FROM TRAJECTORIES TO GROUNDED PREFERENCES: PROCESS PREFERENCE SYNTHESIS VIA INTERACTION ELEMENT GRAPHS FOR WEB PRMS

Anonymous authors
Paper under double-blind review

ABSTRACT

Comparative Process Reward Models (PRMs) provide critical step-level guidance for autonomous web agents by evaluating state-conditioned preferences between candidate actions. However, existing preference training data synthesized via multi-policy sampling suffers from a severe scarcity of Grounded Minimal Contrastive Pairs (GMCPs)—where competing candidates target genuine on-page elements with identical action types. In representative baselines preference data (namely, WebArbiter), GMCPs account for merely 24.19%, biasing PRMs during training to rely on shallow shortcuts (such as element hallucinations and action-type mismatches) rather than acquiring genuine contextual decision semantics. To address these challenges, we propose SURFPRM, a graph-guided process preference synthesis framework for comparative Web PRMs. SURFPRM structures web demonstrations into a persistent *Interaction Element Graph* that acts as an environment-grounded negative action proposal mechanism, systematically synthesizing contrastive negative actions across spatial, temporal, and spatiotemporal confusion axes. This elevates the GMCP proportion from 24.19% to 74.60%, producing the curated SURFPRM-DATA dataset. Across six open-source backbones (3B to 9B parameters), PRMs trained on SURFPRM-DATA outperform baseline-trained models on average on WEBPRMBENCH and rival leading proprietary LLMs. In downstream reward-guided trajectory search on WEBARENA-LITE, SURFPRM provides step-level guidance for both GPT-4o (+14.21%) and GPT-4o-mini (+12.83%) policies, yielding substantial improvements in complex web task success rates. Our code and dataset are anonymously available at https://anonymous.4open.science/r/surfPRM-6188/.

1 INTRODUCTION

Autonomous web agents powered by Large Language Models (LLMs) have demonstrated substantial potential in automating complex, multistep workflows across dynamic websites (Deng et al., 2023; Zhou et al., 2024; He et al., 2024). Unlike static text environments, web interaction presents distinct challenges: observation spaces are vast and heterogeneous, action spaces are combinatorially large, and transitions are frequently irreversible (e.g., submitting forms, committing transactions, or navigating stateful applications) (Zhou et al., 2024). In such settings, unguided greedy decoding frequently leads to irreversible dead ends. Step-level verification and test-time search mechanisms are therefore valuable to guide agents toward successful task completion (Koh et al., 2025; Yang et al., 2026).

Conventional Outcome Reward Models (ORMs) (Ouyang et al., 2022; Rafailov et al., 2023) evaluate agent trajectories solely upon completion, providing delayed and sparse scalar rewards. This outcome-centric evaluation creates acute credit assignment ambiguity: an ORM cannot identify which intermediate step derailed execution, nor can it detect when an agent enters a dead end mid-trajectory. Process Reward Models (PRMs) (Uesato et al., 2022; Lightman et al., 2024; Wang et al., 2024b) address this by verifying steps incrementally. In the web domain, state-of-the-art Web PRMs adopt a *comparative, generative* formulation (Zhang et al., 2026a). Given state $s_t = (I, O_t, H_t, u_t)$—instruction I, linearized Accessibility Tree (AXTree) observation O_t, action history H_t, and URL u_t—a comparative Web PRM evaluates candidate actions (a_t^1, a_t^2) condi-

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.

tioned on s_t: $f_\theta(s_t, a_t^1, a_t^2) \rightarrow (j_t, \hat{y}_t)$ with $\hat{y}_t \in \{1, 2\}$, where j_t is an explicit *judgment rationale* detailing causal distinctions between candidates, and $\hat{y}_t$ indicates the preferred candidate index ($\hat{y}_t = 1 \iff a_t^1 \succ a_t^2$). Rather than predicting uncalibrated absolute scores across disparate web layouts, the comparative PRM learns relative preferences between competing actions under the exact same state (Chae et al., 2025; Zhang et al., 2026a).

To train such comparative verifiers, supervision takes the form of process preference tuples $d_t = (s_t, a_t^+, a_t^-, j_t^*, y_t^*)$. In preference synthesis, successful demonstration trajectories naturally provide execution steps that serve as *reference positives* (a_t^+). Given demonstration steps as reference positives, constructing environment-grounded and informative alternatives (a_t^-) becomes a central challenge in comparative Web PRM data synthesis. Previous pipelines (Zhang et al., 2026a; Chae et al., 2025) typically sample alternative actions a_t^- via multi-policy models and prompt teacher LLMs to synthesize comparative rationales. However, existing preference training data synthesized via multi-policy sampling suffers from a severe scarcity of Grounded Minimal Contrastive Pairs (GMCPs)—where competing candidates target genuine on-page elements with identical action types. In representative baseline data (namely, WebArbiter (Zhang et al., 2026a)), GMCPs account for merely 24.19%, with the remaining majority dominated by shallow shortcuts: (1) *Element Hallucination*, targeting element identifiers (BIDs) absent from O_t; and (2) *Action-Type Mismatch*, diverging in action modality (e.g., valid `click` vs. invalid `fill`). This severe deficit of minimal contrastive pairs biases PRMs during training to rely on superficial cues rather than acquiring genuine contextual decision semantics. Naively filtering multi-policy sampled actions against active AXTree elements fails to resolve this dilemma: it discards the vast majority of generated candidates—incurring substantial LLM sampling costs with low yield of usable pairs—while failing to exploit the rich interaction transitions already embedded in demonstration trajectories.

To address these challenges, we propose SURFPRM, a *Graph-guided Process Preference Synthesis* framework. We aggregate web interaction demonstrations into a persistent *Interaction Element Graph* $G = (V, E_s, E_\tau)$, where nodes represent persistent interactive elements, E_s captures AXTree structural containment within pages, and E_τ captures temporal interaction transitions across successful tasks. Crucially, the graph functions as an environment-grounded *negative action proposal mechanism* $a_t^- \sim q_G(a \mid s_t, a_t^+)$, proposing contrastive candidates across spatial (structural), temporal, and spatiotemporal confusion axes. A strong teacher model then filters false negatives via multi-trial consistency and synthesizes authoritative judgment rationales, yielding the high-quality SURFPRM-DATA preference dataset. Our core contributions are: (1) **Problem Diagnosis**: We identify the severe scarcity of Grounded Minimal Contrastive Pairs (GMCPs) as a key deficiency in existing multi-policy sampling Web PRM data synthesis pipelines, which introduces shallow shortcuts that bias PRMs during training to rely on superficial cues rather than contextual decision semantics. (2) **Graph Proposal Mechanism**: We introduce the Interaction Element Graph proposal mechanism in SURFPRM with three-axis confusion rules, elevating the GMCP proportion from 24.19% in WebArbiter to 74.60% in SURFPRM-DATA while preserving legitimate non-DOM actions. (3) **Empirical Validation**: Across six open-source backbones (3B to 9B parameters), comparative Web PRMs trained on SURFPRM-DATA outperform baseline-trained models on average on WEBPRMBENCH and rival leading proprietary LLMs, delivering +14.21% (GPT-4o) and +12.83% (GPT-4o-mini) success gains in downstream WEBARENA-LITE search.

2 RELATED WORK

Web Agents and Interactive Decision-Making. LLM-powered web and computer-use agents have advanced across benchmarks such as WebShop (Yao et al., 2022), Mind2Web (Deng et al., 2023), WebArena (Zhou et al., 2024), WebVoyager (He et al., 2024), VisualWebArena (Koh et al., 2024), and OSWorld (Xie et al., 2024). To support learning in these environments, prior work has scaled demonstration synthesis (Ou et al., 2024; Xu et al., 2025; Murty et al., 2025; Pahuja et al., 2025; Sun et al., 2025; Wang et al., 2026) and developed synthetic environments and environment models (Gao et al., 2026; Fang et al., 2025; Xiao et al., 2026; Zhang et al., 2026b; Fan et al., 2026; Wu et al., 2026; Chae et al., 2026; Bai et al., 2026). However, web interfaces present dense, visually and semantically overlapping interactable elements, where empirical failure analyses identify low-level execution and grounding errors—rather than high-level planning—as the primary bottleneck (Aghzal et al., 2026). Because imitation learning on positive demonstrations only exposes agents to successful paths, it provides no discriminative signal against superficially plausible decoy

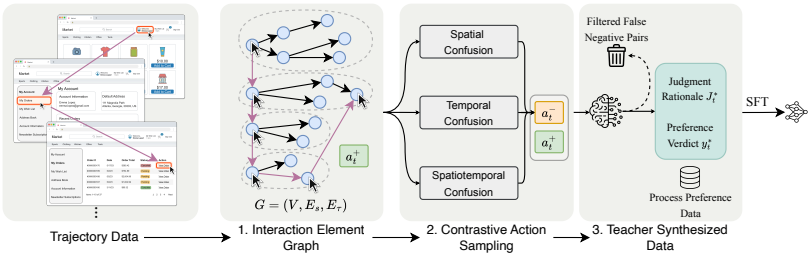

Figure 1: **Overview of the SURFPRM Framework.** Demonstration trajectories are aggregated into an Interaction Element Graph $G = (V, E_s, E_\tau)$ capturing spatial containment and temporal paths. The graph acts as a Negative Action Proposal Mechanism across spatial, temporal, and spatiotemporal axes. Teacher models filter false negatives and synthesize gold rationales and verdicts for comparative PRM training.

elements in the same observation, highlighting the need for fine-grained step-level verification and evaluative feedback to distinguish competing actions.

