies on web agents (Zhang et al., 2025b; Koh et al., 2025) have introduced step-level rewards using LLM-as-judge. While such supervision can be useful, LLM-as-judge suffers from high cost, limited scalability, and susceptibility to hallucination, often rewarding fluent but incorrect actions. This motivates the development of dedicated Process Reward Models (WebPRMs) for web tasks. Existing WebPRMs largely fall into two categories: scalar WebPRM (Miao et al., 2025), which collapse progress into coarse scores with little interpretability or weak grounding; and generative WebPRM (Chae et al., 2025), which rely on checklists that are brittle under dynamic layouts and shifting semantics. Moreover, lacking explicit reasoning, generative WebPRMs remain vulnerable to surface correlations and sensitive to page changes. These limitations highlight the need for a reasoning-first WebPRM that can verify progress, resist superficial biases, and provide interpretable chains for diagnosing errors.

To this end, we propose **WebArbiter**, a reasoning-first, principle-inducing WebPRM. It formulates process reward modeling as text generation: given task context and candidate actions with their reasoning traces, the model produces a structured justification that concludes with a preference verdict, identifying the action most conducive to task completion. Unlike scalar scores or checklist-based methods tied to fixed templates, WebArbiter dynamically derives principles from user intent and the current state, incorporates them into reasoning chains that verify whether an action advances task completion. Training follows a two-stage pipeline: reasoning distillation equips the model with coherent principle-guided reasoning, and reinforcement learning corrects teacher biases and aligns verdicts with correctness. This design transforms reward signals from shallow correlations into auditable analyses, making judgments robust to environment and page variations, resistant to spurious cues, and accurate in credit assignment.

To advance the evaluation of WebPRMs, we introduce WEBPRMBENCH, the first comprehensive evaluation benchmark spanning diverse environments dedicated to WebPRMs. It provides 1,287 step-level preference instances, each consisting of one correct action and four rejected alternatives, collected across 4 web environments: AssistantBench (Yoran et al., 2024), Mind2Web (Deng et al., 2023), WorkArena (Drouin et al., 2024; Boisvert et al., 2025), and WebArena (Zhou et al., 2023). The tasks span everyday activities such as online shopping and forum posting, as well as enterprise scenarios like updating schedules in IT management platforms. By combining scale, diversity, and fine-grained supervision, WEBPRMBENCH establishes a unified standard for systematic evaluation of WebPRMs, with *Pairwise* and *Best-of-N (BoN) Accuracy* as the primary metrics.

Extensive experiments on WEBPRMBENCH show that WebArbiter achieves SOTA *Avg. BoN Acc*, consistently surpassing both proprietary LLMs and previous SOTA WebPRM, WebShepherd, across all environments, and outperforming the strongest LLM baseline, Gemini Flash, by +10.9%. Beyond static evaluation, WebArbiter also proves effective in practice: in reward-guided trajectory search on WebArena-Lite (Liu et al., 2024b), it delivers substantial gains, surpassing WebShepherd by up to 7.2%, further demonstrating robustness in realistic interaction settings.

The key contributions of this work are:

1. We propose WebArbiter, a reasoning-first, principle-inducing PRM trained with reasoning distillation and reinforcement learning, providing auditable reasoning chains and correctness-aligned signals.
2. We release WEBPRMBENCH, the first comprehensive evaluation benchmark to provide systematic WebPRM evaluation across 4 web environments, using *Pairwise* and *Best-of-N (BoN) Accuracy* as standard metrics.
3. We show that WebArbiter achieves SOTA performance on WEBPRMBENCH, surpassing both proprietary LLMs and the previous SOTA WebPRM. WebArbiter delivers up to +7.2% gains in reward-guided trajectory search on WebArena-Lite.

4. We analyze the training dynamics of WebArbiter, revealing how different strategies influence performance.

2 Related Work

2.1 LLM-Based Autonomous Web Agents

LLM advances have enabled browser-operating agents (Kim et al., 2024; Sun et al., 2024; Prasad et al., 2023; Fu et al., 2024; Ma et al., 2023; Zheng et al., 2023b; Tao et al., 2023). One line distills environment-specific state–action pairs from demonstrations, strong on seen states yet brittle on novel ones, with SteP as a leading example on WebArena (Sodhi et al., 2024; Zhou et al., 2023). A second line pursues open-ended exploration via reflexive evaluation and search (Pan et al., 2024; Shinn et al., 2024; Koh et al., 2024; Zhang et al., 2025b). A third direction applies reinforcement learning (Qi et al., 2025; Wei et al., 2025), yet real sites provide sparse and delayed signals, which makes value learning unstable without dense step feedback. Therefore, WebAgents require a process-level judge that assesses progress step by step and supplies auditable signals for search and planning.

2.2 Reward Models in Reasoning and Web Tasks

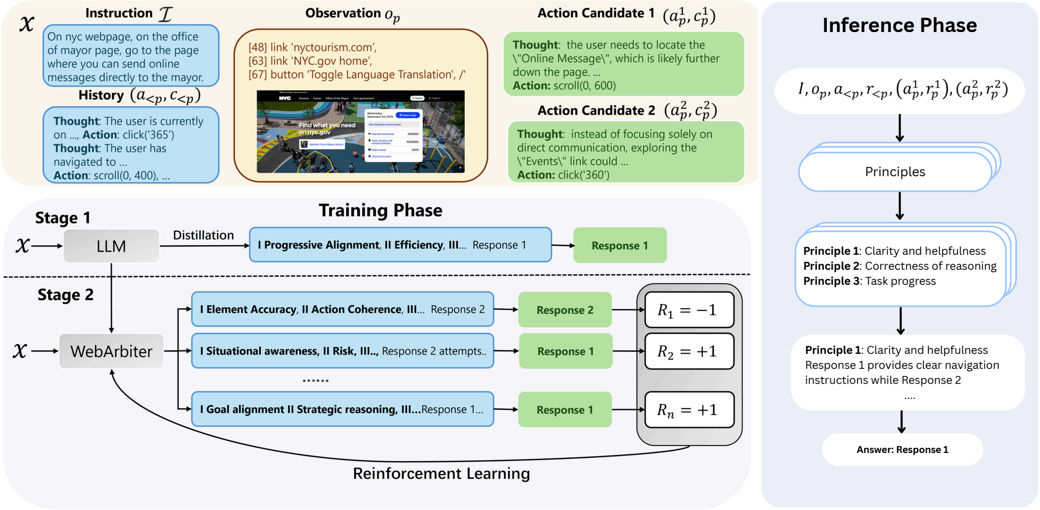
RMs fall into two families. Scalar RMs attach a single numeric score to a response with a linear scorer and use either absolute or discriminative schemes for evaluation (Uesato et al., 2022; Ouyang et al., 2022; Liu et al., 2024a; 2025; Park et al., 2024; Wang et al., 2024a; 2023b; 2024b). Generative RMs instead produce natural–language feedback from which rewards are extracted, aligning with LLM-as-Judge and supporting both single-instance evaluation and multi-response comparison; they show promising scalability but raise reliability concerns due to bias and hallucination (Lightman et al., 2023; Wang et al., 2023a; Zhang et al., 2025c; Wu et al., 2024; Ye et al., 2025; Zhang et al., 2024; Zheng et al., 2023a). Building on these, Reasoning RMs cast judging as a deliberate process: they first generate an explicit, context-grounded chain of principle and analysis, then issue a single preference verdict, yielding adaptive test-time compute, stronger grounding, and interpretable feedback (Chen et al., 2025a; Guo et al., 2025b; Mahan et al., 2024). In web agents, action rewards have been derived by the following methods: LLM-as-Judge (Zhang et al., 2025b; Koh et al., 2025), slow and unstable during search; scalar scoring (Miao et al., 2025), which collapses progress into coarse values with little interpretability and weak grounding; and checklist-driven generative feedback (Chae et al., 2025), whose external templates are brittle under layout and semantic drift and prone to surface correlations. These limitations motivate a reasoning-first approach that turns rewards from shallow correlations into auditable analyses. WebArbiter produces structured justifications with a single preference verdict, induces principles from the current instruction and state, and is trained by reasoning distillation followed by reinforcement learning, so that judgments remain robust to environment variations, resist spurious cues, and provide accurate credit assignment while supporting inference-time scaling.

3 Methodology

In this section, we present the design of WebArbiter. We begin by framing web navigation as a Partially Observable Markov Decision Process (POMDP) in §3.1, then describe how we construct a pairwise-preference dataset for training in §3.2. We introduce the training pipeline of WebArbiter model in §3.3. For clarity, we summarize all notations in Appendix A.

