

FailAlign: Failure-Aligned Bootstrapping for Long-Horizon HTML Based Code Generation Agents

Fanglin Xu¹, Wei Zhang², Jian Yang², Guo Chen^{1*}, Liquan Yang², Zhoujun Li²

¹Hunan University; ²Beihang University

xf1825@hnu.edu.cn, zwpride@buaa.edu.cn, guochen@hnu.edu.cn

Abstract

LLM-based web agents automate browser interactions by generating structured action sequences, yet their deployment in long-horizon tasks is hindered by fragile error propagation in noisy DOM trees. Critically, existing training paradigms rely exclusively on positive supervision from gold demonstrations, teaching models **what** is correct but never **why** plausible alternatives fail, leaving agents unable to distinguish between subtly different yet critically divergent choices. We propose **FailAlign**, a failure-aligned learning framework that systematically integrates error signals as first-class training supervision. We mine a Failure Index from model rollouts by pinpointing first-error steps, then leverage these instances across a three-stage pipeline: (i) action-grounded context compression distills verbose HTML into compact DOM snippets; (ii) hard-negative DPO pairs gold actions with semantically similar incorrect alternatives to sharpen decision boundaries; (iii) trajectory-level GRPO mitigates error cascades by re-simulating corrected trajectories. On Mind2Web with Qwen3-32B, FailAlign achieves **49.7% SSR** and **5.6% SR**, surpassing models 15 $\times$ larger (Qwen3-Coder-480B: 37.0% SSR) and commercial systems (Claude Opus-4.5: 39.4% SSR), validating the efficacy of learning from failures.

1 Introduction

LLM-based web agents have emerged as a transformative paradigm for autonomous browser automation, aiming to emulate human-like interactions in completing complex real-world tasks across diverse domains (Nakano et al., 2022; Yao et al., 2022; Deng et al., 2023). However, deploying these agents faces critical challenges: i) real-world web pages contain noisy DOM trees with thousands of elements and spurious distractors; ii) minor missteps cascade into trajectory failures as downstream

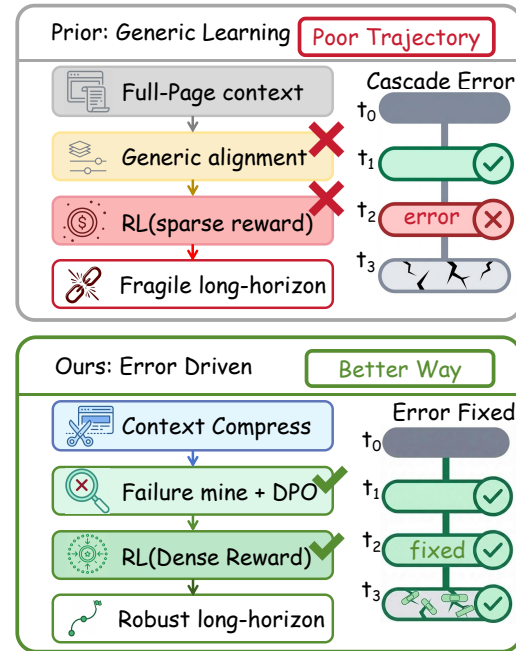


Figure 1: FailAlign helps LLMs fix error and make trajectory more robust.

steps depend on prior state transitions; iii) existing training paradigms rely exclusively on **positive supervision** from gold trajectories, teaching models **what** is correct but never encoding **why plausible alternatives fail**. Consequently, agents lack the contrastive discriminative capacity to distinguish between subtly different yet critically divergent choices, leading to compounding errors in long-horizon plans (Figure 1).

To address these challenges, we propose **FailAlign**, a failure-aligned learning framework that systematically integrates error signals as first-class training supervision. Our approach tackles the aforementioned problems through three complementary mechanisms: i) **action-grounded context compression** distills verbose HTML into compact DOM snippets to reduce input noise; ii) **hard-negative preference learning** via DPO (Rafailov

*Corresponding author.