Web Agent Reward Models. Reward modeling for web agents initially focused on trajectory-level Outcome Reward Models (ORMs), such as the outcome evaluators in WebRL (Qi et al., 2025) and WebAgent-R1 (Wei et al., 2025), which assess overall task completion to guide reinforcement learning. However, trajectory-level feedback provides limited guidance for attributing outcomes to individual intermediate actions (Lù et al., 2025). Web Process Reward Models (PRMs) instead provide step-level guidance: WebShepherd (Chae et al., 2025) constructs a benchmark with 40K preference pairs using multi-policy sampling and heuristic filtering, while WebArbiter (Zhang et al., 2026a) generates preference verdicts with principle-guided judgment rationales. These methods use policy-generated candidates, which do not by themselves ensure grounded, action-type-controlled comparisons; including candidate rationales can additionally expose stylistic cues. Our analysis of WebArbiter data finds that Grounded Minimal Contrastive Pairs (GMCPs) account for only 24.19% of pairs. We focus on constructing process preference supervision: in SURFPRM, an Interaction Element Graph proposes contrastive negative actions for successful demonstration steps, followed by teacher filtering and synthesis of judgment rationales and preference verdicts for comparative Web PRM training.

Process Supervision and Step-Level Data Synthesis. Originating in mathematical reasoning, process supervision frameworks such as Math-Shepherd (Wang et al., 2024b) automated step-level annotation via Monte Carlo (MC) rollouts from intermediate states, while ProcessBench (Zheng et al., 2025) showed that PRM generalization depends crucially on data quality rather than raw volume. However, translating rollout-based process supervision to web environments is fundamentally intractable: live web pages cannot be arbitrarily cloned or reset, actions such as form submission are irreversible, and automated outcome verification is brittle (Aghzal et al., 2026; Lù et al., 2025). Although both domains address step-level credit assignment, recent agentic RL methods (e.g., GiGPO (Feng et al., 2025), GraphGPO (Cheng et al., 2026)) operate under an orthogonal paradigm focused on online policy optimization through interactive exploration. In contrast, SURFPRM constructs an comparative verifier and process reward model to serve as an evaluative guide for downstream inference-time search. To synthesize reliable process preference supervision without requiring active environment rollouts or resets, SURFPRM perturbs golden demonstration steps along spatial, temporal, and spatiotemporal axes on a persistent Interaction Element Graph, constructing environment-grounded, minimally contrastive negatives from existing trajectory graphs.

3 METHOD

3.1 PROBLEM FORMULATION AND COMPARATIVE PRM PIPELINE

State, Action, and Comparative PRM Formulation. We formulate web navigation as a sequential decision process over discrete steps $t \in \{1, \ldots, T\}$. At step t, the environmental state is $s_t = (I, O_t, H_t, u_t)$, where I is the user instruction, O_t is the linearized Accessibility Tree

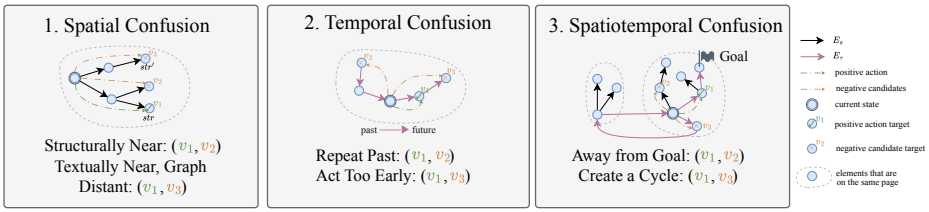

Figure 2: **Negative Action Synthesis across Three Confusion Axes.** Candidate negatives are proposed from the Interaction Element Graph along spatial, temporal, and spatiotemporal axes, targeting distinct confusion modes (formal definitions in Appendix A).

(AXTree) with discrete element identifiers (BID $\in \mathbb{N}$), $H_t = (a_1, \ldots, a_{t-1})$ is the action history, and u_t is the current URL. An action $a_t = (\mathrm{op}_t, \mathrm{target}, \mathrm{value})$ specifies the operation type $\mathrm{op}_t \in \{\texttt{click}, \texttt{fill}, \texttt{hover}, \texttt{select_option}, \ldots\}$, target element identifier, and optional parameters. A comparative Web PRM f_θ evaluates two candidate actions (a_t^1, a_t^2) conditioned on s_t: $f_\theta(s_t, a_t^1, a_t^2) \to (j_t, \hat{y}_t)$ with $\hat{y}_t \in \{1, 2\}$, where j_t is an explicit *judgment rationale* providing comparative causal reasoning, and $\hat{y}_t$ indicates the preferred candidate index ($\hat{y}_t = 1 \iff a_t^1 \succ a_t^2$) (Zhang et al., 2026a). Under this setting, PRM training data consists of preference tuples $d_t = (s_t, a_t^+, a_t^-, j_t^*, y_t^*)$, where a_t^+ is the positive step from an expert demonstration, a_t^- is a contrastive negative action, j_t^* is the gold judgment rationale, and y_t^* is the ground-truth verdict.

3.2 Website Interaction Element Graph Construction

To provide structured proposals for negative actions, SURFPRM aggregates demonstration trajectories into persistent, site-level **Interaction Element Graphs** $G = (V, E_s, E_\tau)$.

Persistent Element Identity Mapping. Runtime Backend Node IDs (BIDs) in AXTrees are ephemeral across reloads and sessions. To track elements across trajectories, we map each element to a persistent node $v = \phi(\text{site, URL, elem_ctx}) \in V_{\text{elem}}$ using BID-invariant context (e.g., accessible role and name, ancestor chain, sibling position, and subtree signature; details in Appendix B). For actions without DOM anchors (`scroll`, `go_back`, `new_tab`, `respond`), we instantiate page-scoped Virtual Nodes $v_{\text{virt}} \in V_{\text{virt}}$ keyed by the landing page and action type. Instantiating virtual nodes per page rather than globally preserves local topological fidelity, yielding the unified node set $V = V_{\text{elem}} \cup V_{\text{virt}}$.

Graph Topology. For each website, the graph $G = (V, E_s, E_\tau)$ captures two complementary topologies: (1) **Spatial Containment Edges** (E_s): On each page observation, nodes represent BID-bearing interactive elements in the AXTree. For each element v, we traverse upward along its ancestor chain, skipping inert structural containers (`listitem`, `generic`, etc.), and connect an edge $(u, v) \in E_s$ to its *nearest interactive ancestor* u, recording the hierarchy gap (depth_gap), with duplicates removed across observations. Bypassing non-operable containers avoids injecting hundreds of action-free, noisy edges per page. Instead, linking solely to the nearest interactive ancestor directly captures *nested interactive structures* (e.g., options within comboboxes)—inducing realistic scoping ambiguities (inner vs. outer targets) critical for contrastive negatives while eliminating transitive redundancy. (2) **Temporal Interaction Edges** (E_τ): Directed transitions $(v_t \xrightarrow{a_t} v_{t+1})$ observed between consecutive steps across successful trajectories. The graph reflects observed structural and temporal relationships rather than a complete world model, proposing plausible alternative candidates rather than proving unobserved paths impossible.

3.3 Graph-Based Grounded Minimal Negative Action Synthesis

Given a positive step (s_t, a_t^+) targeting element v_t^+, the SURFPRM proposal mechanism samples candidate actions $a_t^- \sim q_G(a \mid s_t, a_t^+)$ from G. Importantly, virtual nodes V_{virt} serve to model trajectory flow and represent non-DOM positive steps (e.g., `scroll` or terminal responses) paired with competing page-element negatives, but are never proposed as negative actions. To prevent element hallucinations, negative candidates are drawn strictly from $\mathcal{V}_{\text{grounded}}(s_t)$, the set of concrete DOM elements in G uniquely resolvable in the active observation O_t; proposed candidate nodes

that cannot be uniquely bound to an active BID in O_t are discarded. This mechanism elevates the proportion of Grounded Minimal Contrastive Pairs (GMCPs) to 74.60%, balancing grounded minimal contrast with the need to preserve legitimate virtual positive steps and respect element affordances, rather than enforcing artificial 100% GMCP pairing.

We synthesize contrastive negatives across three confusion axes comprising six rules $\mathcal{R} = \{r_1, \ldots, r_6\}$ (Figure 2; formal definitions are detailed in Appendix A):

- **Spatial Confusion**: Perceptual distraction within the active page: (1) *Structurally Near*, selecting a nearby element within a few AXTree containment hops of the positive target (e.g., a nearby date button *"Feb 25"* instead of *"Jan 28"* in a JetBlue booking calendar); and (2) *Textually Near, Graph-Distant*, selecting an element with high text similarity (> 0.75, character-level sequence-matching ratio) but > 4 hops away or unreachable in the combined element graph (containment plus interaction edges) (e.g., selecting the 1984 film link instead of the 2021 release for *Dune* on Rotten Tomatoes). Note that graph distance is essential: elements reachable within ≤ 4 hops in the combined graph across trajectories are excluded to avoid penalizing viable multi-step paths.
- **Temporal Confusion**: Sequential misordering along the interaction timeline: (3) *Repeat Past*, repeating an operation from k steps prior ($1 \leq k \leq N$) still visible in O_t (e.g., repeating a `hover` action on an artist profile on SoundCloud); and (4) *Act Too Early*, executing an action scheduled k steps ahead ($1 \leq k \leq N$) that is visible prematurely (e.g., clicking search submit on Amazon before entering keywords).
- **Spatiotemporal Confusion**: Workflow divergence from global task progression: (5) *Away from Goal*, increasing shortest graph distance to the task's final interaction element in the combined element graph (e.g., clicking an unrelated promotional item rather than submitting search on Micro Center); and (6) *Create a Cycle*, transitioning back to a previously visited state and creating a loop in E_τ (e.g., clicking the home logo on IMDb while viewing search auto-suggestions).

The complete proposal distribution combines all active rules: $\mathcal{N}(s_t, a_t^+) = \bigcup_{r \in \mathcal{R}} \mathcal{N}_r(s_t, a_t^+; G)$. To prevent position bias, each pair (a_t^+, a_t^-) is randomly shuffled into candidate positions (a_t^1, a_t^2), with the ground-truth preference verdict $y_t^* \in \{1, 2\}$ dynamically set to track the index of the positive action, represented without candidate rationales (Zheng et al., 2023; Wang et al., 2024a) (Appendix C).