3.1 Background

We formalize web navigation as a POMDP. The environment $\mathcal{E}$ is defined by a state space $\mathcal{S}$, an action space $\mathcal{A}$, and an observation space $\mathcal{O}$. $T : \mathcal{S}\times\mathcal{A}\rightarrow\mathcal{S}$ denotes the state transition function. At step p, the agent receives a partial observation $o_p\in\mathcal{O}$, executes $a_p\in\mathcal{A}$, and transitions to $s_{p+1} = T(s_p, a_p)$ with a new observation o_{p+1}. Following WebArena (Zhou et al., 2024), we represent observations using accessibility trees, i.e., text-only encodings of visible interactive elements and their attributes. Given a task instruction $\mathcal{I}$ and the initial state $s_0 \in \mathcal{S}$, the agent aims to generate a trajectory



On nyc webpage, on the office of mayor page, go to the page where you can send online messages directly to the mayor.

Thought: The user is currently on ..., **Action**: click('365')
Thought: The user has navigated to ...
Action: scroll(0, 400), ...

[48] link 'nyctourism.com',
[63] link 'NYC.gov home',
[67] button 'Toggle Language Translation', /'

Thought: the user needs to locate the \"Online Message\", which is likely further down the page. ...
Action: scroll(0, 600)

Thought: instead of focusing solely on direct communication, exploring the \"Events\" link could ...
Action: click('360')

I Progressive Alignment, **II Efficiency**, **III...** Response 1

I Element Accuracy, **II Action Coherence**, **III...** Response 2

I Situational awareness, II Risk, III.., Response 2 attempts..

......

I Goal alignment II Strategic reasoning, III... Response 1...

Principle 1: Clarity and helpfulness
Principle 2: Correctness of reasoning
Principle 3: Task progress

Principle 1: Clarity and helpfulness
Response 1 provides clear navigation instructions while Response 2
....

Figure 2: Overview of WebArbiter. Given an instruction $\mathcal{I}$, current observation o_p, and history $(a_{<p}, c_{<p})$, the model compares candidate actions (a_p^1, c_p^1) and (a_p^2, c_p^2). In **Stage 1**, principle-guided reasoning traces are distilled from a stronger teacher LLM. In **Stage 2**, WEBARBITER is trained with reinforcement learning using verifiable rewards $R \in \{-1, +1\}$, producing structured justifications and a final verdict. During inference, the model induces principles (e.g., clarity, correctness, progress) from $(\mathcal{I}, o_p, a_{<p}, c_{<p}, (a_p^1, r_p^1), (a_p^2, r_p^2))$, applies them to candidate actions, and outputs an auditable judgment identifying the action that best advances task completion.

$\tau = (a_1, \ldots, a_P)$ that completes the task. Here P is the trajectory length and $a_p \in \mathcal{A}$ denotes the action at step p. The task evaluator determines whether the task is completed based on the final state.

3.2 TRAINING DATASET CONSTRUCTION

We build on the WEBPRM COLLECTION (Chae et al., 2025) for training WebArbiter. Each instance consists of an instruction $\mathcal{I}$, a sequence of observations $O = (o_1, \ldots, o_P)$, and expert-annotated trajectories. Specifically, the dataset provides a set of positive actions $A^+ = (a_1^+, \ldots, a_P^+)$ taken from expert demonstrations and negative actions $A^- = (a_1^-, \ldots, a_P^-)$ obtained from rejected trajectories. We convert these into pairwise preference samples where each candidate action is paired with its reasoning trace, yielding the preference dataset $\mathcal{D}_{\text{Train}}$ used for WebArbiter training.

3.3 WEBARBITER: A PRINCIPLE-INDUCING REASONING PROCESS REWARD MODEL

WebArbiter is built on a Transformer-decoder backbone and formulates process reward modeling as a text generation task. At each state, it evaluates candidate actions $\{(a_p^q, c_p^q)\}_{q=1}^{Q}$, where each action a_p^q is paired with a reasoning trace c_p^q explaining why the agent generated this action. Given task instruction $\mathcal{I}$, observation o_p, and history $(a_{<p}, c_{<p})$, the model autoregressively generates a structured justification $j = (j_1, \ldots, j_L)$ of length L that concludes with a preference verdict $\hat{y}$ selecting the most appropriate action among the candidates. The historical traces are $c_{<p} = \{c_1, \ldots, c_{p-1}\}$, i.e., the per-action reasoning traces for previously executed actions. A concrete training example is provided in Appendix B. While our experiments instantiate this framework in the standard pairwise preference setting, the design is general and extends naturally to multi-candidate.

Unlike the scalar WebPRM (Miao et al., 2025) that collapses progress into opaque scores or the checklist-based WebPRM (Chae et al., 2025), WebArbiter is a reasoning-first, principle-inducing WebPRM: it dynamically derives principles from user intent and the current state, integrates them into reasoning chains that explicitly assess whether each candidate action truly advances task completion. This design moves reward signals beyond shallow correlations toward auditable analyses, yielding judgments that are robust to environment changes, resistant to spurious cues, and precise in credit assignment.

Formally, the preference dataset is defined as

$$\mathcal{D}_{\text{Train}} = \{(\mathcal{I}^{(i)}, o_p^{(i)}, a_{<p}^{(i)}, c_{<p}^{(i)}, (a_p^{1(i)}, c_p^{1(i)}), (a_p^{2(i)}, c_p^{2(i)}), y^{(i)})\}_{i=1}^{M}, \tag{1}$$

where $y \in \{a_p^1, a_p^2\}$ denotes the preferred action. For notational simplicity, let

$$x = (\mathcal{I},\, o_p,\, a_{<p},\, c_{<p},\, (a_p^1, c_p^1),\, (a_p^2, c_p^2)). \tag{2}$$

WebArbiter π_θ is parameterized by θ and models the justification j autoregressively:

$$\pi_\theta(j \mid x) = \prod_{l=1}^{L} \pi_\theta(j_l \mid x, j_{<l}). \tag{3}$$

3.3.1 Training Overview

The overall training objective is to maximize the likelihood that the predicted preference matches the ground truth:

$$\max_{\pi_\theta} \quad \mathbb{E}_{(x,y)\sim\mathcal{D}_{\text{Train}},\ \hat{y}\sim\pi_\theta(j|x)} \left[\mathbb{1}(\hat{y} = y)\right]. \tag{4}$$

Training proceeds in two stages. The first stage, described in §3.3.2, is reasoning distillation, which equips the model with the ability to generate coherent principle-guided justifications. This stage encourages judgments to be grounded in explicit reasoning rather than surface correlations, as we later validate through ablation studies in §5.1.3.

Concretely, we sample K examples from $\mathcal{D}_{\text{Train}}$ to form $\mathcal{D}_{\text{SFT}}$ for supervised distillation, while the remaining data is used as $\mathcal{D}_{\text{RL}}$ for reinforcement learning. The second stage, detailed in §3.3.3, is reinforcement learning, which aligns the verdicts with correctness signals and produces interpretable step-level rewards for long-horizon tasks. Together, these stages enable WebArbiter to deliver robust, interpretable, and scalable supervision for web agents.

3.3.2 Stage 1: Reasoning Distillation

Directly prompting an instruction-tuned LLM as a reward model often yields superficial, inconsistent chains that do not justify why an action advances the task. We therefore distill principle-guided reasoning from a stronger teacher. Concretely, `o3` synthesizes structured justifications that first derive task-specific principles from the instruction and state, then ground these principles in the page, compare candidate actions against them, and finally output the preferred action. This equips WebArbiter with principles rather than surface heuristics. From ablations, we observe that removing explicit principles and using reasoning-only justifications markedly degrades performance, underscoring the importance of principle induction for stable step-level judgments on the web. Given $(x^{(i)}, y^{(i)}) \in \mathcal{D}_{\text{SFT}}$, the teacher generates a justification $\hat{j}^{(i)} = (\hat{j}_1^{(i)}, \ldots, \hat{j}_{L_i}^{(i)})$. The distillation dataset is then: $\mathcal{D}_{\text{SFT}} = \{x^{(i)}, \hat{j}^{(i)})\}_{i=1}^{K}$.