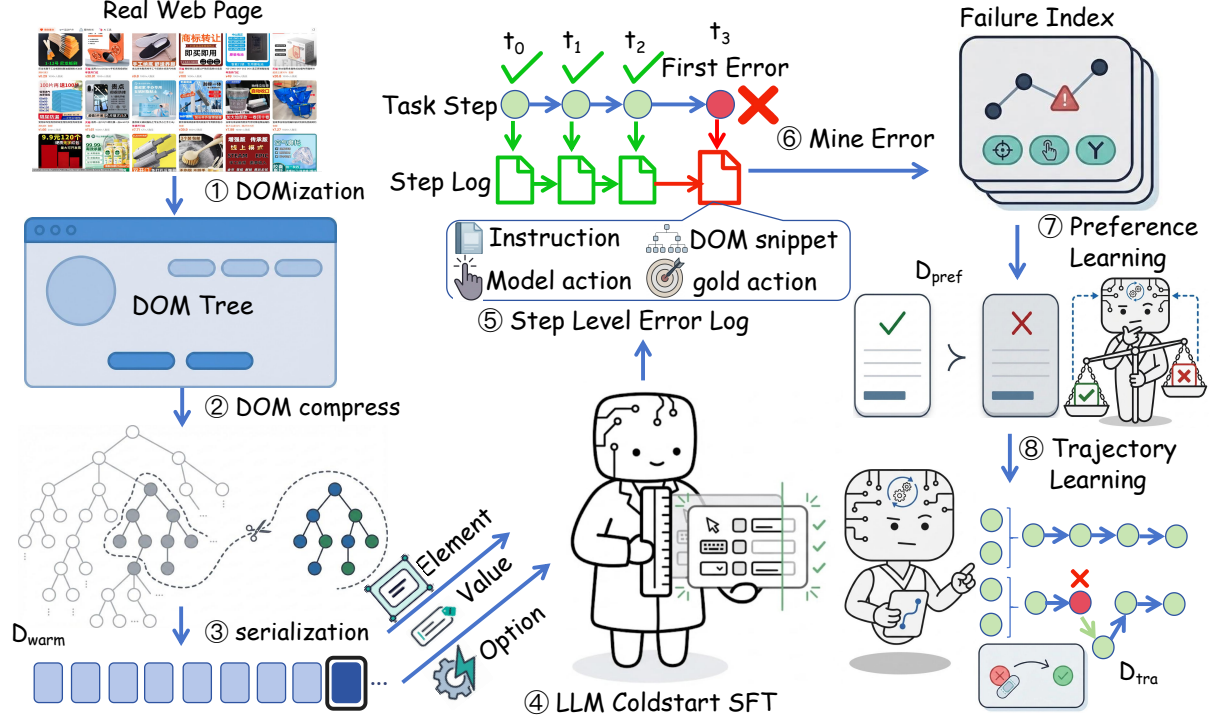


Figure 2: Overview of the FailAlign pipeline. The process begins with DOMization, compression, and serialization of raw web pages to construct structured inputs for LLM Coldstart SFT. During task execution, the system records granular step-level logs, which are subsequently mined to pinpoint the First Error and aggregate these instances into a unified Failure Index. This error-centric dataset drives two distinct optimization phases: Preference Learning, which contrasts specific correct and incorrect actions, and Trajectory Learning, which operates at the trajectory level to refine the model’s holistic planning capabilities by learning from complete task execution paths.

et al., 2023) pairs gold actions with semantically similar incorrect alternatives to sharpen local decision boundaries; iii) **trajectory-level GRPO** (Shao et al., 2024) mitigates error cascades by re-simulating corrected trajectories from first-error states. We mine a unified Failure Index from model rollouts and leverage these instances across the three-stage pipeline. Instantiated on Mind2Web (Deng et al., 2023) with Qwen3-32B (Yang et al., 2025), FailAlign-RL achieves **49.7% SSR** and **5.6% SR**, surpassing larger models (Qwen3-Coder-480B: 37.0% SSR; DeepSeek-V3.2: 26.1% SSR) and Closed-Source LLMs (GPT-4o: 22.9% SSR; Claude Opus-4.5: 39.4% SSR).

Our **key contributions** are threefold:

- We introduce FailAlign, the failure-aligned learning framework for web agents that systematically mines and integrates error signals from model rollouts, enabling explicit learning from both correct and incorrect action sequences to enhance contrastive decision-making.
- We design a three-stage training pipeline combining i) action-grounded DOM compression to reduce input noise, ii) hard-negative DPO for lo-

cal disambiguation over confusable actions, and iii) trajectory-level GRPO for global error cascade mitigation, each targeting distinct failure modes in long-horizon web navigation.

- We achieve state-of-the-art results on Mind2Web across Task/Domain/Website splits, outperforming models up to $15\times$ larger in parameters and leading commercial systems, with consistent gains in both step-level accuracy (EAcc, F1, SSR) and the stricter long-horizon success rate (SR), validating the efficacy of learning from failures.

2 Method

2.1 Problem Formulation

We study long-horizon HTML-based action code generation under a *structured multiple-choice* formulation. Given an instruction x and the current webpage state s_t , we construct a step context c_t and an option set E_t that contains candidates at step t . The model outputs a structured choice $a_t = (o_t, e_t, v_t)$, where o_t denotes the operation type (a member of { select, type, click }), e_t indexes the target element selected from the candidate set

E_t , and v_t is null iff operation is click. Executing a_t updates the environment to s_{t+1} , yielding a trajectory $\tau = (s_0, a_0, \dots, a_{T-1}, s_T)$ where T denotes the termination timestep. This formulation enables deterministic step-level evaluation by comparing the predicted choice with the gold choice, which in turn defines failure traces from interaction logs and supports failure-aligned training in later sections.

2.2 Failure-Aware Supervised Warm-Up

Action-Grounded Context Compression. As shown in Figure 2, we convert each raw webpage state s_t into a compact, action-grounded DOM snippet, yielding a compressed step context c_t . Given the candidate element set E_t , we prune the DOM by taking a small local neighborhood around each candidate and union them into c_t . We preserve visible text and salient attributes with value filtering and truncation, and relabel remaining nodes with consecutive ids for compact serialization.

Warm-Up Instance Construction. Following the generation of the compressed step context, we construct structured multiple-choice instances characterized by a fixed set of five distinct options. The supervision signal for this stage is the gold structured action $a_t = (o_t, e_t, v_t)$, where $e_t \in E_t$ specifically identifies the grounded target element from the provided candidates. In scenarios where the gold target is absent from the available options, we include a designated None choice to represent the “not in the list” condition. This construction process results in a supervised warm-up corpus $\mathcal{D}_{\text{warm}}$ containing approximately 77K instances. The primary objective of this dataset is to train the model to accurately align the relevant step context with the predefined option space while adhering to the required output format.

Supervised Warm-Up and Error Exposure. We train a base policy π_θ on $\mathcal{D}_{\text{warm}}$ and obtain FailAlign-SFT, which equips the model with basic competence for predicting structured actions under the stepwise multiple-choice format. We then roll out FailAlign-SFT on the training task pool and collect stepwise interaction logs, recording $(x, s_t, c_t, E_t, \hat{a}_t, s_{t+1})$ at step t , where s_t is the raw html, c_t is the compressed step context, and s_{t+1} is the state after executing $\hat{a}_t$.

Step-Level Error Mining Interface. On these logs, we apply a deterministic step evaluator and

assign a binary label $r_t \in \{0, 1\}$. A step is correct iff the operation type matches, the selected option corresponds to the gold target, and the value field matches when applicable. We collect all incorrect steps into a unified failure index $\mathcal{I} = \{(t, x, c_t, E_t, \hat{a}_t, a_t^*) \mid r_t = 0\}$ for subsequent preference and trajectory-level construction.

2.3 Hard-Negative Disambiguation

Objective. Given the failure index $\mathcal{I}$ in Figure 2, we focus on highly confusable step contexts in which the model makes a near-miss decision: the predicted action is locally plausible under the same options, yet deviates from the gold choice and leads to drastically different outcomes. Our goal is to construct a compact preference dataset $\mathcal{D}_{\text{pref}}$ that sharpens the decision boundary over operation type, and value fields under the same context.