3.4 Teacher Verification and Gold Supervision Synthesis

To ensure label reliability and filter potential false negatives (where the negative action represents an equally valid alternative path), we evaluate each candidate pair using a high-capacity Teacher Model over $K = 5$ independent trials: $(j_t^{(k)}, \hat{y}_t^{(k)}) \sim \text{Teacher}(s_t, a_t^1, a_t^2)$. A candidate pair is flagged as a false negative and discarded if and only if the teacher contradicts the trajectory label in *every single trial* ($\forall k, \hat{y}_t^{(k)} \neq y_t^*$). Out of 10,000 sampled candidate pairs, only 179 were filtered out under this criterion, yielding a high verification pass rate of 98.21%.

For each retained pair, we extract a consistent judgment rationale corresponding to the correct label as the **Gold Judgment Rationale** j_t^*. The rationale articulates user intent, active page affordances, and the causal justification for why $a_t^{y_t^*}$ advances the goal while the other action represents a failure mode. Using the curated SURFPRM-DATA dataset $\mathcal{D} = \{(s_t, a_t^1, a_t^2, j_t^*, y_t^*)\}_{i=1}^N$, we train the comparative Web PRM f_θ via auto-regressive cross-entropy loss:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\sum_{i=1}^{N} \sum_{k=1}^{|Y_t|} \log P_\theta(y_{t,k} \mid s_t, a_t^1, a_t^2, y_{t,<k}) \tag{1}$$

where target sequence $Y_t = (j_t^* \circ y_t^*)$.

4 Data Collection and Dataset Statistics

4.1 Source Trajectories and Graph Construction

Our dataset, SURFPRM-DATA, is constructed from successful web interaction trajectories drawn from the WPRM demonstration pool (Chae et al., 2025). The source pool spans 50 diverse websites

with 1,409 normalized URL paths across e-commerce, content management systems, travel booking, technical forums, and administrative dashboards. We process 854 successful trajectories comprising 9,497 page observations to construct 50 site-level subgraphs. The aggregated graphs encompass 116,182 element nodes (V), 1,532 deduplicated Spatial Containment Edges (E_s), and 6,739 deduplicated Temporal Interaction Edges (E_τ). On average, each site subgraph contains 2,324 element nodes, 134.8 Temporal Interaction Edges, 30.6 Spatial Containment Edges, and 17.1 trajectories. Notably, our source demonstrations match those of WebArbiter (Zhang et al., 2026a), ensuring performance differences stem solely from the negative action synthesis process.

4.2 Dataset Curation and Final Statistics

Graph traversal across the 50 subgraphs produced 31,648 mined candidates. Applying environmental grounding (requiring elements to uniquely resolve in active observation O_t) yielded a pool of 31,400 candidates. We sampled 10,000 candidate pairs (seed 42) for teacher verification, matching the data scale of the WebArbiter SFT baseline to enable a controlled comparison at an aligned budget. Following multi-trial teacher consistency filtering ($K = 5$), **9,821 pairs passed verification** (179 pairs filtered, yielding an agreement pass rate of 98.21%; detailed teacher verification costs are reported in Appendix C), forming the final SURFPRM-DATA dataset. Table 1 details the distribution of proposal rules. The final dataset covers 1,084 normalized URL paths (after host lowercasing, `www`-prefix stripping, and trailing-slash removal).

Table 1: Distribution of Proposal Rules in SURFPRM-DATA (10k Pairs).

Proposal Rule	**Axis**	**Count**
Textually Near, Graph-Distant	Spatial	5,022
Away from Goal	Spatiotemp.	4,020
Act Too Early	Temporal	304
Structurally Near	Spatial	223
Repeat Past	Temporal	166
Create a Cycle	Spatiotemp.	86
Total		9,821

SURFPRM increases the proportion of Grounded Minimal Contrastive Pairs (GMCPs) from **24.19%** (WebArbiter baseline) to **74.60%** (7,326 out of 9,821 pairs), more than a threefold improvement. Crucially, among the remaining 25.40% non-GMCP pairs, 0% fail due to target elements being absent from the current page; every proposed negative action is strictly grounded in an active element within O_t. Instead, the non-GMCP residual stems from two deliberate design choices: (1) virtual positive pairs (16.6%), where demonstration steps whose positive action is a global viewport or navigation operation (e.g., `scroll` or terminal responses) are paired with competing page-element negatives, diverging in target modality; and (2) affordance-driven action-type divergence (9.1%), where negative candidates adopt the operation type historically demonstrated on that element to respect executable affordances (e.g., clicking a link vs. filling an input) rather than coercing an identical operation type onto incompatible elements. A full structural decomposition is provided in Appendix B.

4.3 Quality Assurance and Iterative Refinement

We instituted automated and rule-based verification protocols to guarantee execution compliance: (1) **Contextual Value Completion**: Missing arguments for input-dependent actions (e.g., `fill`) are contextually completed by a model conditioned on AXTree context, and non-editable controls (e.g., custom comboboxes) are normalized to valid affordances like `click` (Appendix F); audits confirmed zero empty values. (2) **Human False-Negative Audit**: In a manual audit on 100 randomly sampled pairs from both WebArbiter and SURFPRM-DATA, 30 of 100 WebArbiter negatives were false negatives (valid alternative actions mislabeled as inferior), whereas SURFPRM-DATA yielded only 12—a 2.5× reduction in false-negative rate (30% → 12%), confirming substantially cleaner and more dependable process preference supervision.

5 Experiments

5.1 Experimental Setup

Benchmarks and Evaluation Protocol. We evaluate comparative Web PRMs across two tasks: (1) **WebPRMBench**: State-conditioned pairwise preference discrimination evaluated on Pairwise and Best-of-N (BoN) Accuracy across WebArena (Zhou et al., 2024), AssistantBench (Yoran

Table 2: **Step-Level Process Preference Discrimination on WebPRMBench.** Evaluated on Pairwise and BoN Accuracy (%). Average reports macro-average across the three environments. $*$ denotes baseline results directly reported from the WEBARBITER paper. Models fine-tuned on SURFPRM-DATA are shaded; **bold** marks best within each group in Average.

Model	WebArena		AssistantBench		WorkArena		Average	
	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN
Closed-Source LLM								
GPT-5*	84.83	71.64	81.67	63.33	81.14	64.62	82.55	66.53
Claude-4.6-Sonnet	84.58	71.14	80.00	60.00	83.02	66.51	82.53	65.88
Gemini-3.7-Flash	83.71	68.66	83.33	73.33	78.77	64.15	81.94	68.71
Jev-1.13 (choice mode)	84.83	70.65	79.17	60.00	78.42	58.49	80.81	63.05
Jev-1.13 (noul mode)	84.70	69.65	76.67	56.67	78.89	57.55	80.09	61.29
Open-Source LLM								
Llama-3-70B-Instruct*	77.36	50.75	85.83	70.00	79.08	40.09	80.76	53.61
DeepSeek-R1*	82.04	60.21	78.49	56.18	84.12	63.89	81.55	60.09
DeepSeek-V4-Flash-0731	83.21	71.64	80.00	63.33	79.36	58.49	80.86	64.49
Fine-Tuned LLM								
WebShepherd-3B*	68.16	41.29	66.67	46.67	50.00	21.23	61.61	36.40
WebShepherd-8B*	68.33	43.88	55.92	30.00	54.56	25.53	59.60	33.14
Qwen2.5-3B-Instruct	70.02	35.82	78.33	46.67	68.99	29.25	72.45	37.25
+ WebArbiter 10k SFT	81.84	55.22	69.17	40.00	71.58	35.38	74.20	43.53
+ Ours 10k SFT	81.59	55.22	85.00	56.67	75.35	41.98	**80.65**	**51.29**
Qwen2.5-7B-Instruct	73.88	44.78	81.67	50.00	72.76	37.26	76.10	44.01
+ WebArbiter 10k SFT	80.35	54.23	83.33	60.00	78.07	51.42	80.58	55.22
+ Ours 10k SFT	84.33	62.19	85.83	56.67	77.95	50.00	**82.70**	**56.29**
Qwen3-4B	73.38	47.26	78.33	43.33	69.93	36.79	73.88	42.46
+ WebArbiter 10k SFT	84.95	63.18	76.67	46.67	75.12	48.11	78.91	52.65
+ Ours 10k SFT	83.71	60.20	80.83	56.67	79.36	51.42	**81.30**	**56.10**
Qwen3-8B	76.24	50.75	80.83	56.67	73.00	38.21	76.69	48.54
+ WebArbiter 10k SFT	84.58	64.18	80.83	60.00	75.94	48.11	80.45	57.43
+ Ours 10k SFT	84.58	62.19	88.33	73.33	79.72	52.36	**84.21**	**62.63**
Qwen3.5-4B	81.72	62.69	75.83	50.00	80.07	53.77	79.21	55.49
+ WebArbiter 10k SFT	83.83	59.70	86.67	60.00	76.89	47.64	82.46	55.78
+ Ours 10k SFT	85.70	69.15	88.33	73.33	81.84	58.96	**85.29**	**67.15**
Qwen3.5-9B	82.46	69.65	80.83	66.67	79.48	55.66	80.92	63.99
+ WebArbiter 10k SFT	85.57	68.16	90.83	73.33	77.12	52.36	84.51	64.62
+ Ours 10k SFT	87.44	70.65	88.33	73.33	83.14	60.85	**86.30**	**68.28**

et al., 2024), and WORKARENA (Drouin et al., 2024). Because the demonstration trajectories used for SFT (in both SURFPRM and the WebArbiter baseline) are drawn from the WPRM demonstration pool (Chae et al., 2025) built on MIND2WEB environments (Deng et al., 2023), the MIND2WEB subset of WEBPRMBENCH overlaps with the SFT training distribution. We therefore exclude it, so that evaluation on the three unseen environments directly measures the cross-scenario transferability of the trained PRMs rather than in-distribution memorization. (2) **WEBARENA-LITE** (Liu et al., 2024): Reward-guided trajectory search across five web environments, evaluating whether comparative PRMs improve downstream agent performance. At each step, the policy generates five candidate actions, and the PRM selects one via a knockout tournament (Guo et al., 2025). We evaluate GPT-4o and GPT-4o-mini as policy model, reporting task success rates and absolute gains (Δ) over the corresponding policies without PRM-guided search.