Objective. Reasoning distillation adjusts θ to maximize the likelihood of generating the teacher justification $\hat{j}$ that concludes with the preferred action y given x. We minimize the standard negative log-likelihood:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\frac{1}{K}\sum_{i=1}^{K}\sum_{l=1}^{L_i} \log \pi_\theta\left(\hat{j}_l^{(i)} \mid x^{(i)},\, \hat{j}_{<l}^{(i)}\right). \tag{5}$$

3.3.3 Stage 2: Reinforcement Learning

While distillation provides initial reasoning ability, it inherits teacher biases and may overfit to superficial patterns, limiting generalization to unseen environments. To further enhance judgment accuracy, stability, and generalization, we introduce a reinforcement learning stage. WebArbiter π_θ is treated as a policy that outputs a justification j that concludes with a final verdict $\hat{y}$. During rollout, π_θ generates the full justification and verdict, after which a correctness reward $R(x, \hat{y}) \in \{-1, 1\}$ is assigned solely based on whether $\hat{y}$ matches the ground-truth preference y. The distilled model from §3.3.2 serves as the reference policy π_{ref}, ensuring stable optimization.

Table 1: Data distribution of WEBPRMBENCH, the first comprehensive evaluation benchmark spanning diverse environments for WebPRMs.

Models	**Mind2Web**			**WebArena**	**AssistantBench**	**WorkArena**	**Avg.**
	Cross-Task	Cross-Website	Cross-Domain				
Count	142	148	417	371	29	180	1287

Objective. Reinforcement learning adjusts θ to maximize the expected reward while stabilizing reasoning style via KL regularization. The optimization objective is defined as:

$$\mathcal{L}_{\mathrm{RL}}(\theta) \;=\; \max_{\pi_\theta} \; \mathbb{E}_{(x,y)\sim\mathcal{D}_{\mathrm{RL}},\,\hat{y}\sim\pi_\theta(j|x)}\Big[R(x,\hat{y})\Big] - \beta\,\mathbb{D}_{\mathrm{KL}}(\pi_\theta \,\|\, \pi_{\mathrm{ref}})\,. \tag{6}$$

In practice, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024) to optimize this objective, which enables stable updates under binary verifiable rewards. Through this reinforcement learning stage, WebArbiter directly aligns its verdicts with correctness signals and converts structured justifications into reliable, interpretable step-level reward signals.

4 WEBPRMBENCH

In this section, we introduce WEBPRMBENCH, the first comprehensive evaluation benchmark spanning diverse environments for WebPRMs. Details of dataset construction and the evaluation protocol are provided below.

4.1 BENCHMARK CONSTRUCTION

WEBPRMBENCH is constructed from sucessful trajectories in AGENTREWARDBENCH (Lù et al., 2025), expanding beyond WEBREWARDBENCH (Chae et al., 2025), which only provides Mind2Web and limited WebArena data. We enrich WebArena with additional trajectories and incorporate AssistantBench and WorkArena, resulting in broader coverage of real-world tasks across four domains: Mind2Web, WebArena, AssistantBench, and WorkArena. Mind2Web emphasizes cross-task generalization across heterogeneous websites. WebArena provides controlled environments such as shopping, CMS, forums, and GitLab. AssistantBench introduces open-world tasks on real websites. WorkArena focuses on enterprise workflows, including IT and HR. This diversity enables systematic evaluation across both consumer-facing and enterprise scenarios, while covering different levels of control, openness, and task complexity.

For each state, the action from the successful trajectory is retained as the positive label, and four rejected alternatives with associated reasoning traces are synthesized to form preference pairs. To ensure data quality, we sample negatives from diverse policy models to broaden coverage, apply rule-based filters to remove invalid or mismatched actions, discard inconsistent cases, and conduct expert verification to further ensure reliability. We also conduct targeted auditing to eliminate potential false negatives. Reasoning traces are truncated to a fixed length to minimize formatting noise. The resulting benchmark spans 1,287 preference pairs across four environments, as shown in Tab. 1.

4.2 EVALUATION PROTOCOL

Evaluating WebPRMs requires metrics that capture both local preference fidelity and global decision reliability under realistic multi-candidate settings. Inspired by RMB (Zhou et al., 2025), we adopt two complementary metrics: *Pairwise Accuracy*, which measures correctness on individual preference pairs, and *Best-of-N (BoN) Accuracy*, which evaluates robustness when ranking among multiple distractors. Compared with *Pairwise Acc*, *BoN Acc* applies a stricter criterion by requiring the correct action to outrank all distractors simultaneously, providing stronger discriminative power and better alignment with downstream agent performance.

Pairwise Acc. Given a preference pair (a^+, a^-), where a^+ is the correct action and a^- a rejected one, the WebPRM is correct if it assigns higher preference to a^+. Formally:

$$\mathrm{Acc}_{\text{Pairwise}} = \frac{1}{|\mathcal{D}_{\text{Bench}}|} \sum_{(a^+,a^-)\in\mathcal{D}_{\text{Bench}}} \mathbb{1}\big[\pi_\theta(a^+) \succ \pi_\theta(a^-)\big]. \tag{7}$$

Table 2: Results on WEBPRMBENCH with *Pairwise* and *BoN Acc*. ★ denotes our models. Bold numbers indicate the best results, while underlined numbers denote the second best. Our WebArbiter-7B achieves the highest *BoN Acc* across all environments, with an *Avg. BoN Acc* of 77.78%, outperforming the second-best baseline, i.e., Gemini Flash, by 10.85%.

Models	Mind2Web		WebArena		AssistantBench		WorkArena		Avg.	
	Pairwise	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*
LLM-as-judge, Proprietary Language Models										
GPT-4o-mini	80.28	45.69	81.40	61.46	**88.79**	68.97	84.86	66.67	83.83	60.70
GPT-4o	80.60	52.84	79.78	56.06	83.62	58.62	85.69	66.67	82.42	58.55
GPT-5	82.06	62.26	76.89	55.26	76.72	58.62	80.83	65.56	79.12	60.42
Claude Sonnet	81.19	59.60	81.40	62.53	76.92	53.85	88.06	72.22	81.89	62.05
Gemini Flash	81.94	57.86	83.89	67.39	84.62	65.38	**92.36**	77.08	85.70	66.93
DeepSeek-R1	82.18	56.44	81.13	60.38	75.00	51.72	88.61	72.78	81.73	60.33
LLM-as-judge, Open-source Language Models										
Qwen2.5-3B-Instruct	76.89	37.66	69.54	35.04	80.17	48.28	70.56	42.22	74.29	40.80
Qwen2.5-7B-Instruct	79.02	41.00	72.44	40.16	76.72	37.93	78.06	51.67	76.56	42.69
Llama-3-70B-Instruct	79.43	50.96	77.90	56.60	78.85	46.15	87.50	70.14	80.92	55.96
WebPRMs (3B)										
WebShepherd-3B	37.41	21.22	20.33	9.47	36.54	17.24	10.49	2.44	26.19	12.59
★ WebArbiter-3B	93.28	78.75	83.29	56.87	76.72	44.83	84.03	60.56	84.33	60.25
WebPRMs (7B+)										
WebShepherd-8B	72.35	46.68	33.16	12.37	55.77	44.83	35.85	12.68	49.28	29.14
★ WebArbiter-7B	**96.47**	**90.07**	**84.30**	**71.43**	80.17	**72.41**	87.36	**77.22**	**87.08**	**77.78**

BoN Acc. For each instance $(a^+, a^{-1}, \ldots, a^{-Q}) \in \mathcal{D}_{\text{Bench}}$, the WebPRM is considered correct only when a^+ is consistently ranked above all Q distractors, with $Q = 4$ in our benchmark. BoN Acc is:

$$\text{Acc}_{\text{BoN}} = \frac{1}{|\mathcal{D}_{\text{Bench}}|} \sum_{i=1}^{|\mathcal{D}_{\text{Bench}}|} \prod_{q=1}^{Q} \mathbb{1}[\pi_\theta(a_i^+) \succ \pi_\theta(a_i^{-q})]. \tag{8}$$

5 EXPERIMENTS

We conduct comprehensive experiments to evaluate WebArbiter on the reward modeling benchmark WEBPRMBENCH in § 5.1 and on practical applications in § 5.2.

5.1 WEBPRMBENCH

5.1.1 EXPERIMENTAL SETUP

Baselines. We compare WebArbiter against three categories of baselines. (1) Proprietary LLM-as-judge models, including GPT-4o-mini (OpenAI, 2024a), GPT-4o (OpenAI, 2024b), GPT-5 (OpenAI, 2025b), Claude-3.7-Sonnet (Anthropic, 2025), Gemini-2.5-Flash (Comanici et al., 2025), and DeepSeek-R1 (Guo et al., 2025a), which are prompted to act as judges by selecting the preferred action given task context. (2) Open-source LLM-as-judge models, represented by Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct (Qwen et al., 2025), and Llama-3-70B-Instruct (Grattafiori et al., 2024), providing accessible yet competitive instruction-tuned baselines. (3) WebPRMs, where we include WebShepherd (Chae et al., 2025).