Hard-Negative Construction. For each failure instance $(t, x, c_t, E_t, \hat{a}_t, a_t^*) \in \mathcal{I}$, we generate preference pairs using the compressed context c_t that was originally constructed during the supervised warm-up phase via the action-grounded DOM compression pipeline described in Section 2.2. Specifically, we extract all candidate negative actions from the stepwise interaction logs collected during FailAlign-SFT rollouts, where each negative action shares the exact same context (x, c_t, E_t) as the gold action a_t^* . To ensure physical validity, we first filter these candidates by verifying successful environment execution, an action is retained only if it produces a valid state transition $s_t \rightarrow s'_{t+1}$ without triggering any runtime exceptions, particularly “element not found” errors that would break the task flow. From this executable pool, we identify the most challenging hard negatives a_t^- by computing semantic embeddings with Qwen3-Embedding-8B (Zhang et al., 2025) and selecting actions with maximum cosine similarity to the gold action a_t^* . This similarity-based selection targets near-miss decisions where the model’s predicted action appears visually and functionally plausible within the compressed DOM context c_t , yet leads to divergent task trajectories compared to the gold action. Each validated pair (a_t^*, a_t^-) is then formatted into a preference comparison $(x, c_t, E_t, a_t^* \succ a_t^-)$. After applying these dual constraints of executability and semantic confusability across all failure instances in $\mathcal{I}$, we obtain $\mathcal{D}_{\text{pref}}$, a refined dataset of approximately 43K high-signal comparisons that specifically trains the policy to resolve ambiguities

Model	Size	Task				Domain				WebSite			
		EAcc	F1	SSR	SR	EAcc	F1	SSR	SR	EAcc	F1	SSR	SR
Closed-Source LLMs													
GPT-4o	🔒	26.1	63.1	22.9	1.6	30.1	64.2	26.8	1.4	24.9	59.8	20.9	0.6
GPT-5.1-2025-11-13	🔒	38.3	72.8	33.5	2.0	38.3	70.8	33.4	2.7	36.9	68.5	38.7	2.3
Gemini-2.5-flash	🔒	39.3	70.8	34.4	3.8	38.0	70.2	33.5	2.9	36.9	67.5	30.0	2.2
Gemini-2.5-pro	🔒	39.3	70.8	34.4	3.6	40.1	69.3	35.7	3.1	39.1	70.1	31.2	2.2
Claude-haiku-4-5-20251001	🔒	41.4	74.6	33.1	2.0	39.3	71.7	33.8	2.4	35.5	69.1	29.2	1.1
Claude-sonnet-4-5-20250929	🔒	48.5	75.2	40.6	2.8	45.3	71.2	46.7	2.3	43.6	68.9	35.3	0.6
Claude-opus-4-5-20251101	🔒	50.1	66.2	39.4	3.1	44.6	63.4	37.3	1.8	43.5	64.2	36.5	1.9
Open-Source LLMs													
Qwen3-0.6B	0.6B	7.0	3.5	0.7	0.0	8.6	6.9	1.0	0.0	6.5	6.0	0.7	0.0
Qwen3-1.7B	1.7B	13.5	44.1	8.3	0.4	16.3	42.9	10.0	0.3	13.1	40.1	6.6	0.0
Qwen3-4B	4B	21.0	53.6	15.6	0.2	23.6	54.9	17.9	0.2	19.7	50.4	12.7	0.3
Seed-Coder-8B-Instruct	8B	13.8	65.8	12.4	1.2	14.4	65.0	13.1	0.4	11.3	62.3	9.2	0.0
Qwen3-8B	8B	24.5	63.9	20.2	1.2	25.4	65.2	21.6	1.9	21.6	60.5	16.6	0.6
Qwen3-14B	14B	28.0	60.7	21.6	1.7	29.0	51.5	23.3	1.5	23.7	57.7	17.2	0.6
Qwen3-30B-A3B-Instruct-2507	3/30B	24.0	67.0	21.1	1.6	27.2	66.9	23.5	1.4	22.7	64.1	17.8	0.0
Qwen3-30B-A3B-Thinking-2507	3/30B	27.3	65.5	24.6	1.3	24.1	61.6	20.9	1.0	26.1	62.7	21.1	1.1
Qwen3-Coder-30B-A3B-Instruct	3/30B	31.3	71.4	28.3	1.4	31.7	70.6	29.0	2.4	26.5	66.5	22.7	1.7
QwQ-32B	32B	30.0	70.0	26.0	1.6	31.4	69.2	27.5	2.1	28.2	62.8	22.4	0.6
Qwen2.5-Coder-32B-Instruct	32B	24.6	64.8	21.9	2.0	25.8	64.2	22.6	1.4	21.3	60.0	18.0	0.0
Qwen3-235B-A22B-Thinking-2507	22/235B	37.4	68.4	31.7	2.8	36.2	68.1	31.1	3.0	33.9	64.4	27.7	1.1
Qwen3-235B-A22B-Instruct-2507	22/235B	30.6	67.2	26.7	1.5	30.9	68.3	27.6	2.8	28.7	64.6	22.7	1.1
GLM-4.7	32/358B	41.4	64.7	34.5	2.1	38.5	62.7	32.7	2.3	39.7	58.7	30.1	1.8
MiniMax-M2.1	10/230B	32.1	69.4	27.5	1.2	31.7	66.9	27.3	2.5	28.0	63.3	21.7	0.9
Qwen3-Coder-480B-A35B-Instruct	35/480B	40.9	74.6	37.0	2.8	37.6	72.1	34.4	3.2	36.1	70.0	30.6	2.4
DeepSeek-V3.2	37/671B	30.3	67.4	26.1	2.0	32.4	67.4	27.8	2.2	30.2	62.7	22.7	1.7
Kimi-K2-Thinking	32B/1T	40.5	72.2	34.8	2.4	38.9	70.7	34.3	3.7	36.9	67.1	30.0	1.1
Kimi-K2-Instruct-0905	32B/1T	38.3	70.6	30.1	2.1	38.9	64.4	31.8	1.9	37.4	65.0	31.6	1.8
Qwen3-32B	32B	30.7	65.4	24.9	2.4	31.7	66.7	26.8	2.2	30.2	62.6	22.7	1.1
Qwen3-32B-Raw-HTML	32B	33.0	69.1	28.1	2.4	32.3	68.6	28.4	2.3	30.9	65.0	24.6	1.6
Our Method													
FailAlign-SFT	32B	50.1	77.8	45.8	4.8	43.6	70.9	39.2	3.0	41.9	68.7	37.0	2.3
FailAlign-DPO	32B	52.5	78.9	47.5	5.1	44.2	72.0	40.2	3.4	42.3	70.0	38.1	2.8
FailAlign-RL	32B	54.6	79.4	49.7	5.6	46.8	74.6	42.1	3.8	44.5	72.7	39.6	3.5