Evaluated Models and Baselines. We benchmark: (1) **Closed-Source LLMs**: GPT-5 (Singh et al., 2026), Claude-4.6-Sonnet (Anthropic, 2026), Gemini-3.7-Flash (Google DeepMind, 2026), and jev-1.13 (TypeSafe AI, 2026) (Appendix D.4); (2) **Open-Source LLMs**: Llama-3-70B-Instruct (Grattafiori et al., 2024), DeepSeek-R1 (DeepSeek-AI et al., 2025), and DeepSeek-V4-Flash-0731 (DeepSeek-AI et al., 2026); and (3) **Fine-Tuned Comparative Web PRMs**: Baselines WEBSHEPHERD-3B/8B (Chae et al., 2025), alongside systematic comparisons across six open-

Table 3: **Downstream Trajectory Search on WebArena-Lite.** Success rates (%) across 5 environments with Best-of-N search guided by Web PRMs. Avg. is macro-average; Δ is absolute gain over unguided execution. $*$ denotes baseline results directly reported from the WEBARBITER paper. Rows trained on SURFPRM-DATA are shaded; **bold** marks best within each policy group.

Policy Model	WebPRM Guidance	Shopping	CMS	Reddit	GitLab	MAP	Avg.	Δ
GPT-4o-mini	w/o Trajectory Search*	21.74	22.86	19.05	34.38	19.35	23.48	–
	GPT-4o-mini*	24.44	22.86	26.32	33.33	15.38	24.47	+0.99
	WebShepherd-8B*	26.09	45.71	23.81	40.62	35.48	34.34	+10.87
	Qwen3.5-9B + WebArbiter 10k	34.78	38.89	33.33	38.24	19.35	32.92	+9.44
	Qwen3.5-9B + Ours 10k	36.96	30.56	50.00	38.24	25.81	**36.31**	**+12.83**
GPT-4o	w/o Trajectory Search*	23.91	31.43	28.57	56.25	19.35	31.90	–
	GPT-4o-mini*	26.67	37.14	42.11	40.00	19.23	33.03	+1.13
	WebShepherd-8B*	30.43	42.86	47.62	46.88	35.48	40.65	+8.75
	Qwen3.5-9B + WebArbiter 10k	34.78	30.56	45.83	44.12	35.48	38.15	+6.25
	Qwen3.5-9B + Ours 10k	47.83	50.00	62.50	41.18	29.03	**46.11**	**+14.21**

source Qwen backbones (3B–9B) (Qwen Team et al., 2024; Yang et al., 2025; Qwen Team, 2026) fine-tuned on WebArbiter 10k SFT (Zhang et al., 2026a) vs. SURFPRM-DATA under an aligned $\sim$10k budget and identical protocols.

5.2 MAIN EXPERIMENTAL RESULTS

Step-Level Process Preference Discrimination on WEBPRMBENCH. Table 2 presents preference discrimination results on WEBPRMBENCH: (1) **Consistent Macro-Average Superiority Across Scales**: Models trained on SURFPRM-DATA consistently outperform the WebArbiter baseline across all six backbones (3B to 9B) in both macro-average Pairwise and BoN metrics (e.g., Qwen2.5-3B achieves +6.45% Pairwise and +7.76% BoN gains; Qwen3.5-9B reaches 86.30% / 68.28%), yielding the highest macro-average discrimination at every scale. (2) **Data Efficiency & Generalization**: Qwen2.5-3B trained on SURFPRM-DATA (80.65% Pairwise) surpasses larger models trained on WebArbiter (Qwen2.5-7B at 80.58%, Qwen3-8B at 80.45%), peaking on enterprise WORKARENA (Qwen3.5-4B: 81.84% / 58.96% vs. 76.89% / 47.64%) and ASSISTANTBENCH (Qwen3-8B: 88.33% / 73.33% vs. 80.83% / 60.00%). (3) **Competitive with Frontier LLMs**: Fine-tuned open-source PRMs match or exceed proprietary LLMs like Claude-4.6-Sonnet (82.53% / 65.88%) and GPT-5 (82.55% / 66.53%).

Downstream Reward-Guided Trajectory Search on WEBARENA-LITE. Table 3 evaluates comparative Web PRMs as search guides over external policy generators on WEBARENA-LITE: (1) **Macro-Average Guidance Gains**: Across environments, our PRM (SURFPRM, Qwen3.5-9B + Ours 10k) outperforms unguided execution, WEBSHEPHERD, and WEBARBITER. Guiding GPT-4o, it achieves 46.11% macro-average success (+14.21% over unguided, +7.96% over WebArbiter); guiding lightweight GPT-4o-mini, it reaches 36.31% (+12.83%), surpassing unguided GPT-4o (31.90%). (2) **Resolving High-Branching Bottlenecks**: SURFPRM excels in complex environments: under GPT-4o, it doubles success on Shopping (23.91% $\rightarrow$ 47.83%) and leads on Reddit (28.57% $\rightarrow$ 62.50%) and CMS (31.43% $\rightarrow$ 50.00%); under GPT-4o-mini, it lifts Reddit from 19.05% to 50.00%. On GitLab, guidance declines (56.25% $\rightarrow$ 41.18% under GPT-4o), likely because code repository UIs diverge from consumer web domains in training demonstrations. (3) **Effective Plug-and-Play Verifier**: Unlike policy self-evaluation (+0.99%), SURFPRM exhibits robust out-of-policy discrimination across distinct generators, confirming the practical utility of comparative process reward modeling.

5.3 IN-DEPTH ANALYSIS AND ABLATION STUDIES

Impact of the Grounded Minimal Contrastive Pair (GMCP) Ratio. We evaluate Qwen2.5-3B across controlled GMCP ratios under an aligned $\sim$10k budget (Figure 3; Table 6 in Appendix E): (1) **Overall Upward Scaling Trend**: Macro-average Pairwise Accuracy climbs from 74.20% (24.19% GMCP) to 80.65% (74.60%), and BoN accuracy improves from 43.53% to 51.29%, confirming that

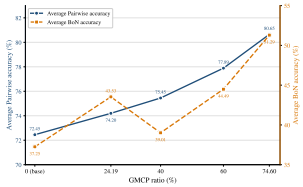

Figure 3: **Effect of GMCP Ratio.** Average Pairwise/BoN accuracy of Qwen2.5-3B-Instruct under controlled GMCP ratios (24.19%–74.60%) vs. un-finetuned base (0%); overall performance exhibits an upward scaling trend as the GMCP ratio increases (Table 6 in Appendix E).

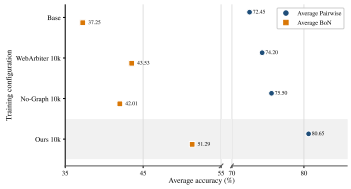

Figure 4: **Ablation on Graph Necessity.** Average Pairwise/BoN accuracy of PRMs on WebArbiter, No-Graph, and SURFPRM (SURFPRM-DATA); graph topology provides critical structural signals (Table 7 in Appendix E).

higher proportions of grounded minimal contrastive supervision yield superior discrimination. (2) **Contextual Decisions**: We additionally partition the pooled test pairs of all three benchmark environments by whether both candidates operate on genuine page elements with identical action types: on the *Full GMCP subset*, BoN accuracy scales steadily: 45.89% (base) $\rightarrow$ 66.67% (WebArbiter) $\rightarrow$ 71.50% (Ours, +4.83% over WebArbiter; Pairwise reaching 85.71%). Conversely, on the *Full Non-GMCP subset*, accuracy saturates early across all variants ($\sim$51% BoN), confirming that gains stem from contextual decision-making rather than superficial heuristics.

Ablation on Graph Necessity. We compare against a strictly **No-Graph** baseline (Figure 4; Table 7, Appendix E) mining negatives purely from local AXTree snapshots without cross-trajectory graphs. Both settings share identical demonstration states, BID remapping, teacher scoring ($K = 5$), matched GMCP ratios (75.56% vs 74.60%), and aligned scales (9,643 vs 9,821 pairs). (1) **Clear Graph Advantage**: Ours 10k SFT achieves 80.65% Pairwise and 51.29% BoN, substantially outperforming No-Graph 10k SFT (75.50% / 42.01%) and WebArbiter (74.20% / 43.53%), with large gains on multi-step benchmarks like AssistantBench (85.00% / 56.67% vs. 70.00% / 30.00%) and WorkArena (75.35% / 41.98% vs. 72.05% / 36.32%). (2) **Topological Signals**: This confirms that cross-trajectory topological signals provide critical structural discrimination that local heuristics cannot replicate.

Ablation on Graph Confusion Axes. Evaluating PRMs trained on individual axes (Figure 5; Table 5 in Appendix E) reveals: (1) **No Single Axis Is Sufficient**: Each axis exhibits clear weaknesses—`Only-Spatial` (76.85% / 49.43%) and `Only-Spatiotemporal` (78.00% / 43.36%) lag behind the full mixture, while `Only-Temporal` (74.63% / 35.01%) underperforms substantially on AssistantBench and WorkArena. (2) **All Three Axes Are Necessary**: The full multi-axis mixture achieves highest macro-average (80.65% / 51.29%), outperforming every single-axis variant on average Pairwise and BoN accuracy and confirming that the three axes are complementary and jointly necessary across disparate web environments (Figure 5; `All Axes` normalized to 100% to illustrate the relative performance).

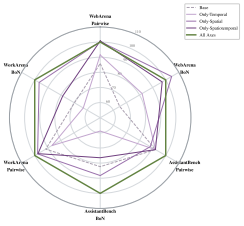

Figure 5: Axis ablation.