Implementation Details. We train WebArbiter on WEBPRM Collection (Chae et al., 2025), which comprises 30k step-level preference pairs drawn from the Mind2Web environment. We use 10k pairs for stage-1 reasoning distillation and the remainder for stage-2 reinforcement learning. Models are initialized from Qwen2.5-3B-Instruct and Qwen2.5-7B-Instruct (Qwen et al., 2025) and fine-tuned with LoRA (Hu et al., 2022). Further implementation details are provided in the Appendix C.

Evaluation Metrics. We report results using two complementary metrics: *Pairwise Accuracy*, which measures correctness on individual preference pairs, and *Best-of-N (BoN) Accuracy*, which evaluates robustness under multi-candidate settings. Detailed definitions are provided in § 4.2.

Table 3: Ablation results on WEBPRMBENCH with Qwen2.5-7B-Instruct as backbone. Best results are in bold and the second best underlined. WEBARBITER, combining principle-guided reasoning distillation with RL, achieves the highest overall performance.

Method	**Mind2Web**		**WebArena**		**AssistantBench**		**WorkArena**		**Avg.**	
	Pairwise	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*
Instruct (Original)	79.02	41.00	72.44	40.16	76.72	37.93	78.06	51.67	76.56	42.69
Instruct + Cold Start RL	**97.63**	**91.38**	67.59	43.40	71.55	34.48	73.33	55.00	77.53	56.07
Instruct + Cold Start RL + Principles	96.42	87.88	84.10	60.65	79.31	55.17	83.19	55.56	85.75	64.81
Instruct + SFT$_{\text{w/o Principles}}$ + RL	94.26	82.39	75.34	49.87	68.97	41.38	78.61	54.44	79.30	57.02
★ WebArbiter	96.47	90.07	**84.30**	**71.43**	**80.17**	**72.41**	**87.36**	**77.22**	**87.08**	**77.78**

5.1.2 MAIN RESULTS

WebArbiter Significantly Outperforms Baselines. As shown in Tab. 2, WebArbiter consistently surpasses both proprietary and open-source LLMs across all environments with *BoN Acc*. While LLM-as-judge methods often maintain moderate *Pairwise Acc*, their performance drops sharply on *BoN Acc*, revealing poor robustness to hard negatives. In contrast, WebArbiter sustains strong results on both metrics, establishing its reliability under realistic multi-candidate settings.

Advantage over the SOTA WebPRM. WebShepherd (Chae et al., 2025) represents the previous SOTA WebPRMs. Trained on the same WEBPRM Collection, which was drawn from the Mind2Web environment, WebArbiter-7B achieves an *Avg. BoN Acc* of 77.78%, surpassing WebShepherd-8B by an absolute gain of 48%. Unlike WebShepherd, which relies on fragile checklists, WebArbiter employs principle-guided reasoning, yielding judgments robust to environment and page variations. Case studies illustrating these differences are provided in Appendix E.

Robust Generalization Across Environments. WebArbiter not only excels in-domain, achieving 96.47% *Pairwise Acc* and 90.07% *BoN Acc* on Mind2Web, but also generalizes across diverse benchmarks. On WebArena, it outperforms the second-best baseline by nearly 4% in *BoN Acc*, gains about 3% on AssistantBench, and still edges out strong baselines on WorkArena with 77.22%. These results confirm that principle-guided reasoning supports both strong in-domain learning and robustness across heterogeneous, noisy, and enterprise-level environments.

5.1.3 ABLATION STUDY

We compare four training variants to disentangle the effects of reinforcement learning, principle guidance, and justification style. *Instruct (Original)* denotes a purely instruction-tuned model without additional optimization. *Cold Start RL* directly applies RL on top of the instruction model. *Cold Start RL + Principles* augments RL with principle prompting during training, enabling explicit principle induction before judgment. $SFT_{w/o\ Principles} + RL$ performs reasoning distillation without principles, followed by RL, thereby testing whether narrative-style justifications alone are sufficient. As shown in Tab. 3, WebArbiter achieves the best performance. Explicit principles anchor judgments to progress, producing stable supervision under multi-candidate web settings.

RL Alone is Unstable Across Web Environments. *Cold Start RL* performs well on in-domain Mind2Web but collapses on out-of-domain benchmarks. This highlights that reward optimization without reasoning distillation struggles in noisy and complex environments.

Principles Enable Cross-Environment Generalization. Augmenting RL with principles boosts *Avg. BoN Acc*, especially in structurally diverse environments such as WebArena and AssistantBench. Principles provide transferable facets for reasoning, reducing reliance on brittle layout cues and improving robustness to web variability.

Reasoning Without Principles is Insufficient. $SFT_{w/o\ Principles} + RL$, i.e., narrative-style justifications alone, improves fluency but lags behind principle-aware settings. This confirms that narrating reasoning chains without principles cannot ensure alignment with true task progress in complex, long-horizon real-world web navigation.

Table 4: Success Rates (%) of trajectory search with GPT-4o-mini and GPT-4o as policy on WebArena-lite. * Results reported from the WebShepherd (Chae et al., 2025). Δ is relative to the *w/o Trajectory Search* baseline. Our WebArbiter consistently achieves the highest gains across both policy models.

Policy	**WebPRM**	**Shopping**	**CMS**	**Reddit**	**GitLab**	**Avg.**	**Δ**
GPT-4o-mini	w/o Trajectory Search*	21.74	22.86	19.05	34.38	24.51	–
	GPT-4o-mini	24.44	22.86	26.32	33.33	26.74	+2.23
	WebShepherd-8B*	26.09	**45.71**	23.81	40.62	34.06	+9.55
	★ WebArbiter-7B	**37.78**	42.86	**36.84**	**46.67**	**41.04**	**+16.53**
GPT-4o	w/o Trajectory Search*	23.91	31.43	28.57	56.25	35.04	–
	GPT-4o-mini	26.67	37.14	42.11	40.00	36.48	+1.44
	WebShepherd-8B*	30.43	**42.86**	47.62	46.88	41.95	+6.91
	★ WebArbiter-7B	**44.44**	**42.86**	**52.63**	**56.67**	**49.15**	**+14.11**

5.2 Reward-Guided Trajectory Search

5.2.1 Experimental Setup and Implementations

Reward-guided trajectory search represents one of the most practical applications of PRMs, as it directly leverages fine-grained step-level supervision to improve decision quality during agent execution. To evaluate WebArbiter in this setting, we conduct experiments on WebArena-Lite[1] (Liu et al., 2024b), which contains diverse, long-horizon tasks such as online shopping and content management, closely reflecting real-world web activities. Performance is measured with Success Rate. Following WebShepherd (Chae et al., 2025), we adopt a Best-of-N sampling strategy: the policy model generates $N = 5$ candidate actions for each step, and WebArbiter selects the most promising one through a Knockout Tournament mechanism (Guo et al., 2025b). We evaluate two policies, GPT-4o-mini (OpenAI, 2024a) and GPT-4o (OpenAI, 2024b).

5.2.2 Analysis

As shown in Tab. 4, WebArbiter achieves substantial average improvements under both policy models, far surpassing baselines. Its advantages arise from three main factors. First, reasoning mitigates spurious correlations that often mislead WebPRMs in domains such as Shopping and Reddit. Gains in Shopping are particularly striking, as tasks require dense semantic retrieval and inference; stronger policies can roll out more promising candidate actions, and WebArbiter's structured reward modeling further amplifies these benefits. Second, in GitLab, tasks frequently allow multiple equivalent paths. WebShepherd is brittle under such variability, whereas WebArbiter leverages reasoning over historical trajectories and current states to evaluate action validity, enabling stronger generalization in dynamic workflows. By contrast, CMS exhibits a more template-driven structure, where actions closely follow standardized patterns. In such cases, checklist-based supervision remains comparatively effective, which narrows the relative performance margin. Overall, WebArbiter's reasoning-first design consistently provides robust, interpretable, and scalable supervision across diverse domains.