Table 1: Results on three test sets.

between actions with near-identical representations in c_t but critically different functional outcomes in the actual webpage environment.

Usage. We optimize the policy π_θ on the preference dataset $\mathcal{D}_{\text{pref}}$ using Direct Preference Optimization (DPO). This approach explicitly shapes the policy’s likelihood distribution to satisfy $\pi_\theta(a_t^* | x, c_t, E_t) \succ \pi_\theta(a_t^- | x, c_t, E_t)$ for all preference pairs, where the identical step context (x, c_t, E_t) ensures the comparison isolates decision quality from contextual variations. The resulting policy exhibits refined discrimination capability between semantically similar actions that would otherwise lead to divergent task outcomes.

2.4 Mitigating Error Cascades in Long Trajectories

Long-Horizon Goal. Beyond local correction and hard-negative disambiguation, our final goal is

Dataset	# Instances	Avg. Tokens
$\mathcal{D}_{\text{warm}}$	77,750	442.3
$\mathcal{D}_{\text{pref}}$	43,205	1,997.1
$\mathcal{D}_{\text{tra}}$	1,009	15,375.7

Table 2: Overview of instance counts and average input length across training datasets.

to mitigate error cascades in which early near-miss decisions compound into downstream failures over long horizons. Given an instruction x and an initial state s_0 , we aim to learn a policy π_θ that improves trajectory-level robustness while maintaining per-step correctness.

Trajectory-Level Signal Construction. Leveraging the failure index $\mathcal{I}$ from Section 2.2, we construct trajectory-level supervision by identifying tasks containing a *first failure* point. For each task, we form the rollout trajectory $\hat{\tau} = (s_0, \hat{a}_0, \dots, s_T)$ as defined in Section 2.1. Let t^* be the first failure

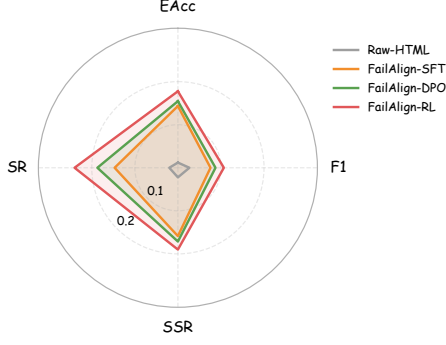


Figure 3: Normalized performance deltas using Qwen3-32B as the base. It visualizes the improvements across four metrics, all scaled to the [0, 1] range, illustrating the progressive gains of each training stage.

step from $\mathcal{I}$ (i.e., the earliest t where $\hat{a}_t \neq a_t^*$). We repair $\hat{\tau}$ by **resetting the environment to the golden state** $s_{t^*}^*$ recorded during warm-up data collection, then appending the corresponding gold suffix.

$$\tau^* = \underbrace{(s_0, \hat{a}_0, \dots, \hat{a}_{t^*-1})}_{\text{executed prefix}} \oplus \underbrace{(s_{t^*}^*, a_{t^*}^*, \dots, s_T^*)}_{\text{gold suffix}} \quad (1)$$

where $\oplus$ denotes sequence concatenation. This reset ensures the gold suffix executes in the intended context despite prefix errors. The resulting dataset $\mathcal{D}_{\text{tra}}$ comprises approximately 1,000 repaired trajectories, concentrating supervision on first failure correction while preserving trajectory structure from Section 2.1.

Usage. We use $\mathcal{D}_{\text{tra}}$ as the task pool for Group Relative Policy Optimization (GRPO): for each instruction x , we sample a group of trajectory completions from π_θ and optimize group-relative advantages under a trajectory-level reward. This training encourages globally consistent planning over greedy local steps and mitigates error cascades without a separate critic.

3 Experiment

3.1 Implementation Details

We utilize Qwen3-32B as the foundational model for all experiments. Our training pipeline sequentially progresses through supervised fine-tuning on the initial and failure-aligned corpora ($\mathcal{D}_{\text{warm}}$), followed by preference alignment via DPO ($\mathcal{D}_{\text{pref}}$) and trajectory-level optimization via GRPO ($\mathcal{D}_{\text{tra}}$). All models are trained on 8 NVIDIA A800 GPUs

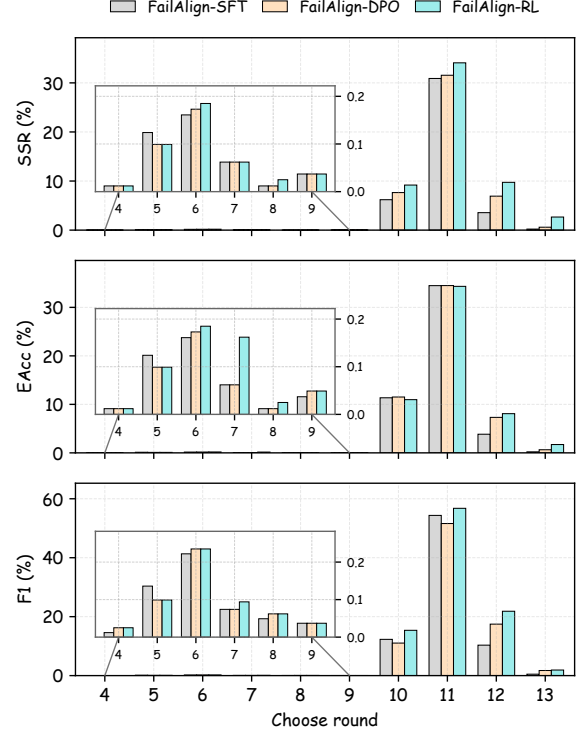


Figure 4: Round-wise SSR, EAcc and F1 comparison of SFT, DPO, and RL in FailAlign. The inset zooms into the early rounds (4-9) for clearer visibility.

using LLaMA-Factory¹ and Verl². For the supervised stages, we adopt a learning rate of 5×10^{-5} with a cosine decay scheduler and a global batch size of 256, truncating inputs to 8192 tokens. In the subsequent alignment stages, we reduce the learning rate to 1×10^{-6} for DPO (with $\beta = 0.1$) and 5×10^{-7} for GRPO with a group size of 5 to ensure stable convergence.