6 CONCLUSION

We introduce SURFPRM, a graph-guided framework for synthesizing environment-grounded, minimally contrastive process preference supervision for Web PRMs. By proposing negatives across spatial, temporal, and spatiotemporal axes on an Interaction Element Graph, SURFPRM elevates the GMCP proportion from 24.19% to 74.60% in SURFPRM-DATA, eliminating shallow shortcuts like element hallucinations. Across six backbones (3B–9B), PRMs trained on SURFPRM-DATA outperform baselines on WEBPRMBENCH and provide substantial search gains (+14.21% on GPT-4o, +12.83% on GPT-4o-mini) on WEBARENA-LITE.

Reproducibility Statement

To facilitate the reproduction of our findings, we provide complete algorithmic, implementation, and experimental specifications across the paper. The Interaction Element Graph formulation, negative proposal rules, and action mappings are detailed in Section 3.2 and Appendices A and B. The data synthesis pipeline, including Action-Only formatting, teacher verification ($K = 5$), quality assurance, and full prompt templates, is described in Sections 3.4 and 4.3 and Appendices C, D, and F. Evaluation setups and extended per-environment ablations are provided in Section 5.1 and Appendix E. Our synthesized datasets and trained model checkpoints are anonymously accessible at https://anonymous.4open.science/r/surfPRM-6188/, and full source code will be publicly released upon acceptance.

AI Use Statement

In this work, we used generative AI tools to generate synthetic datasets, implement methods, assist with translation, clean and reformat dataset, support qualitative and thematic data analysis, and interpret results. We have not used generative AI tools to propose or refine hypotheses, design or provide feedback on research methodology or experiments, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, or assist in the writing of proofs (not applicable to this work). Additionally, we used generative AI tools to create or edit software code and identify relevant literature. We have reviewed all AI-assisted work. LLM-generated code was verified and tested for correctness by authors. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

References

Mohamed Aghzal, Gregory J. Stein, and Ziyu Yao. Why do llm-based web agents fail? a hierarchical planning perspective. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026.

Anthropic. Claude Sonnet 4.6 system card. https://www.anthropic.com/claude-sonnet-4-6-system-card, 2026.

Hao Bai, Alexey Taymanov, Tong Zhang, Aviral Kumar, and Spencer Whitehead. Webgym: Scaling training environments for long-horizon visual web agents with realistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

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, Beongwoo Kwak, Dongjin Kang, and Jinyoung Yeo. Web-shepherd: Advancing prms for reinforcing web agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Hyungjoo Chae, Jungsoo Park, and Alan Ritter. Safe and scalable web agent learning via recreated websites. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026.

Xin Cheng, Shuo He, Lang Feng, Haiyang Xu, Ming Yan, Lei Feng, and Bo An. Beyond trajectory-level attribution: Graph-based credit assignment for agentic reinforcement learning. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

DeepSeek-AI, Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026.

Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Alexandre Drouin, Maxime Gasse, Massimo Caccia, Issam H. Laradji, Manuel Del Verme, Tom Marty, David Vázquez, Nicolas Chapados, and Alexandre Lacoste. Workarena: How capable are web agents at solving common knowledge work tasks? In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, 2024.

Sicheng Fan, Qingyun Shi, Shengze Xu, Shengbo Cai, Tieyong Zeng, Li Ling, Yanyi Shang, and Dehan Kong. Webfactory: Automated compression of foundational language intelligence into grounded web agents. In *Proceedings of the 14th International Conference on Learning Representations (ICLR)*, 2026.

Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. Webevolver: Enhancing web agent self-improvement with coevolving world model. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Yifei Gao, Junhong Ye, Yifan Yang, Jiaqi Wang, Yi Zhang, Ruichen Zhang, and Jitao Sang. Websynthesis: World model-guided monte carlo tree search for efficient webagent trajectory synthesis. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026.

Google DeepMind. Gemini 3.7 Flash model card. https://deepmind.google/models/model-cards/gemini-3-7-flash/, 2026.

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

Jiaxin Guo, Zewen Chi, Li Dong, Qingxiu Dong, Xun Wu, Shaohan Huang, and Furu Wei. Reward reasoning model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Hongliang He, Wenlin Yao, Kaixin Ma, Wenhao Yu, Yong Dai, Hongming Zhang, Zhenzhong Lan, and Dong Yu. Webvoyager: Building an end-to-end web agent with large multimodal models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024.

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

Jing Yu Koh, Stephen Marcus McAleer, Daniel Fried, and Ruslan Salakhutdinov. Tree search for language model agents. *Transactions on Machine Learning Research (TMLR)*, 2025.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, 2024.

Xiao Liu, Tianjie Zhang, Yu Gu, Iat Long Iong, Yifan Xu, Xixuan Song, Shudan Zhang, 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. *arXiv preprint arXiv:2408.06327*, 2024.

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. In *Proceedings of the 2nd Conference on Language Modeling (COLM)*, 2025.

Shikhar Murty, Hao Zhu, Dzmitry Bahdanau, and Christopher D. Manning. Nnetnav: Unsupervised learning of browser agents through environment interaction in the wild. In *Proceedings of the ICLR 2025 Workshop on Scaling Self-Improving Foundation Models without Human Supervision*, 2025.

Tianyue Ou, Frank F. Xu, Aman Madaan, Jiarui Liu, Robert Lo, Abishek Sridhar, Sudipta Sengupta, Dan Roth, Graham Neubig, and Shuyan Zhou. Synatra: Turning indirect knowledge into direct demonstrations for digital agents at scale. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pp. 27730–27744, 2022.

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 *Findings of the Association for Computational Linguistics (ACL)*, 2025.

Zehan Qi, Xiao Liu, Iat Long Iong, Hanyu Lai, Xueqiao Sun, Jiadai Sun, Xinyue Yang, Yu Yang, Shuntian Yao, Wei Xu, Jie Tang, and Yuxiao Dong. Webrl: Training llm web agents via self-evolving online curriculum reinforcement learning. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, 2025.

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

Qwen Team, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, et al. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 53728–53741, 2023.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, A. J. Ostrow, Akhila Ananthram, et al. OpenAI GPT-5 system card. *arXiv preprint arXiv:2601.03267*, 2026.

Qiushi Sun, Kanzhi Cheng, Zichen Ding, Chuanyang Jin, Yian Wang, Fangzhi Xu, Zhenyu Wu, Chengyou Jia, Liheng Chen, Zhoumianze Liu, Ben Kao, Guohao Li, Junxian He, Yu Qiao, and Zhiyong Wu. Os-genesis: Automating gui agent trajectory construction via reverse task synthesis. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

TypeSafe AI. Jev 1.13: Typesafe's first system one model. https://docs.typesafe.ai/models, 2026.

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 feedbacks. *arXiv preprint arXiv:2211.14275*, 2022.

Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Lingpeng Kong, Qi Liu, Tianyu Liu, and Zhifang Sui. Large language models are not fair evaluators. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9440–9450, 2024a.

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2024b.

Zhaoyang Wang, Yiming Liang, Xuchao Zhang, Qianhui Wu, Siwei Han, Anson Bastos, Rujia Wang, Chetan Bansal, Baolin Peng, Jianfeng Gao, Saravan Rajmohan, and Huaxiu Yao. Synthagent: Adapting web agents with synthetic supervision. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026.

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. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2025.

Yifan Wu, Yiran Peng, Yiyu Chen, Jianhao Ruan, Zijie Zhuang, Cheng Yang, Jiayi Zhang, Man Chen, Yenchi Tseng, Zhaoyang Yu, Liang Chen, Yuyao Zhai, Bang Liu, Chenglin Wu, and Yuyu Luo. Autowebworld: Synthesizing infinite verifiable web environments via finite state machines. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026.

Zikai Xiao, Jianhong Tu, Chuhang Zou, Yuxin Zuo, Zhi Li, Peng Wang, Bowen Yu, Fei Huang, Junyang Lin, and Zuozhu Liu. Webworld: A large-scale world model for web agent training. In *Proceedings of the 43rd International Conference on Machine Learning (ICML)*, 2026.

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. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

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. In *Proceedings of the 13th International Conference on Learning Representations (ICLR)*, 2025.

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

Yan Yang, Dongxu Li, Yutong Dai, Yuhao Yang, Ziyang Luo, Zirui Zhao, Zhiyuan Hu, Junzhe Huang, Amrita Saha, Zeyuan Chen, Ran Xu, Liyuan Pan, Caiming Xiong, and Junnan Li. Gta1: Gui test-time scaling agent. In *Proceedings of the 14th International Conference on Learning Representations (ICLR)*, 2026.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Ori Yoran, Samuel Joseph Amouyal, Chaitanya Malaviya, Ben Bogin, Ofir Press, and Jonathan Berant. Assistantbench: Can web agents solve realistic and time-consuming tasks? In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Yao Zhang, Shijie Tang, Zeyu Li, Zhen Han, and Volker Tresp. Webarbiter: A generative reasoning process reward model for web agents. In *Proceedings of the 14th International Conference on Learning Representations (ICLR)*, 2026a.

Ziyun Zhang, Zezhou Wang, Xiaoyi Zhang, Zongyu Guo, Jiahao Li, Bin Li, and Yan Lu. Infiniteweb: Scalable web environment synthesis for gui agent training. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2026b.

Chujie Zheng, Zhenru Zhang, Beichen Zhang, Runji Lin, Keming Lu, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Processbench: Identifying process errors in mathematical reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (ACL)*, 2025.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pp. 46595–46623, 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. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*, 2024.