6 Conclusion

We presented WebArbiter, a reasoning-first, principle-inducing process reward model that frames reward modeling as structured text generation and produces auditable step-level judgments with rationales. Through reasoning distillation and reinforcement learning, WebArbiter transforms superficial correlations into robust signals that verify genuine task progress, enforce trajectory consistency, and generalize across dynamic websites. To support systematic evaluation, we released WebPRMBench, the first comprehensive evaluation benchmark spanning diverse environments for WebPRMs in web navigation, covering four domains with diverse tasks and fine-grained step-level supervision. Extensive experiments demonstrate SOTA performance on WebPRMBench and substantial improvements in reward-guided trajectory search on WebArena-Lite, establishing principle-guided reasoning WebPRMs as a robust and interpretable foundation for scalable web agents.

[1] We did not have access to the MAP domain during this work and therefore excluded it from our experiments.

REFERENCES

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

Adept. Act-1: Transformer for actions. `adept.ai/blog/act-1/` , 2022.

Anthropic. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. `anthropic.com/news/3-5-models-and-computer-use`, 2024a.

Anthropic. Introducing computer use, a new claude 3.5 sonnet, and claude 3.5 haiku. `https://www.anthropic.com/news/3-5-models-and-computer-use`, October 2024b.

Anthropic. Claude 3.7 sonnet and claude code. `anthropic.com/news/claude-3-7-sonnet`, 2025.

David Anugraha, Zilu Tang, Lester James V Miranda, Hanyang Zhao, Mohammad Rifqi Farhansyah, Garry Kuwanto, Derry Wijaya, and Genta Indra Winata. R3: Robust rubric-agnostic reward models. *arXiv preprint arXiv:2505.13388*, 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, 2025. URL `https://arxiv.org/abs/2407.05291`.

Hyungjoo Chae, Sunghwan Kim, Junhee Cho, Seungone Kim, Seungjun Moon, Gyeom Hwangbo, Dongha Lim, Minjin Kim, Yeonjun Hwang, Minju Gwak, Dongwook Choi, Minseok Kang, Gwanhoon Im, ByeongUng Cho, Hyojun Kim, Jun Hee Han, Taeyoon Kwon, Minju Kim, Beong woo Kwak, Dongjin Kang, and Jinyoung Yeo. Web-shepherd: Advancing prms for reinforcing web agents, 2025. URL `https://arxiv.org/abs/2505.15277`.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, Hanghang Tong, and Heng Ji. Rm-r1: Reward modeling as reasoning, 2025a. URL `https://arxiv.org/abs/2505.02387`.

Xiusi Chen, Gaotang Li, Ziqi Wang, Bowen Jin, Cheng Qian, Yu Wang, Hongru Wang, Yu Zhang, Denghui Zhang, Tong Zhang, et al. Rm-r1: Reward modeling as reasoning. *arXiv preprint arXiv:2505.02387*, 2025b.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

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

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

Yao Fu, Dong-Ki Kim, Jaekyeom Kim, Sungryull Sohn, Lajanugen Logeswaran, Kyunghoon Bae, and Honglak Lee. Autoguide: Automated generation and selection of state-aware guidelines for large language model agents. *arXiv preprint arXiv:2403.08978*, 2024.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

Jiaxin Guo, Zewen Chi, Li Dong, Qingxiu Dong, Xun Wu, Shaohan Huang, and Furu Wei. Reward reasoning model, 2025b. URL https://arxiv.org/abs/2505.14674.

Ilgee Hong, Changlong Yu, Liang Qiu, Weixiang Yan, Zhenghao Xu, Haoming Jiang, Qingru Zhang, Qin Lu, Xin Liu, Chao Zhang, et al. Think-rm: Enabling long-horizon reasoning in generative reward models. *arXiv preprint arXiv:2505.16265*, 2025.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022.

Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. Understanding the planning of llm agents: A survey, 2024. URL https://arxiv.org/abs/2402.02716.

Geunwoo Kim, Pierre Baldi, and Stephen McAleer. Language models can solve computer tasks. *Advances in Neural Information Processing Systems*, 36, 2024.

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

Jing Yu Koh, Stephen McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language model agents, 2025. URL https://arxiv.org/abs/2407.01476.

Manling Li, Shiyu Zhao, Qineng Wang, Kangrui Wang, Yu Zhou, Sanjana Srivastava, Cem Gokmen, Tony Lee, Erran Li Li, Ruohan Zhang, et al. Embodied agent interface: Benchmarking llms for embodied decision making. *Advances in Neural Information Processing Systems*, 37:100428–100534, 2024.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*, 2023.

Chris Yuhao Liu, Liang Zeng, Jiacai Liu, Rui Yan, Jujie He, Chaojie Wang, Shuicheng Yan, Yang Liu, and Yahui Zhou. Skywork-reward: Bag of tricks for reward modeling in llms, 2024a. URL https://arxiv.org/abs/2410.18451.

Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, Hanyu Lai, Xinyi Liu, Hanlin Zhao, Jiadai Sun, Xinyue Yang, Yu Yang, Zehan Qi, Shuntian Yao, Xueqiao Sun, Siyi Cheng, Qinkai Zheng, Hao Yu, Hanchen Zhang, Wenyi Hong, Ming Ding, Lihang Pan, Xiaotao Gu, Aohan Zeng, Zhengxiao Du, Chan Hee Song, Yu Su, Yuxiao Dong, and Jie Tang. Visualagentbench: Towards large multimodal models as visual foundation agents, 2024b. URL https://arxiv.org/abs/2408.06327.

Yantao Liu, Zijun Yao, Rui Min, Yixin Cao, Lei Hou, and Juanzi Li. Pairjudge rm: Perform best-of-n sampling with knockout tournament, 2025. URL https://arxiv.org/abs/2501.13007.

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

Kaixin Ma, Hongming Zhang, Hongwei Wang, Xiaoman Pan, and Dong Yu. Laser: Llm agent with state-space exploration for web navigation. *arXiv preprint arXiv:2309.08172*, 2023.

Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. Generative reward models, 2024. URL https://arxiv.org/abs/2410.12832.

Bingchen Miao, Yang Wu, Minghe Gao, Qifan Yu, Wendong Bu, Wenqiao Zhang, Yunfei Li, Siliang Tang, Tat-Seng Chua, and Juncheng Li. Boosting virtual agent learning and reasoning: A step-wise, multi-dimensional, and generalist reward model with benchmark, 2025. URL https://arxiv.org/abs/2503.18665.

OpenAI. Gpt-4o mini: advancing cost-efficient intelligence. openai.com/gpt-4o-mini, 2024a.

OpenAI. Gpt-4o. platform.openai.com/gpt-4o, 2024b.

OpenAI. Introducing operator. openai.com/introducing-operator, 2025a.

OpenAI. Gpt-5 is here. openai.com/gpt-5, 2025b.

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

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

Junsoo Park, Seungyeon Jwa, Meiying Ren, Daeyoung Kim, and Sanghyuk Choi. Offsetbias: Leveraging debiased data for tuning evaluators, 2024. URL https://arxiv.org/abs/2407.06551.

Archiki Prasad, Alexander Koller, Mareike Hartmann, Peter Clark, Ashish Sabharwal, Mohit Bansal, and Tushar Khot. Adapt: As-needed decomposition and planning with language models. *arXiv preprint arXiv:2311.05772*, 2023.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Wenyi Zhao, Yu Yang, Xinyue Yang, Jiadai Sun, Shuntian Yao, Tianjie Zhang, Wei Xu, Jie Tang, and Yuxiao Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning, 2025. URL https://arxiv.org/abs/2411.02337.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi 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, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. Qwen2.5 technical report, 2025. URL https://arxiv.org/abs/2412.15115.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024. URL https://arxiv.org/abs/2402.03300.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems*, 36, 2024.

Paloma Sodhi, S. R. K. Branavan, Yoav Artzi, and Ryan McDonald. Step: Stacked llm policies for web actions, 2024.

Haotian Sun, Yuchen Zhuang, Lingkai Kong, Bo Dai, and Chao Zhang. Adaplanner: Adaptive planning from feedback with language models. *Advances in Neural Information Processing Systems*, 36, 2024.

Heyi Tao, Sethuraman TV, Michal Shlapentokh-Rothman, Derek Hoiem, and Heng Ji. Webwise: Web interface control and sequential exploration with large language models. *arXiv preprint arXiv:2310.16042*, 2023.

Jonathan Uesato, Nate Kushman, Ramana Kumar, Francis Song, Noah Siegel, Lisa Wang, Antonia Creswell, Geoffrey Irving, and Irina Higgins. Solving math word problems with process-and outcome-based feedback. *arXiv preprint arXiv:2211.14275*, 2022.

Peiyi Wang, Lei Li, Zhihong Shao, RX Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. *arXiv preprint arXiv:2312.08935*, 2023a.