3.2 Evaluation Metrics

Element Accuracy (EAcc). Element Accuracy indicates the correctness of the chosen element: $\text{EAcc} = \frac{1}{N} \sum_{t=1}^N \mathbb{I}[\hat{e}_t = e_t^*]$.

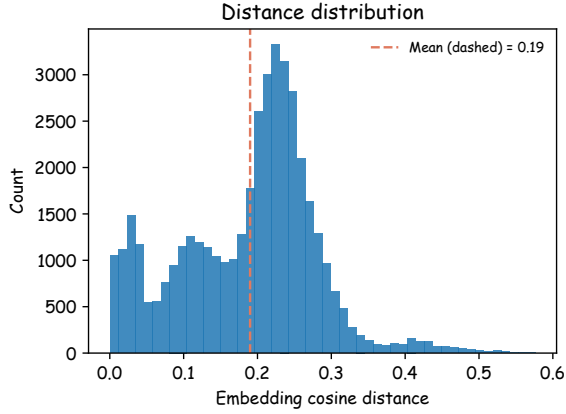
Operation F1. Operation F1 evaluates operation prediction at the token level. We compute precision P and recall R between predicted and gold operation tokens: $\text{F1} = \frac{2PR}{P+R}$.

Step Success Rate (SSR). SSR counts a step as successful only when both the target element is right and the operation prediction is fully correct: $\text{SSR} = \frac{1}{N} \sum_{t=1}^N \mathbb{I}[(\hat{o}_t = o_t^*) \wedge (\hat{e}_t = e_t^*)]$.

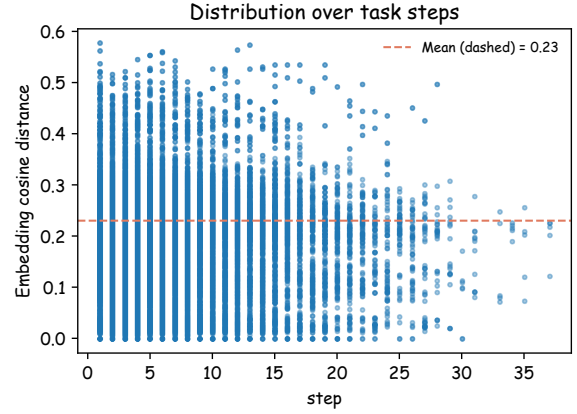
Task Success Rate (SR). SR measures the proportion of tasks completed with all steps successful: $\text{SR} = \frac{1}{M} \sum_{i=1}^M \mathbb{I}[\forall t \in \tau_i, \text{step}_t \text{ is successful}]$.

¹<https://github.com/hiyouga/LLaMA-Factory>

²<https://github.com/volcengine/verl>



(a) Distribution of positive and negative examples in the $\mathcal{D}_{\text{pref}}$.



(b) Distribution of candidate difficulty and steps in the $\mathcal{D}_{\text{tra}}$.

Figure 5: Difficulty distribution of DPO and RL stages.

SR is a stricter, long-horizon metric and is typically much lower than SSR.

3.3 Evaluation Benchmark

Mind2Web. The Mind2Web dataset (Deng et al., 2023) is a fundamental benchmark designed to evaluate generalist web agents under real-world settings. It comprises exactly 2,350 open-ended tasks collected from 137 diverse websites spanning 31 domains, covering varying topics such as travel, shopping, and finance. Unlike synthetic environments, Mind2Web requires agents to interpret raw HTML and perform step-by-step interactions to complete complex instructions. The test set is stratified into three distinct levels: Cross-Task, Cross-Website, and Cross-Domain, to comprehensively measure the agent’s generalization capabilities across unseen tasks and web environments.

3.4 Top-K Rolling Evaluation

Given the diversity of DOM elements, we adopt a relevance-based filtering strategy to retain a top-ranked candidate subset. Specifically, we rank candidates based on their relevance scores with respect to the task and select the top- K elements (defaulting to 50 in our experiments). We further split these candidates into several groups and let the model make selections under group-specific DOM context views. To mitigate potential LLM noise in single-shot matching, we use a lightweight rolling scheme that appends previously preferred options to later candidate sets. This multi-round re-contrast encourages the model to converge to a more stable final choice without over-specifying evaluation heuristics.

3.5 Evaluation Baselines

Closed-Source Models. The evaluation includes frontier proprietary baselines represented by OpenAI’s GPT series (OpenAI, 2024, 2025a), Google’s Gemini 2.5 Pro and Flash (Comanici et al., 2025), and Anthropic’s Claude Sonnet series (Anthropic, 2025c). As widely recognized top-tier systems with proven efficacy in reasoning and code generation, these models serve as stringent comparators for our models.

Open-Source Models. We employ large-parameter general models, such as Qwen3-Coder-480B-A35B-Instruct (Qwen, 2025) and DeepSeek-V3.2 (DeepSeek-AI et al., 2025b), to provide high-performance baselines based on the correlation between model scale and reasoning capability. Furthermore, to isolate the efficacy of our method, a baseline is established using Qwen3-32B evaluated directly on the raw Mind2Web dataset (Deng et al., 2023).