A Formal Definitions of Graph-Based Proposal Rules

Let $s_t = (I, O_t, H_t, u_t)$ denote the environmental state, $a_t^+ = (\text{op}^+, v_t^+, \text{val}^+)$ the positive step targeting element $v_t^+ \in V$, and $\mathcal{V}_{\text{grounded}}(s_t)$ the set of elements in G uniquely resolvable in O_t. The distance $d_\tau(u, v)$ denotes the shortest path distance from u to v along temporal edges E_τ. The distance $d_s(u, v)$ denotes the shortest path distance from u to v along spatial edges E_s. Let $d_{\text{comb}}(u, v)$ denote the shortest graph distance between u and v on the combined element graph formed by spatial edges E_s and temporal edges E_τ. Let $\text{sim}_{\text{text}}(u, v) \in [0, 1]$ denote the character-level sequence-matching ratio between the normalized textual descriptors of u and v.

For each proposed candidate node $v^- \in \mathcal{V}_{\text{grounded}}(s_t)$, the negative action is represented as $a^- = (\text{op}^-, v^-, \text{val}^-)$. Rather than copying the positive step's operation or value, op^- and val^- are instantiated by preferring the most frequently demonstrated complete action for that node across trajectories (falling back to the element's observed executable action label in the current observation, e.g., `click` for clickable roles or `fill` with placeholder values subsequently completed by an LLM, when demonstration history is absent). The six proposal rules $\mathcal{R} = \{r_1, \ldots, r_6\}$ are formalized as follows:

1. Structurally Near.

$$\mathcal{N}_{\text{struct_near}}(s_t, a_t^+) = \{(\text{op}^-, v^-, \text{val}^-) \mid v^- \in \mathcal{V}_{\text{grounded}}(s_t) \setminus \{v_t^+\},\ 1 \le d_s(v_t^+, v^-) \le 5\} \tag{2}$$

where $d_s(v_t^+, v^-)$ is the path distance along spatial containment edges E_s.

2. Textually Near, Graph-Distant.

$$\begin{aligned}\mathcal{N}_{\text{text_near_graph_dist}}(s_t, a_t^+) = \{(\text{op}^-, v^-, \text{val}^-) \mid & v^- \in \mathcal{V}_{\text{grounded}}(s_t) \setminus \{v_t^+\}, \\ & \text{sim}_{\text{text}}(v_t^+, v^-) \ge 0.75, \\ & d_{\text{comb}}(v_t^+, v^-) > 4\}\end{aligned} \tag{3}$$

Elements reachable within ≤ 4 hops on the combined element graph (containment plus interaction edges) across trajectories are excluded, as they often constitute valid multi-step navigation paths observed across trajectories.

3. Repeat Past.

$$\begin{aligned}\mathcal{N}_{\text{repeat_past}}(s_t, a_t^+) = \{(\text{op}^-, v_{t-k}^+, \text{val}^-) \mid & 1 \le k \le \min(N, t-1), \\ & v_{t-k}^+ \in \mathcal{V}_{\text{grounded}}(s_t) \setminus \{v_t^+\},\ \text{url}(v_{t-k}^+) = \text{url}(s_t)\}\end{aligned} \tag{4}$$

where k indexes prior successful targeted interactions within the current trajectory (rather than raw environment steps) sharing the current page URL ($\text{url}(v_{t-k}^+) = \text{url}(s_t)$). The negative action is executed using the demonstrated action identity of node v_{t-k}^+. In our implementation, we set the lookback window to $N = 5$.

4. Act Too Early.

$$\begin{aligned}\mathcal{N}_{\text{act_early}}(s_t, a_t^+) = \{(\text{op}^-, v_{t+k}^+, \text{val}^-) \mid & 1 \le k \le \min(N, T-t), \\ & v_{t+k}^+ \in \mathcal{V}_{\text{grounded}}(s_t) \setminus \{v_t^+\},\ \text{url}(v_{t+k}^+) = \text{url}(s_t)\}\end{aligned} \tag{5}$$

where T denotes the total number of demonstrated steps in the trajectory, and k indexes subsequent successful targeted interactions scheduled ahead in the demonstration trajectory that share the current page URL ($\text{url}(v_{t+k}^+) = \text{url}(s_t)$). The negative action is executed using the demonstrated action identity of node v_{t+k}^+. We set the lookahead horizon to $N = 5$.

5. Away from Goal. Let v_{goal} denote the final interaction element executed in the current task trajectory. When $d_{\text{comb}}(v_t^+, v_{\text{goal}}) < \infty$, the rule proposes:

$$\begin{aligned}\mathcal{N}_{\text{away_goal}}(s_t, a_t^+) = \{(\text{op}^-, v^-, \text{val}^-) \mid & v^- \in \mathcal{V}_{\text{grounded}}(s_t) \setminus \{v_t^+\}, \\ & d_{\text{comb}}(v^-, v_{\text{goal}}) > d_{\text{comb}}(v_t^+, v_{\text{goal}})\}\end{aligned} \tag{6}$$

If $d_{\text{comb}}(v_t^+, v_{\text{goal}}) = \infty$ (i.e., the current positive target has no path to the goal element on the combined element graph), this rule is skipped ($\mathcal{N}_{\text{away_goal}}(s_t, a_t^+) = \emptyset$).

6. Create a Cycle.

$$\mathcal{N}_{\text{cycle}}(s_t, a_t^+) = \{(\text{op}^-, v^-, \text{val}^-) \mid v^- \in \mathcal{V}_{\text{grounded}}(s_t) \setminus \{v_t^+\}, \exists k \in \{1, \ldots, \min(N, t-1)\},\ d_\tau(v^-, v_{t-k}^+) < \infty\} \tag{7}$$

with $N = 5$. Here, candidate node v^- need not have been visited previously itself; rather, transitioning to v^- can reach one of the last $N = 5$ demonstrated historical interaction targets v_{t-k}^+ along temporal interaction edges E_τ ($d_\tau(v^-, v_{t-k}^+) < \infty$), inducing an interaction cycle of length $d_\tau(v^-, v_{t-k}^+) + 1$ back to recent history.

B Persistent Node Identity, Action Mapping, and Virtual Nodes

Persistent Element Identity Function (ϕ). Runtime Backend Node IDs (BIDs) in AXTrees are fundamentally ephemeral: they fluctuate across page reloads, dynamic AJAX updates, client-side rerenderings, and separate browser sessions. To track and align elements across distinct trajectories, we construct a persistent element identity mapping function $v = \phi(\text{site, URL, elem_ctx}) \in V_{\text{elem}}$ grounded on BID-invariant structural context:

- **Site and URL Scope**: site identifies the target website domain, and URL records the verbatim page URL observed in the demonstration. Scoping by normalized URL paths ensures that semantically distinct pages within the same site (e.g., checkout buttons on separate cart vs. order confirmation endpoints) instantiate distinct nodes rather than collapsing into false topological aliases.
- **Canonical Element Context (elem_ctx)**: The element context is a deterministic serialization of the element's ID-free structural position within the AXTree snapshot, defined by a four-tuple:
  - (i) *Accessible Descriptor*: The element's accessibility role and accessible name (e.g., `role="combobox", name="From"`), capturing the semantic function intended by web accessibility standards.
  - (ii) *Ancestor Descriptor Chain*: The ordered sequence of accessibility roles and names along the path from the element upward to the document root (e.g., `root` → `main` → `region "Flight search"` → `group`), omitting non-operable, generic wrapper containers.
  - (iii) *Sibling Position*: The ordinal index of the element among sibling nodes sharing the identical accessibility role, disambiguating repetitive UI arrays (e.g., consecutive date cells within a monthly calendar grid).
  - (iv) *Subtree Signature*: A compact canonical signature reflecting the immediate subtree structure (child roles and text labels), distinguishing structurally distinct containers even when sharing ancestor paths.

This mapping ensures that when an element reappears in subsequent sessions or independent trajectories under a mutated runtime BID, it resolves deterministically to the same persistent node $v \in V_{\text{elem}}$.

Page-Scoped Virtual Node Formulation. For actions that do not operate on physical DOM elements—specifically global viewport manipulations (`scroll`), browser-level navigation (`go_back`, `goto`), tab lifecycle actions (`new_tab`, `tab_close`), and terminal task outputs (`respond`)—no concrete DOM BID exists. To represent such actions within the Interaction Element Graph $G = (V, E_s, E_\tau)$, we define typed Virtual Nodes $v_{\text{virt}} \in V_{\text{virt}}$ under the same key function $\phi(\text{site}, \text{URL}, \text{ctx})$:

- **Landing Page Scoping**: The URL parameter in ϕ is set to the *landing URL*—defined as the page URL observed immediately after executing the action (or the current page for terminal response actions). The context ctx pairs the action type with the landing page's accessibility root descriptor.
- **Preserving Local Topological Fidelity**: Instantiating virtual nodes per landing page rather than as site-wide singletons is critical: a `scroll` on the search results page and a `scroll` on the product details page represent fundamentally distinct state transitions with different downstream reachable elements. Co-locating virtual nodes with their respective landing pages ensures that directed temporal interaction edges ($u \xrightarrow{a_t} v_{\text{virt}}$) and ($v_{\text{virt}} \xrightarrow{a_{t+1}} w$) preserve the precise local navigational flow of the website, yielding the unified node set $V = V_{\text{elem}} \cup V_{\text{virt}}$.

Action Modality to Graph Mapping. Browser interactions in web agents encompass both element-grounded interactions and global page-level commands. We define standard mappings between raw environment actions and the Interaction Element Graph $G = (V, E_s, E_\tau)$:

- **Element-Grounded Actions**: Actions targeting specific DOM elements—including `click(bid)`, `hover(bid)`, `fill(bid, text)`, `select_option(bid, opt)`, `press(bid, key)`, `focus(bid)`, `clear(bid)`, `dblclick(bid)`, and `drag_and_drop(...)`—are mapped to concrete persistent element nodes $v \in V_{\text{elem}}$ via our persistent identity function ϕ.
- **Global Viewport and Navigation Actions**: Actions that operate without an element anchor—`scroll(x, y)` (offset-based), `goto(url)`, and `go_back()`—are mapped to typed global virtual nodes (`virtual:viewport` and `virtual:browser_navigation`) $v^{\text{scroll}}_{\text{virt}}, v^{\text{nav}}_{\text{virt}} \in V_{\text{virt}}$.
- **Tab Operations**: Tab management operations—`new_tab()`, `tab_close()`, and `tab_focus(idx)`—are assigned to a session-level tab virtual node (`virtual:browser_tabs`) $v^{\text{tab}}_{\text{virt}} \in V_{\text{virt}}$.
- **Terminal Task Responses**: Terminal actions that communicate directly with the user, e.g., `respond(answer)` (implemented as `send_msg_to_user`), are mapped to a user-facing virtual node (`virtual:user`) $v^{\text{user}}_{\text{virt}} \in V_{\text{virt}}$.