Tianlu Wang, Ilia Kulikov, Olga Golovneva, Ping Yu, Weizhe Yuan, Jane Dwivedi-Yu, Richard Yuanzhe Pang, Maryam Fazel-Zarandi, Jason Weston, and Xian Li. Self-taught evaluators, 2024a. URL `https://arxiv.org/abs/2408.02666`.

Zhilin Wang, Yi Dong, Jiaqi Zeng, Virginia Adams, Makesh Narsimhan Sreedhar, Daniel Egert, Olivier Delalleau, Jane Polak Scowcroft, Neel Kant, Aidan Swope, and Oleksii Kuchaiev. Helpsteer: Multi-attribute helpfulness dataset for steerlm, 2023b. URL `https://arxiv.org/abs/2311.09528`.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. Helpsteer2: Open-source dataset for training top-performing reward models, 2024b. URL `https://arxiv.org/abs/2406.08673`.

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, 2025. URL `https://arxiv.org/abs/2505.16421`.

Tianhao Wu, Weizhe Yuan, Olga Golovneva, Jing Xu, Yuandong Tian, Jiantao Jiao, Jason Weston, and Sainbayar Sukhbaatar. Meta-rewarding language models: Self-improving alignment with llm-as-a-meta-judge, 2024. URL `https://arxiv.org/abs/2407.19594`.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Dingwen Yang, Chenyang Liao, Xin Guo, Wei He, et al. Agentgym: Evolving large language model-based agents across diverse environments. *arXiv preprint arXiv:2406.04151*, 2024.

Ziyi Ye, Xiangsheng Li, Qiuchi Li, Qingyao Ai, Yujia Zhou, Wei Shen, Dong Yan, and Yiqun Liu. Learning llm-as-a-judge for preference alignment. In *The Thirteenth International Conference on Learning Representations*, 2025.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. Assistantbench: Can web agents solve realistic and time-consuming tasks?, 2024. URL `https://arxiv.org/abs/2407.15711`.

Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. Generative verifiers: Reward modeling as next-token prediction. *arXiv preprint arXiv:2408.15240*, 2024.

Yao Zhang, Chenyang Lin, Shijie Tang, Haokun Chen, Shijie Zhou, Yunpu Ma, and Volker Tresp. Swarmagentic: Towards fully automated agentic system generation via swarm intelligence. *arXiv preprint arXiv:2506.15672*, 2025a.

Yao Zhang, Zijian Ma, Yunpu Ma, Zhen Han, Yu Wu, and Volker Tresp. Webpilot: A versatile and autonomous multi-agent system for web task execution with strategic exploration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23378–23386, 2025b.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*, 2025c.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in neural information processing systems*, 36:46595–46623, 2023a.

Longtao Zheng, Rundong Wang, Xinrun Wang, and Bo An. Synapse: Trajectory-as-exemplar prompting with memory for computer control. In *The Twelfth International Conference on Learning Representations*, 2023b.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, Bangkok, Thailand, 2024. Association for Computational Linguistics. URL `http://arxiv.org/abs/2403.13372`.

Enyu Zhou, Guodong Zheng, Binghai Wang, Zhiheng Xi, Shihan Dou, Rong Bao, Wei Shen, Limao Xiong, Jessica Fan, Yurong Mou, Rui Zheng, Tao Gui, Qi Zhang, and Xuanjing Huang. Rmb: Comprehensively benchmarking reward models in llm alignment, 2025. URL `https://arxiv.org/abs/2410.09893`.

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

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents, 2024. URL `https://arxiv.org/abs/2307.13854`.

CONTENTS

A NOTATION SUMMARY

For clarity, we summarize the main notations used throughout this paper:

- $\mathcal{E}$: web environment, defined by state space $\mathcal{S}$, action space $\mathcal{A}$, and observation space $\mathcal{O}$.
- T: state transition function $T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$.
- $\mathcal{I}$: task instruction.
- s_p, o_p, a_p: state, observation, and action at step p.
- c_p: reasoning trace associated with action a_p.
- $c_{<p}$: reasoning traces of all previously executed actions.
- $\tau = (a_1, \ldots, a_P)$: trajectory of length P.
- $j = (j_1, \ldots, j_L)$: structured justification of length L, consisting of explicit reasoning and a final verdict.
- π_θ: WebArbiter model parameterized by θ.
- $\hat{y}$: predicted preference verdict.
- $\mathcal{D}_{\text{Train}}, \mathcal{D}_{\text{SFT}}, \mathcal{D}_{\text{RL}}$: training datasets for supervised distillation and reinforcement learning.
- $\mathcal{D}_{\text{Bench}}$: evaluation dataset for WEBARBITER-BENCH.
- a^+: action from a successful trajectory (positive label).
- a^- or a^{-q}: rejected action ($q = 1, \ldots, Q$).
- Q: number of distractors in WEBARBITER-BENCH (set to 4).
- M: dataset size (training or evaluation).
- $R(x, \hat{y})$: verifiable reward signal.

B EXAMPLE OF PREFERENCE DATASET

To illustrate the construction of a preference pair, consider a task instruction from WEBARENA:

- **Instruction** $\mathcal{I}$: “Find the 2026 conference submission page on the ICLR website.‘’
- **Current observation** o_p: [356] banner ’header’, role=’banner’ [359] link ’Home’ [380] button ’Select Year (2026)’ [386] button ’Dates’ [391] button ’Calls’ [396] button ’Guides’ [401] button ’Organization’ [403] heading ’ICLR 2026’
- **History** $(a_{<p}, c_{<p})$:
  - a_1: Search "ICLR", c_1: “I need to find the official ICLR website first. Let me search for ’ICLR’ to locate it.‘’
  - a_2: Click link "ICLR homepage", c_2: “I can see the ICLR official website link in the search results. Clicking on it will take me to the ICLR homepage where I can find the conference submission information.‘’
- **Candidate actions and reasoning traces**:
  - (a_p^1, c_p^1): Click link "Call for Papers" ; c_p^1: “I can see a ’Call for Papers’ link on the ICLR homepage. This link would likely lead to the submission details page, which should contain information about the 2026 conference submission process that I’m looking for.‘’
  - (a_p^2, c_p^2): Click "About" link; c_p^2: “I can see an ’About’ link on the ICLR homepage. Since I need to find the 2026 conference submission page, the ’About’ section might contain conference overview information including links to submission details or important dates for the 2026 conference.‘’
- **Label** y: a_p^1 is preferred.

This example is represented in the dataset as:

$$(\mathcal{I},\ o_p,\ a_{<p},\ c_{<p},\ (a_p^1, c_p^1),\ (a_p^2, c_p^2),\ y = a_p^1).$$

C Training Details

All training is conducted on 8 NVIDIA A100-80GB GPUs with fixed random seeds. Our training framework is bead on LLama-Factory (Zheng et al., 2024) and VERL (Sheng et al., 2024)

Distillation Stage. We train the model for 5 epochs with a learning rate of 8e-4, using LoRA with a rank of 128. We apply a cosine learning rate scheduler with a warmup ratio of 0.1. We set the batch size to 256 and the maximum sequence length to 8,192 tokens.

RLVR Stage. We employ the VERL framework for GRPO training. The learning rate is set to 7.0×10^{-6}. The training uses a fixed batch size of 1,024 with mini-batch size of 128, and adopts Fully Sharded Data Parallel (FSDP) for enhanced memory efficiency. For rollout generation, we deploy vLLM with tensor parallelism of 4 and GPU memory utilization limited to 0.4. Response sampling uses standard parameters (temperature=1.0, top-p=1.0), generating 7 candidate responses per prompt. We apply KL regularization with a coefficient of 1.0×10^{-3} and clip ratio of 0.2. The maximum input sequence length is 8,192 tokens, and the maximum response length is 4,096 tokens.