3.6 Main Results

Table 1 shows the results on three test sets. Overall, FailAlign delivers consistent gains across Task, Domain, and Website, with steady improvements from SFT to DPO to RL, indicating that our training pipeline yields cumulative benefits rather than isolated wins. Compared to the same scale baseline, FailAlign-RL improves both SSR and SR, suggesting stronger decision quality and more stable long-horizon behavior. Importantly, FailAlign also outperforms the Raw-HTML baseline, showing that directly training on raw page inputs is insufficient, while our failure-driven way more effectively organizes supervision signals and improves training efficiency. Furthermore, when compared against

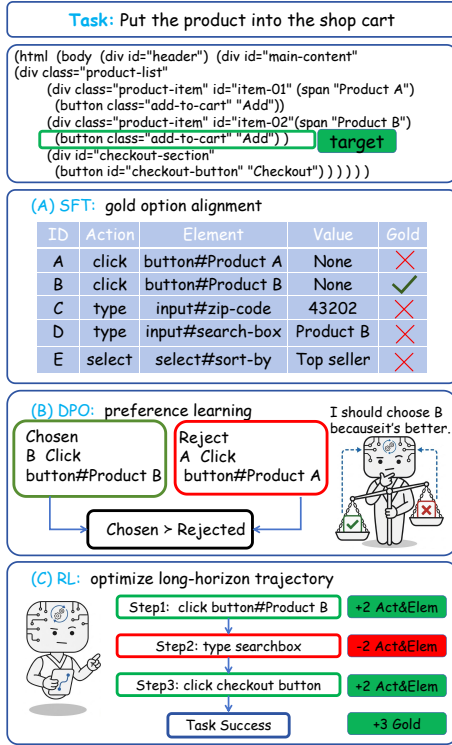


Figure 6: Illustration of data formats for different objectives. (A) Action selection based on gold labels. (B) Error driven preference learning using chosen-rejected pairs. (C) Long-horizon planning with trajectory-level optimization and step-wise rewards.

strong open-source and closed-source models, our method achieves near state-of-the-art performance across metrics.

4 Analysis

Ablation Study. Figure 3 illustrates how each stage of FailAlign contributes to the overall performance of the LLM. We utilize the Raw-HTML configuration as a baseline, which reflects the capabilities of a model trained on standard uncompressed HTML sequences without any error-correction mechanisms. It clearly shows that our specialized variants yield consistent improvements over this baseline across all four evaluation metrics. We observe that FailAlign-RL achieves the most substantial gains, particularly in terms of SR and EAcc. This trend indicates that learning how to recover from complete failures in RL stage is crucial for the model. The steady outward expansion of the performance frontier from FailAlign-SFT through to FailAlign-RL confirms that error-driven refinement provides cumulative benefits throughout the training process. These results suggest that learning specifically from the model’s past errors can significantly enhance its reasoning abilities and



Figure 7: Relative performance improvements across training stages. The chart displays the accuracy delta for 3 stages across Domain, Task, and Website evaluation splits.

performance in the environment.

Round-wise Analysis. As illustrated in Figure 4, continuous alignment training yields consistent performance gains across all metrics. Both FailAlign-DPO and FailAlign-RL show improvements over the FailAlign-SFT, validating the efficacy of FailAlign. Notably, FailAlign-RL demonstrates a significant advantage in scenarios requiring multi-turn reasoning. By effectively penalizing wrong states in tasks, we can ensure robustness over long horizon trajectory during extended interactions.

Dataset Composition and Difficulty. As presented in Table 2, the $\mathcal{D}_{\text{warm}}$ comprises the largest volume of instances but fewest input tokens, serving as a valuable warmup for establishing significant base capabilities. In contrast, the $\mathcal{D}_{\text{pref}}$ and the $\mathcal{D}_{\text{tra}}$ are derived mainly from error mining. We further evaluate the complexity of these candidates using embedding cosine distance in Figure 5. Specifically, we observe that the distance between DPO positive and negative pairs concentrates around 0.19, providing challenging samples that facilitate robust preference learning. Furthermore, the RL candidates’ difficulty score center near 0.23 combined with tasks with approximately 11 steps, a composition that enables the model to learn stronger long-horizon planning capabilities.

Each Stage Training Signal. Figure 6 provides a unified walkthrough of our data formats under three objectives. Given a natural-language task and a bracket-serialized DOM snapshot, the dataset defines a structured action space. For SFT, supervision is a k-way option table that annotates a single gold action aligned with the target element, while keeping the remaining plausible distractors as negative choices. For DPO, supervision is expressed as a chosen versus rejected pair constructed from two competing options, explicitly encoding prefer-

ence learning over closely related actions. For RL, signal comes from trajectory rollouts composed of multiple steps, where each executed step receives a reward signal and the final outcome indicates whether the long-horizon plan succeeds, enabling optimization beyond single-step correctness.

Error Analysis The analytical results presented in Figure 7 illustrate the progressive rectification and systematic elimination of prediction errors throughout the FailAlign training pipeline. We observe consistent positive accuracy deltas across all primary evaluation splits, including the Domain, Task, and Website categories, which strongly indicates that the subsequent DPO and RL stages successfully recover various failure instances that were originally mispredicted during the initial SFT phase. Specifically, the incremental improvements from SFT to DPO and from DPO to RL demonstrate a continuous refinement of the agent’s policy as it encounters more sophisticated feedback. Furthermore, the substantial cumulative performance gain observed when comparing the initial SFT baseline directly to the final RL stage confirms that our framework effectively corrects initial errors and refines the agent decision-making process over the entire optimization trajectory. These multi-stage improvements highlight the robustness of the iterative optimization approach in enhancing the overall reliability of the model across increasingly complex and diverse web environments.