Virtual Node Representation. Virtual nodes ensure that every successfully resolved action in a demonstration trajectory maintains a topological presence in G.[1] Transitions between concrete elements and virtual nodes (e.g., scrolling down to locate a product link, or navigating back to a search page) are seamlessly recorded as directed temporal edges in E_τ, preserving complete trajectory flow.

Structural Decomposition of the Non-GMCP Residual. Grounded Minimal Contrastive Pairs (GMCPs) require two criteria to be fulfilled simultaneously: (i) both candidate target elements must be physically present and uniquely resolvable in the active observation snapshot O_t, and (ii) both candidate actions must share the exact same operation type ($\text{op}^+ = \text{op}^-$). As reported in Section 4.2, SURFPRM-DATA achieves a 74.60% GMCP proportion (7,326 out of 9,821 verified pairs).

An audit of the remaining non-GMCP pairs confirms that 0% fail due to target elements being absent from the active AXTree snapshot; all proposed negative targets resolve uniquely to active BIDs in O_t. Instead, the 25.40% residual decomposes cleanly into two structural sources:

1. **Virtual Positive Steps (16.6%)**: In web navigation demonstrations, essential steps frequently involve global non-DOM operations (e.g., `scroll` to locate elements, `send_msg_to_user` to return task answers, or `go_back` to recover from dead ends). When such steps serve as the positive demonstration action a_t^+, their target corresponds to a virtual node $v_{\text{virt}} \in V_{\text{virt}}$, whereas the competing negative action a_t^- is sampled from the active page's concrete DOM elements in $\mathcal{V}_{\text{grounded}}(s_t)$. This contrast represents a crucial high-level strategic dilemma (e.g., "continue scrolling to find the target link" vs. "clicking an irrelevant distractor button on the visible viewport"), but inherently violates both identical target grounding and action-type matching.
2. **Affordance-Preserving Action-Type Mismatches (9.1%)**: For pairs where both positive and negative actions target genuine DOM elements in O_t, the negative candidate adopts the operation type that was historically demonstrated on that element during expert trajectories (or inferred from its AXTree role affordance, such as `fill` for text inputs). This affordance-aware assignment ensures that negative actions are executable and semantically meaningful rather than syntactically invalid (e.g., executing a `fill` on a non-editable link). Because proposal rules traverse topological, spatial, or temporal relations without coercing the negative candidate to adopt op^+, an action-type mismatch arises whenever the selected distractor element possesses a different native affordance from the positive target. In the sampled 10,000 candidate set, the most frequent mismatch directions are `fill` → `click` (422 pairs), `press` → `click` (137 pairs), `hover` → `click` (136 pairs), `select_option` → `click` (65 pairs), and `click` → others (110 pairs).

[1] A small fraction of successful actions (685 targets) cannot be resolved in the AXTree/SOM marker space and are therefore excluded from the graph rather than approximated; one `keyboard_type` action was discarded, and eight `mouse_click` targets were recovered through a manual BID audit table.

Together, these two factors account for 100% of the non-GMCP pairs (16.6% + 9.1% = 25.7% in the 10,000 candidate pool, aligning with the 74.60% verified training set), confirming that the sub-100% GMCP ratio is an intentional design consequence of supporting global workflow steps and element affordances rather than grounding failure.

C Implementation Details

Action-Only Input Isolation. As part of the concrete implementation of the SURFPRM data construction pipeline, we enforce an *Action-Only* input protocol across filtering, supervision synthesis, and PRM training. We first distinguish the actor's exploratory thought before generating an action (*candidate rationale* r_t) from the evaluator's comparative reasoning in hindsight (*judgment rationale* j_t). Concatenating candidate rationales into PRM inputs, as in prior pipelines (e.g., Zhang et al. (2026a)), allows models to exploit generator-specific stylistic signatures and confidence discrepancies instead of contextual decision semantics. Under our protocol, all candidate rationales are stripped away: the teacher model and the downstream PRM receive solely (s_t, a_t^1, a_t^2), forcing them to generate autonomous judgment rationales grounded purely in page affordances and task intent. Concretely, each training tuple is formatted as $d_t = (s_t, a_t^1, a_t^2, j_t^*, y_t^*)$, where s_t contains only the instruction, the AXTree observation, the action history, and the URL, and a_t^1, a_t^2 are rendered as raw action descriptions (operation type, target BID, and arguments) without any accompanying candidate rationale.

Teacher Verification Cost. All teacher verification and gold supervision synthesis calls use `gpt-5.6-sol`. Table 4 summarizes the cost of verifying the 10,000 sampled candidate pairs. In total, 11,210 API calls were issued (1.12 calls per pair on average): 9,528 pairs (95.3%) passed on the first attempt, 293 passed after one to three retries (168 on the second, 67 on the third, 40 on the fourth, and 18 on the fifth attempt), and 179 pairs failed all five attempts and were discarded as suspected false negatives, consistent with the 98.21% agreement pass rate reported in Section 4.2. The run consumed 35.6M input tokens (∼3.1k per call) and approximately 5.3M billed output tokens (∼472 per call), corresponding to an average cost of $0.0334 per pair ($0.0299 per call). Combined with contextual value completion for `fill` (199 calls, $< $1), the total teacher-side production cost of SURFPRM-DATA amounts to approximately $335.

Table 4: Teacher Verification and Gold Supervision Synthesis Cost for the 10,000 Sampled Candidate Pairs.

Metric	**Value**
Judged pairs	10,000
API calls	11,210
First-attempt pass	9,528 (95.3%)
Passed after retries	293
2nd / 3rd / 4th / 5th attempt	168 / 67 / 40 / 18
Exhausted all 5 attempts (filtered)	179
Average calls per pair	1.12
Input tokens	35.6M (∼3.1k per call)
Output tokens (billed)	∼5.3M (∼472 per call)
Average cost per pair	$0.0334
Average cost per call	$0.0299

D Prompt Specifications: Alignment Filtering, Gold Synthesis, Fill Value Completion, and JEV Evaluation

D.1 Alignment Filtering Prompt Specification

To ensure that the Teacher Model operates strictly under the SURFPRM protocol during alignment filtering, we adopt the WebArbiter-style pairwise evaluation template shown below. The two candidates are presented as *Response 1* and *Response 2*, whose order is randomized per pair

(fair coin), with the true position of the grounded-correct action recorded as the ground-truth label $y_t^* \in \{\text{response_1}, \text{response_2}\}$.

[System]
You are a skilled expert at evaluating assistant responses. You should evaluate given responses based on the judging criteria. Given the context of the conversation and two responses from the Assistant, you need to refer to determine the better response. Provide an overall comprehensive comparison upon them.

[User Input]
`#### Intent ####` {`instruction`}
`#### 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. {`accessibility_tree`}
`#### Trajectory ####` Note: The trajectory contains the sequence of previous actions performed by the agent. {`action_history`}
`#### 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]`
`ACTION:` {`action_1`}
`[The End of Response 1]`
`[The Begin of Response 2]`
`ACTION:` {`action_2`}
`[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.

The teacher decodes with temperature 0.2 under a fixed per-attempt seed. Given the extracted verdict Verdict $\in \{\text{Response 1}, \text{Response 2}\}$, filtering proceeds in up to $K{=}5$ ***sequential*** re-evaluation attempts. A pair is discarded if and only if the verdict still disagrees with y_t^* after all attempts.

D.2 Gold Judgment Rationale and Verdict Synthesis

For pairs that pass the alignment filtering, the assistant response from the ***first*** attempt whose verdict coincides with y_t^* is retained as the gold response; no further selection is applied among multiple aligned trials. The retained gold text naturally decomposes into a reasoning prefix $j_t^* = \langle \texttt{State} \rangle \circ \langle \texttt{Criteria} \rangle \circ \langle \texttt{Analysis} \rangle$ followed by the verdict token $y_t^* = \langle \texttt{Answer} \rangle$, so the final supervision target is

$$Y_t = j_t^* \circ y_t^*, \tag{8}$$

which is inserted verbatim as the assistant message of the SFT triple (system, user, assistant).

D.3 Contextual Fill Value Completion Prompt Specification

As outlined in Section 4.3 and Appendix F, when candidate negative proposals select a text-entry element via graph mining that was not interacted with during the demonstration trajectory, the negative action lacks a concrete string argument. Rather than inserting trivial dummy text or risking format hallucinations, we prompt an LLM to generate a concise, syntactically valid, and contextually plausible input value that differs from any existing content in the field. The completion prompt consists of a fixed system instruction and a state-conditioned user prompt:

[System]
You complete the text value for a fixed browser fill action. The target BID and action type are immutable. Return only a JSON object with one string field named value. Choose a concise, non-empty value that is syntactically valid and locally plausible for the target input in the current page state. The value must differ from every current value listed in the request. Do not output an action, BID, explanation, Markdown, or extra fields.

[User Input]

```
User instruction:
{instruction}
Current URL:
{current_url}
Previous actions:
{action_history}
Fixed negative target BID: {target_id}
Target description: {target_name}
Current AXTree evidence:
{target_evidence}
Values already present in the target (do not repeat):
{current_values}
Known positive action (do not copy its BID):
{positive_action}
Supply only the value for fill(<fixed BID>, value).
```

The completion model decodes with temperature 0.2.