D Prompt Repository

WebArbiter

```
You are a skilled expert at evaluating assistant responses. You
    should evaluate given responses based on the given judging
    criteria.\n Given the context of the conversation and two
    responses from the Assistant, you need to determine the better
    response. Provide an overall comprehensive comparison upon them.
#### Intent ####
{intent}
#### AXTREE ####
Note: [bid] is the unique alpha-numeric identifier at the
    beginning of lines for each element in the AXTree. Always use
    bid to refer to elements in your actions.
{observation}
#### Trajectory ####
Note: The trajectory contains the sequence of previous actions and
    their corresponding thoughts. Each entry reflects the agent’s
    internal reasoning (‘thought‘) and the concrete operation it
    performed (‘action‘).
{trajectory}
#### start url ####
{start_url}
#### current url ####
The URL provides clues about the user’s position in the
    application flow. Use both the path and query parameters to
    infer page type (e.g., homepage, search results, product
    detail, cart, checkout).
{current_url}
#### Assistant Responses ####
[The Begin of Response 1]
THOUGHT:
{thought1}
ACTION:
{action1}
[The End of Response 1]
[The Begin of Response 2]
THOUGHT:
{thought2}
ACTION:
{action2}
[The End of Response 2]
```

```
### Output Instructions ###
Format your output strictly using the following XML-style tags:
<State>Summarize the current state based on the URL, AXTree, and
    previous actions. Include what page the user is currently on,
    and what relevant UI elements or information are
    visible.</State>
<Criteria>Other potential criteria specific to the query and the
    context, and the weights of each criteria.</Criteria>
<Analysis>Compare Response 1 and Response 2 in detail according to
    the <State> and <Criteria>.</Analysis>
<Answer>Response 1 or Response 2</Answer>
Rules for <Answer>:
- If Response 1 is better, output exactly: <Answer>Response
    1</Answer>
- If Response 2 is better, output exactly: <Answer>Response
    2</Answer>
Important Notes:
- Be objective and base your evaluation strictly on the content of
    the responses.
- Do not let the response order, length bias your judgment.
```

E Case Study: WebArbiter vs. WebShepherd

To further illustrate the differences between WebArbiter and WebShepherd, we present a representative example from WebArena. The task instruction is:

> *"What is the rating of* Ugreen lightning to 3.5mm cable*? Round to the nearest whole number."*

At the current step, the agent observes a search-results page listing the target product. The snippet already shows a "65%" rating in the result list, but the product detail page has not been opened yet. Two candidate actions are considered:

- a_p^1: `send_msg_to_user` with "65%".
  c_p^1: "The list view already shows a 65% rating, so answer directly."
- a_p^2: Click the target product entry to open its detail page, then extract and (if needed) round the rating.
  c_p^2: "Verify the rating on the product page (correct product, correct field) before responding."

WebShepherd. WebShepherd evaluates candidates using *checklist*-style templates that are *precompiled* before the next observation. These checklists typically include predicates such as "verify on the product page"; hence, even when a rating is already visible in an earlier search-results snippet, the checklist *still requires verification*, and thus tends to favor a_p^2. When the actual page deviates from the ex-ante forecast (e.g., an early results page surfaces), the precompiled predicates become stale: they enforce a verification path that may be unnecessary or even brittle under interstitials or layout drift.

WebArbiter. WebArbiter derives principles such as *"answer directly when the objective's required field is already unambiguously satisfied by the current observation," "ensure correct rounding,"* and *"avoid redundant navigation when the answer is already grounded."* It performs *dynamic expectation alignment*: (i) it forms expectation about what evidence is needed, (ii) compares the actual page with that expectation, and (iii) revises principle weights and candidate scoring accordingly. Concretely, upon seeing a clear "65%" rating in the snippet, it downweights "must verify on product page" and upweights "answer directly with proper rounding," issuing a preference verdict for a_p^1 and correctly completing the task with minimal steps.

Discussion. This case illustrates a key limitation of precompiled, open-loop checklists: they conflate *procedural requirements* ("must navigate to detail page") with *goal satisfaction* and thus underperform when early observations already satisfy the objective. In contrast, WebArbiter grounds decisions in explicit, principle-guided reasoning *and* closed-loop, dynamic expectation alignment (predict → observe → compare → revise), enabling it to act on already-sufficient evidence and remain robust to goal–observation mismatches.

F Benchmark Construction

Positive samples. We construct WebPRMBench using the successful trajectories from AgentRewardBench, a human-verified evaluation suite that aggregates over a thousand trajectories generated by multiple LLM-based web agents across diverse real-world environments. Each trajectory in AgentRewardBench is annotated for success and execution quality by expert annotators, providing a reliable source of environment-grounded optimal behavior. From this dataset, we select only those trajectories that complete each task with the minimum number of steps. Each trajectory is independently reviewed by annotators to ensure monotonic progress and to verify that no redundant or detour actions are present. When deviations are identified, annotators revise the trajectory to recover the shortest valid execution path consistent with successful task completion. For consistency, missing reasoning traces are completed to ensure that every state–action pair is paired with a coherent rationale. The resulting actions from these validated minimal-step trajectories serve as positive labels, reflecting actions empirically verified to succeed in the real web environment.

Negative samples. For each state, we sample four alternative actions and their associated reasoning from a diverse ensemble of policy models, covering both open-source and proprietary LLMs. The pool includes high-capacity instruction-tuned models such as Qwen2.5-7B / 72B-Instruct (Qwen et al., 2025), Llama-3.3-8B / 70B-Instruct (Grattafiori et al., 2024), as well as frontier commercial models including GPT-4o / 4o-mini (OpenAI, 2024a;b), Claude-3.5-Haiku / Claude-3.7-Sonnet (Anthropic, 2024b; 2025), and Gemini-2.5-Flash / Gemini-2.5-Pro (Comanici et al., 2025). This ensures that alternative actions exhibit broad stylistic and policy diversity rather than reflecting any single model's reasoning behavior. Since alternative actions may still succeed under certain web interfaces, we apply a rule-based filtering procedure to remove actions that remain potentially valid. We retain only actions that are clearly invalid or non-progressing, ensuring that negative samples correspond to failures under the actual environment dynamics rather than differences in reasoning style. To ensure consistency and avoid false negatives, the filtered actions are manually reviewed, and any remaining actions that appear potentially valid are discarded. If more than four valid rejected actions remain after filtering, we randomly sample a subset to maintain a consistent number of action pairs per instance. All rationales are truncated to a fixed length to reduce formatting noise while preserving semantic content.

The final benchmark consists of 1,287 step-level preference pairs across four environments, each containing one environment-verified positive action and four rule-filtered negative alternatives.

G General-Domain Generative Reward Models and Their Transferability to Web Trajectories

This section presents additional evaluations of general-domain generative reward models, including RM-R1 (Chen et al., 2025b), RRM (Guo et al., 2025b), Think-RM (Hong et al., 2025), and R3 (Anugraha et al., 2025). Although these models represent SOTA approaches within preference-based reward modeling, they are trained primarily on static QA, dialogic reasoning, mathematical problem solving, and related preference datasets. Consequently, their training objectives do not incorporate key structural properties of interactive web environments, such as AXTree-grounded observations.

To enable comparison, we adapt each model's preference interface for step-level scoring on WebPRMBench. As shown in Tab. 5, all general-domain reward models achieve substantially lower *Avg. Pairwise Acc* and *Avg. BoN Acc* than WebArbiter. Because these models are trained exclusively on static, text-only preference corpora, their learned reward functions emphasize linguistic plausibility and abstract reasoning rather than the procedural validity required for web actions. They do not model the environment-dependent factors that govern real web interaction, such as action executability under the current state, UI structural changes, and whether an action produces measurable task progress, making them fundamentally mismatched to process-level reward modeling. Overall, these results

Table 5: Results on WEBPRMBENCH with *Pairwise* and *BoN Acc*. ★ denotes our models. Bold numbers indicate the best results, while underlined numbers denote the second best. Our WebArbiter-7B achieves the highest *BoN Acc* across all environments, with an *Avg. BoN Acc* of 77.78%, outperforming the second-best baseline, i.e., Gemini Flash, by 10.85%.