D.4 Jev Evaluation Prompt Specifications

For the proprietary decision model `jev-1.13` (TypeSafe AI, 2026) evaluated in Table 2, inputs are structured into typed state blocks (`State`) and decision queries (`Question`) according to the Jev API specification. We evaluate two decision modes: **Choice mode** and **Noul mode**.

In both modes, the input context consists of two shared state blocks: `[State: judging_guidance]` providing overall evaluation directives, and `[State: web_agent_context]` providing task intent, accessibility tree observation, trajectory history, URLs, and the two candidate assistant responses.

[State: judging_guidance]
You are a skilled expert at evaluating assistant responses. You should evaluate given responses based on the judging criteria.
Given the context of the conversation and two responses from the Assistant, you need to refer to determine the better response. Provide an overall comprehensive comparison upon them.
Be objective and base the evaluation strictly on the content of the two responses. Do not let the response order or response length bias the judgment.

[State: web_agent_context]

```
#### 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.

```
{axtree}
#### Trajectory ####
```

Note: The trajectory contains the sequence of previous actions performed by the agent.

{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]
{response_1}
[The End of Response 1]
[The Begin of Response 2]
{response_2}
[The End of Response 2]

Under **Choice Mode**, the question block is specified as:

[Question: better_response]
type: choice
instructions: Compare the assistant's Response 1 and Response 2 found in state.web_agent_context and decide which one is the better next step for the web agent. Follow the judging guidance in state.judging_guidance, and check each response against the intent, the AXTree of the current page, the trajectory of previous actions, and the start/current URLs. Prefer the response whose thought and action are more correct, better grounded in the current page state, and more effective at progressing toward the intent.
criteria:
Response 1: Response 1, the block between '[The Begin of Response 1]' and '[The End of Response 1]', is the better response: compared with Response 2 it is more correct, better grounded in the current page state, and more effective for the intent.
Response 2: Response 2, the block between '[The Begin of Response 2]' and '[The End of Response 2]', is the better response: compared with Response 1 it is more correct, better grounded in the current page state, and more effective for the intent.

Under **Noul Mode**, the two [State] blocks remain identical, and only the [Question] block is replaced with:

[Question: better_response]
type: noul
instructions: Comparing the assistant's Response 1 and Response 2 found in state.web_agent_context under the judging guidance in state.judging_guidance, is Response 1 the better next step for the web agent?
criteria:
true: Response 1 is the better response: it is more correct, better grounded in the current page state, and more effective for the intent than Response 2.
false: Response 2 is the better response, or the two responses are indistinguishable in quality.

E Extended Evaluation and Repeated Runs

Ablation on Graph Confusion Axes. Table 5 details the full benchmark performance of comparative Web PRMs fine-tuned on individual confusion axes (Spatial, Temporal, and Spatiotemporal) versus the complete multi-axis mixture using Qwen2.5-3B-Instruct. Because the temporal axis naturally yields fewer negative candidates from expert demonstration trajectories (470 pairs vs. >4k pairs for spatial and spatiotemporal axes; Table 1), we scale up its fine-tuning epochs so that the cumulative training compute and gradient update steps are aligned to the same order of magnitude across all single-axis configurations.

Table 5: **Ablation on Graph Confusion Axes.** Performance comparison of comparative Web PRMs fine-tuned on individual graph confusion axes versus the complete multi-axis mixture (evaluated on WEBPRMBENCH with Qwen2.5-3B-Instruct). Pairwise accuracy and state-level Best-of-N (BoN) accuracy are reported in percent (%). The row corresponding to our full confusion mixture is shaded, and **bold** indicates the best result in the Average columns.

Model / Training Configuration	WebArena		AssistantBench		WorkArena		Average	
	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN
Qwen2.5-3B-Instruct	70.02	35.82	78.33	46.67	68.99	29.25	72.45	37.25
+ Only-Temporal	74.63	45.27	78.33	33.33	65.45	26.42	72.80	35.01
+ Only-Spatial	81.84	57.71	75.00	50.00	73.70	40.57	76.85	49.43
+ Only-Spatiotemporal	82.21	53.73	78.33	43.33	73.47	33.02	78.00	43.36
+ All Axes	81.59	55.22	85.00	56.67	75.35	41.98	**80.65**	**51.29**

Table 6: **Effect of the Grounded Minimal Contrastive Pair (GMCP) Ratio.** Qwen2.5-3B-Instruct results with approximately 10k SFT pairs per fine-tuned variant. Pairwise and state-level Best-of-N (BoN) accuracies are reported in percent (%). The best results in the Average column are in **bold**.

Model / SFT Data	GMCP Ratio	WebArena		AssistantBench		WorkArena		Average	
		Pairwise	BoN	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN
Qwen2.5-3B-Instruct	–	70.02	35.82	78.33	46.67	68.99	29.25	72.45	37.25
+ WebArbiter 10k	24.19%	81.84	55.22	69.17	40.00	71.58	35.38	74.20	43.53
+ GMCP40 10k	40.00%	81.47	50.75	70.83	26.67	74.06	39.62	75.45	39.01
+ GMCP60 10k	60.00%	82.71	56.72	76.67	36.67	74.29	40.09	77.89	44.49
+ Ours 10k	74.60%	81.59	55.22	85.00	56.67	75.35	41.98	**80.65**	**51.29**

Table 7: **Ablation on Graph Necessity.** Comparison of comparative Web PRMs trained with the WebArbiter baseline, the No-Graph variant, and SURFPRM on WEBPRMBENCH. Pairwise accuracy and state-level Best-of-N (BoN) accuracy are reported in percent (%). The best results in the Average columns are in **bold**.

Model / Training Configuration	WebArena		AssistantBench		WorkArena		Average	
	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN
Qwen2.5-3B-Instruct	70.02	35.82	78.33	46.67	68.99	29.25	72.45	37.25
+ WebArbiter 10k SFT	81.84	55.22	69.17	40.00	71.58	35.38	74.20	43.53
+ No-Graph 10k SFT	84.45	59.70	70.00	30.00	72.05	36.32	75.50	42.01
+ Ours 10k SFT	81.59	55.22	85.00	56.67	75.35	41.98	**80.65**	**51.29**

Effect of the Grounded Minimal Contrastive Pair (GMCP) Ratio. Table 6 details the full benchmark performance of the GMCP-ratio ablation discussed in Section 5.3, comparing Qwen2.5-3B-Instruct fine-tuned on $\sim$10k SFT pairs with controlled GMCP ratios against the un-finetuned base model and the WebArbiter baseline.

Graph Necessity Ablation. Table 7 details the full benchmark performance of the Graph Necessity ablation discussed in Section 5.3, comparing SURFPRM against the No-Graph variant and the WebArbiter baseline.

Repeated Evaluation Runs and Statistical Stability. Table 8 reports the pairwise accuracy and state-level Best-of-N (BoN) accuracy across 3 repeated evaluation runs on WEBPRMBENCH using Qwen3.5-9B.

The results show low variance across runs (average Pairwise standard deviation is $\pm 0.20\%$, and average BoN standard deviation is $\pm 0.24\%$). On overall macro-average, our model fine-tuned on SURFPRM-DATA demonstrates consistent improvements over both the un-finetuned base model and the WebArbiter 10k baseline (reaching $86.22 \pm 0.20\%$ Pairwise and $68.01 \pm 0.24\%$ BoN, compared to

Table 8: **Multiple Repeated Evaluation Results on WEBPRMBENCH.** Performance of comparative Web PRMs across repeated evaluation across 3 runs. Pairwise Accuracy (%) and State-level Best-of-N (BoN) Accuracy (%) are reported as mean ± std. Rows fine-tuned on our synthesized data (SURFPRM-DATA) are shaded, and **bold** indicates the best mean result in the Average columns.

Model	WebArena		AssistantBench		WorkArena		Average	
	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN	Pairwise	BoN
Qwen3.5-9B	81.88 ± 0.50	68.32 ± 1.25	81.67 ± 0.84	66.67 ± 3.34	78.69 ± 0.68	54.72 ± 0.95	80.75 ± 0.21	63.24 ± 1.58
+ WebArbiter 10k SFT	85.28 ± 0.36	68.33 ± 0.29	89.72 ± 0.96	71.11 ± 1.92	77.48 ± 0.20	52.83 ± 0.81	84.16 ± 0.51	64.09 ± 1.01
+ Ours 10k SFT	87.31 ± 0.57	70.98 ± 1.04	87.78 ± 1.74	71.11 ± 3.85	83.57 ± 0.65	61.95 ± 2.76	**86.22** ± 0.20	**68.01** ± 0.24

$84.16 \pm 0.51\%$ and $64.09 \pm 1.01\%$), with clear gains on WebArena and WorkArena while maintaining competitive performance on AssistantBench, confirming the statistical robustness of our gains.

F DATA QUALITY ASSURANCE PIPELINE DETAILS

Parameter Completion for Fill Actions. Among the 262 `fill` negatives produced by graph traversal in the mined pool, 199 lacked a concrete input string value, since the candidate element was not typed into during the reference trajectory under identical context. To ensure environmental executability, we prompt an LLM with the page context and target input box attributes to generate a contextually plausible but incorrect or distracting input string (e.g., typing an incompatible postal code format or an alternative movie name) for each empty value (the full prompt specification and decoding protocol are detailed in Appendix D.3). In addition, 11 non-editable targets were normalized to `click` (Appendix F, Combobox Control Correction), yielding 251 `fill` negatives in the pool. The sampled set of 10,000 pairs contains 73 `fill` negatives, of which 68 survive in the final verified set of 9,821 pairs. Quality audits on the final set confirmed 0 empty values, 0 repetitions of current element values, and 0 invalid string formats.

Combobox Control Correction. During initial auditing, we observed that certain custom dropdown elements (e.g., ARIA role `combobox` without an underlying editable `<input>` tag) were assigned `fill` operations by heuristics. Attempting to execute `fill` on such elements causes runtime execution errors in browser environments. We audited all such controls and remapped their actions to `click`, resolving the control selection via standard dropdown click-and-select semantics.