Models	**Mind2Web**		**WebArena**		**AssistantBench**		**WorkArena**		**Avg.**	
	Pairwise	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*	*Pairwise*	*BoN*
LLM-as-judge, Proprietary Language Models										
GPT-4o-mini	80.28	45.69	81.40	61.46	**88.79**	68.97	84.86	66.67	83.83	60.70
GPT-4o	80.60	52.84	79.78	56.06	83.62	58.62	85.69	66.67	82.42	58.55
GPT-5	82.06	62.26	76.89	55.26	76.72	58.62	80.83	65.56	79.12	60.42
Claude Sonnet	81.19	59.60	81.40	62.53	76.92	53.85	88.06	72.22	81.89	62.05
Gemini Flash	81.94	57.86	83.89	67.39	84.62	65.38	**92.36**	77.08	85.70	66.93
DeepSeek-R1	82.18	56.44	81.13	60.38	75.00	51.72	88.61	72.78	81.73	60.33
LLM-as-judge, Open-source Language Models										
Qwen2.5-3B-Instruct	76.89	37.66	69.54	35.04	80.17	48.28	70.56	42.22	74.29	40.80
Qwen2.5-7B-Instruct	79.02	41.00	72.44	40.16	76.72	37.93	78.06	51.67	76.56	42.69
Llama-3-70B-Instruct	79.43	50.96	77.90	56.60	78.85	46.15	87.50	70.14	80.92	55.96
Generative RMs										
RM-R1-Qwen2.5-Instruct-7B	69.11	23.77	63.68	20.22	73.28	34.48	62.36	18.33	67.11	24.20
RRM-7B	82.28	48.51	74.60	48.25	86.21	68.97	77.78	55.56	80.22	55.32
Think-RM-3B	70.93	28.57	60.44	26.42	62.93	24.14	68.75	33.89	65.76	28.25
Think-RM-8B	75.45	45.54	74.33	48.79	77.59	51.72	83.47	63.33	77.71	52.35
R3-Qwen3-4B-LoRA-4k	78.32	43.56	76.48	48.25	83.19	62.78	83.19	62.78	80.30	54.34
WebPRMs (3B)										
WebShepherd-3B	37.41	21.22	20.33	9.47	36.54	17.24	10.49	2.44	26.19	12.59
★ WebArbiter-3B	93.28	78.75	83.29	56.87	76.72	44.83	84.03	60.56	84.33	60.25
WebPRMs (7B+)										
WebShepherd-8B	72.35	46.68	33.16	12.37	55.77	44.83	35.85	12.68	49.28	29.14
★ WebArbiter-7B	**96.47**	**90.07**	**84.30**	**71.43**	80.17	**72.41**	87.36	**77.22**	**87.08**	**77.78**

show that general-domain Generative RMs do not generalize to procedural, state-dependent web tasks and highlight the need for domain-grounded reasoning and environment-verified supervision in WebPRMs.

$\mathcal{I}$ **Instruction**

What is the rating of Ugreen lightning to 3.5mm cable? Round to the nearest whole number.

$(a_{<p}, r_{<p})$ **Trajectory**

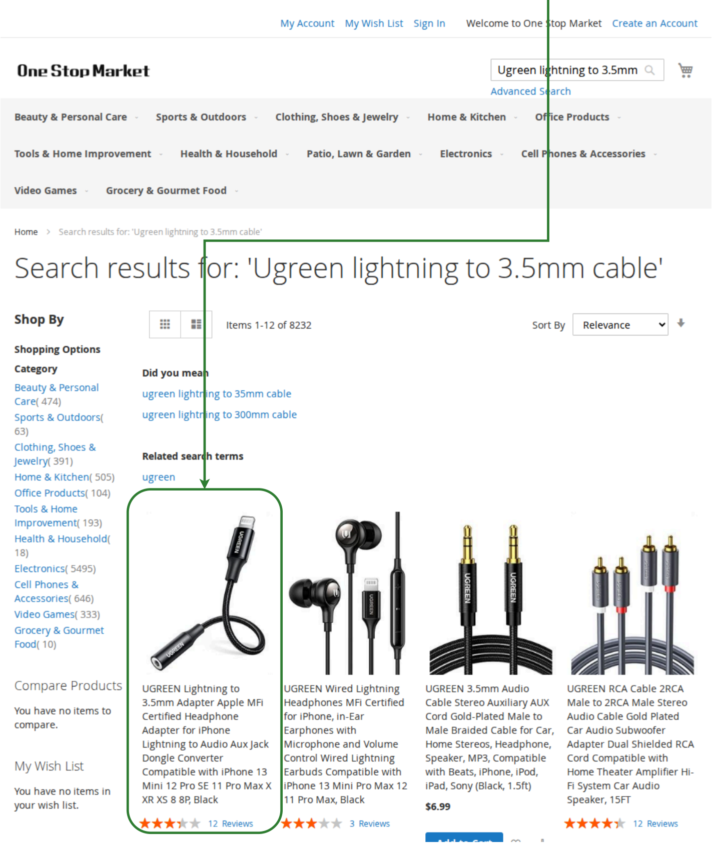
Thought: On One Stop Market's homepage, the search ...
Action: *type [2826] "Ugreen lightning to 3.5mm cable"*
Thought: The current Page is on ....
...

(a_p^1, r_p^1) **Action Candidate 1**

Thought: The page lists a 65% rating on the search results snippet.
`Action: send_msg_to_user ["65%"]`

(a_p^2, r_p^2) **Action Candidate 2**

Thought: Open the product page to verify the rating, then round to nearest whole.
`Action: click [6368]`



My Account My Wish List Sign In Welcome to One Stop Market Create an Account

One Stop Market

Ugreen lightning to 3.5mm

Advanced Search

Beauty & Personal Care · Sports & Outdoors · Clothing, Shoes & Jewelry · Home & Kitchen · Office Products · Tools & Home Improvement · Health & Household · Patio, Lawn & Garden · Electronics · Cell Phones & Accessories · Video Games · Grocery & Gourmet Food

Home > Search results for: 'Ugreen lightning to 3.5mm cable'

Search results for: 'Ugreen lightning to 3.5mm cable'

Shop By

Shopping Options

Category

Beauty & Personal Care(474)
Sports & Outdoors(63)
Clothing, Shoes & Jewelry(391)
Home & Kitchen(505)
Office Products(104)
Tools & Home Improvement(193)
Health & Household(18)
Electronics(5495)
Cell Phones & Accessories(646)
Video Games(333)
Grocery & Gourmet Food(10)

Compare Products

You have no items to compare.

My Wish List

You have no items in your wish list.

Items 1-12 of 8232 Sort By Relevance

Did you mean

ugreen lightning to 35mm cable
ugreen lightning to 300mm cable

Related search terms

ugreen

UGREEN Lightning to 3.5mm Adapter Apple MFi Certified Headphone Adapter for iPhone Lightning to Audio Aux Jack Dongle Converter Compatible with iPhone 13 Mini 12 Pro SE 11 Pro Max X XR XS 8 8P, Black — 12 Reviews

UGREEN Wired Lightning Headphones MFi Certified for iPhone, in-Ear Earphones with Microphone and Volume Control Wired Lightning Earbuds Compatible with iPhone 13 Mini Pro Max 12 11 Pro Max, Black — 3 Reviews

UGREEN 3.5mm Audio Cable Stereo Auxiliary AUX Cord Gold-Plated Male to Male Braided Cable for Car, Home Stereos, Headphone, Speaker, MP3, Compatible with Beats, iPhone, iPod, iPad, Sony (Black, 1.5ft) — $6.99

UGREEN RCA Cable 2RCA Male to 2RCA Male Stereo Audio Cable Gold Plated Car Audio Subwoofer Adapter Dual Shielded RCA Cord Compatible with Home Theater Amplifier Hi-Fi System Car Audio Speaker, 15FT — 12 Reviews

Websheperd Analysis

✗**Action Candidate 1**

Checklist 1: Search for Product (Yes)
Results for 'Ugreen lightning to 3.5mm cable' are visible; the list view already shows a 65% rating, so I can answer directly.

Checklist 2: Extract Rating (No)
Outputs '65%' from the results snippet without locating a rating field or confirming formatting on the product page.

Checklist 3: Verify Page (No)
Does not open the product detail page; no on-page verification or source evidence is performed.

✓ **Action Candidate 2**

Checklist 1: Search for Product (Yes)
Results are visible, but the correct next step is to open the product page to verify the rating.

Checklist 2: Extract Rating (In Progress)
Plan to locate the rating on the product page and then round, the action moves toward the rating field, but the value has not yet been located or parsed at this step.

Checklist 3: Verify Page (No)
Does not open the product detail page; no on-page verification or source evidence is performed.

WebAribiter Analysis

Principle

1. Correctness of the rating value (60%)
2. Correct rounding to the nearest whole number (20%)
3. Reference to the right product (not a different product) (10%)
4. Action appropriateness (should not click a wrong element) (10%)

Comparing against the principle:

1. Correct rating – both fine.
2. Rounding – both fine.
3. Correct product
   - Action Candidate1 implicitly correct
   - Action Candidate 2 may be off.
4. Action appropriateness
   - Action Candidate 1 minimal

Conclusion:

✓ **Action Candidate 1** — more accurate and progressive (Chosen)

✗ **Action Candidate 2** — may click the wrong element

Figure 3: Case study on product rating in WebArena-Lite. The snippet shows "65%" before opening the product page. WebShepherd, constrained by fixed checklists, may enforce redundant verification. WebArbiter, using principle-guided reasoning, recognizes the snippet as sufficient and selects the correct action.