5 Related Work

Code Large Language Models. Code large language models have moved from code synthesis toward repository-level software engineering, where long-context understanding, tool use, and iterative verification become central. Proprietary systems such as the GPT series (OpenAI et al., 2024; OpenAI, 2025b,c), Gemini (Gemini Team et al., 2025; Comanici et al., 2025), Claude (Anthropic, 2025a,b,c), and Grok (xAI, 2025) increasingly couple strong base models with agent-style training and reinforcement learning on engineering tasks, setting the strongest results on repository repair benchmarks such as SWE-Bench. Recent open-weight releases narrow the gap by combining mixture-of-experts scaling with practical deployment and long-context training. Qwen3-Coder (Qwen, 2025) strengthens agentic coding under long context, while the DeepSeek series (DeepSeek-AI et al., 2025a) has rapidly it-

erated through DeepSeek-V3.1 (DeepSeek, 2025) and DeepSeek-V3.2 (DeepSeek-AI et al., 2025b) to improve tool use and multi-step coding workflows. The GLM-4 series advances to GLM-4.7 (Z.ai, 2025) with stronger programming capability and more stable multi-step execution, and Kimi-K2 (Team et al., 2026) further emphasizes tool calling in long-horizon tasks. MiniMax-M2 (MiniMax, 2025) targets efficient agentic coding with long-context support, enabling end-to-end development workflows under constrained cost and latency.

GUI-Agentic Systems. GUI agents automate web navigation and task execution by analyzing interface content to synthesize structured commands, such as JSON or executable code. The field has witnessed a significant paradigm shift, evolving from early text-centric approaches toward systems with richer multimodal perception and interaction capabilities. Initial efforts, exemplified by RUSS (Xu et al., 2021) and WebGPT (Nakano et al., 2022), relied primarily on textual or structured representations of interfaces. Subsequent work has expanded this direction through multimodal integration and broader environments, benchmarks, and infrastructure, as reflected in WebShop (Yao et al., 2022), WebGUM (Furuta et al., 2024), Agent-X (Ashraf et al., 2026), and Explorer (Chaimalas et al., 2025). Recent systems such as WebEvolver (Fang et al., 2025) further extend the autonomy of web agents. Overall, progress has centered on interface perception and multi-step interaction.

6 Conclusion

We present FailAlign, a failure-aligned learning framework for long-horizon HTML-based action code generation that couples (i) action-grounded context compression of DOM states, (ii) step-level error mining to build a unified failure index, (iii) hard-negative preference construction for DPO to sharpen confusable local decisions, and (iv) trajectory-level GRPO to propagate dense corrective signals and mitigate error cascades. On Mind2Web, FailAlign consistently improves element grounding and structured action correctness, yielding clear gains in EAcc, Operation F1, SSR, and the stricter long-horizon SR across Task/Domain/Website splits, indicating more robust end-to-end behavior under multi-step web interactions.

7 Limitations

We strictly evaluate our method on the Mind2Web, and while it spans many websites and domains, the generality of the proposed failure-aligned data engine and its training signals remains to be confirmed under other interactive settings, interface dynamics, and task distributions. In addition, specific components of the pipeline may depend on dataset-specific UI/DOM characteristics and could require adaptation when the environment exposes different observability or action spaces. Future work will therefore extend evaluation to WebArena and Mind2Web-derived variants, testing generalization, robustness, and scalability under broader web-agent environments.

Ethic Statement

All data used in this work is derived from publicly available, open-source resources, and we follow the original dataset terms and licenses when constructing training signals and reporting results. We do not collect new private user information, and our processing focuses on task-relevant DOM/visible context; any released artifacts should be used in compliance with website terms and applicable policies. As with other web automation and code-generation systems, the approach could be misused for unintended automation; we encourage responsible deployment with appropriate access controls, logging, and safeguards.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (62525601), CCF-Huawei Populus Grove Fund, the Fundamental Research Funds for the Central Universities (Grant No. GW2025-19), and the State Key Laboratory of Complex & Critical Software Environment (Grant No. SKLCCSE-2025ZX-26).

References

- Anthropic. 2025a. Claude 3.7 sonnet and claude code. <https://www.anthropic.com/news/claude-3-7-sonnet>.
- Anthropic. 2025b. Introducing claude 4. <https://www.anthropic.com/news/claude-4>.
- Anthropic. 2025c. Introducing claude sonnet 4.5. <https://www.anthropic.com/news/claude-sonnet-4-5>.
- Tajamul Ashraf, Amal Saqib, Hanan Gani, Muhra AlMahri, Yuhao Li, Noor Ahsan, Umair Nawaz, Jean Lahoud, Hisham Cholakkal, Mubarak Shah, Philip Torr, Fahad Shahbaz Khan, Rao Muhammad Anwer, and Salman Khan. 2026. *Agent-X: Evaluating deep multimodal reasoning in vision-centric agentic tasks*. In *International Conference on Learning Representations*, volume 2026, pages 49814–49874.
- Iason Chaimalas, Arnas Vyšniauskas, and Gabriel Brostow. 2025. *Explorer: Robust collection of interactive gui elements*. *Preprint*, arXiv:2504.09352.
- Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, Luke Marris, Sam Petulla, Colin Gaffney, Asaf Aharoni, Nathan Lintz, Tiago Cardal Pais, Henrik Jacobsson, Idan Szpektor, Nan-Jiang Jiang, and 3416 others. 2025. *Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities*. *Preprint*, arXiv:2507.06261.
- DeepSeek. 2025. Deepseek-v3.1 release. <https://www.deepseek.com/news/deepseek-v3-1/>.
- DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, and 181 others. 2025a. *Deepseek-v3 technical report*. *Preprint*, arXiv:2412.19437.
- DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenhao Xu, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, and 245 others. 2025b. *Deepseek-v3.2: Pushing the frontier of open large language models*. *Preprint*, arXiv:2512.02556.
- Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Sam Stevens, Boshi Wang, Huan Sun, and Yu Su. 2023. *Mind2web: Towards a generalist agent for the web*. In *Advances in Neural Information Processing Systems*, volume 36, pages 28091–28114. Curran Associates, Inc.
- Tianqing Fang, Hongming Zhang, Zhisong Zhang, Kaixin Ma, Wenhao Yu, Haitao Mi, and Dong Yu. 2025. *Webevolver: Enhancing web agent self-improvement with co-evolving world model*. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 8959–8975. Association for Computational Linguistics.
- Hiroki Furuta, Kuang-Huei Lee, Ofir Nachum, Yutaka Matsuo, Aleksandra Faust, Shixiang Gu, and Izzeddin Gur. 2024. *Multimodal web navigation with instruction-finetuned foundation models*. In *International Conference on Learning Representations*, volume 2024, pages 29528–29558.

- Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, and 1332 others. 2025. *Gemini: A family of highly capable multimodal models*. *Preprint*, arXiv:2312.11805.
- MiniMax. 2025. Minimax-m2. <https://huggingface.co/MiniMaxAI/MiniMax-M2>.
- Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2022. *Webgpt: Browser-assisted question-answering with human feedback*. *Preprint*, arXiv:2112.09332.
- OpenAI. 2024. Hello gpt-4o. <https://openai.com/index/hello-gpt-4o/>.
- OpenAI. 2025a. Gpt-5.1: A smarter, more conversational chatgpt. <https://openai.com/index/gpt-5-1/>.
- OpenAI. 2025b. Introducing gpt-5.
- OpenAI. 2025c. Introducing upgrades to codex. <https://openai.com/index/introducing-upgrades-to-codex/>.
- OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. *Gpt-4 technical report*. *Preprint*, arXiv:2303.08774.
- Qwen. 2025. *Qwen3-coder: Agentic coding in the world*.
- Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. 2023. *Direct preference optimization: Your language model is secretly a reward model*. In *Advances in Neural Information Processing Systems*, volume 36, pages 53728–53741. Curran Associates, Inc.
- Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. *Deepseekmath: Pushing the limits of mathematical reasoning in open language models*. *Preprint*, arXiv:2402.03300.
- Kimi Team, Yifan Bai, Yiping Bao, Y. Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, Ruijue Chen, Yanru Chen, Yuankun Chen, Yutian Chen, Zhuofu Chen, Jialei Cui, Hao Ding, Mengnan Dong, Angang Du, and 181 others. 2026. *Kimi k2: Open agentic intelligence*. *Preprint*, arXiv:2507.20534.
- xAI. 2025. Grok 4. <https://x.ai/news/grok-4>.
- Nancy Xu, Sam Masling, Michael Du, Giovanni Campagna, Larry Heck, James Landay, and Monica Lam. 2021. *Grounding open-domain instructions to automate web support tasks*. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1022–1032, Online. Association for Computational Linguistics.
- An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. *Qwen3 technical report*. *Preprint*, arXiv:2505.09388.
- Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. 2022. *Webshop: Towards scalable real-world web interaction with grounded language agents*. In *Advances in Neural Information Processing Systems*, volume 35, pages 20744–20757.
- Z.ai. 2025. GLM-4.7. <https://huggingface.co/zai-org/GLM-4.7>.
- Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. 2025. *Qwen3 embedding: Advancing text embedding and reranking through foundation models*. *Preprint*, arXiv:2506.05176.

A Example of Evaluation Prompt

Evaluation Prompt

```
(html (body (div id=0 (div button my store open store locator (div (small Open until 8:00pm ) (span San Diego ) ) ) (div button delivery zip code 43202 open (div (small Deliver to ) (span 43202 ) ) ) ) (div (div (button id=1 show sorting options modal (span Sort ) ) (button show filter modal color ) (button filter products by in stock ) (button show filter modal material ) (button filter products by top seller ) (button filter products by last chance ) (button filter products by new products ) (button id=2 show more filters modal (span All filters ) ) ) (div (div id=3 (a frnuft - 20-piece flatware set (img frnuft 20-piece flatware set, stainless ) ) (div (div (div (span FRNUFT ) (span 20-piece flatware set ) ) (span (span $ ) (span 12 ) (span (span . ) 99 ) ) (span review: 4.7 out of 5 (span (407) ) ) ) ) (div (button button add to bag, frnuft, 20-piece ) (button button save to igift, frnuft, 20-piece ) (button button save to favorites, frnuft, 20-piece ) ) ) ) (div (input checkbox select for product comparison, 20-piece checkboxname on ) (span id=4 Available for delivery ) (span In stock at San Diego, CA ) ) ) ) ) ) )
```

Based on the HTML webpage above, try to complete the following task:

Task: Find the cheapest set of stainless steel spoons, which are available in stock in San Diego, California.

Previous actions:

- [span] Columbus -> CLICK
- [searchbox] Search by ZIP code or city, state -> TYPE: san diego
- [button] Submit -> CLICK
- [span] Select store -> CLICK
- [link] Flatware -> CLICK

What should be the next action? Please select from the following choices (If the correct action is not in the page above, please select A. None of the above):

- A. None of the above
- B. (div id=0 (div button my store open store locator (div
- C. (button id=1 show sorting options modal (span Sort))
- D. (button id=2 show more filters modal (span All filters)
- E. (div id=3 (a frnuft - 20-piece flatware set (img frnuft)
- F. (span id=4 Available for delivery)

Answer Example:

Answer: C.

Action: SELECT

Value: Pickup

Figure 8: Example evaluation prompt with webpage context and candidate actions.

B DOM Representation and Context Compression Details

B.1 Raw HTML

```
<!-- Raw HTML (excerpt; noisy & redundant).
      Many nodes are irrelevant to action
      selection. -->
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8"/>
  <title>Flatware | My Store</title>
  <script src="/static/vendor/react.bundle.
    js"></script>
  <script>>window.__ANALYTICS__ = {
    sessionId:"...", ab:"exp_42" };/*
    dozens of tracking beacons, feature
    flags, personalization, etc. */</script>

  <style>/* large CSS bundle, icons,
    responsive rules, animations ... */</
    style>
</head>
<body>
  <div id="cookie-banner" class="overlay">
    <p>We use cookies for performance and
    personalization...</p>
    <button aria-label="Accept all">Accept</
    button>
    <button aria-label="Manage preferences">
    Manage</button>
  </div>
  <header class="site-header">
    <nav>
      <a href="/">Home</a><a href="/shop">
      Shop</a><a href="/deals">Deals</a>
      <form role="search"><input
      placeholder="Search products"/></form>
      <svg><!-- icons --></svg>
    </nav>
  </header>
  <main id="content">
    <!-- Store context bar (relevant
    context but surrounded by wrappers) -->
    <section class="store-bar">
      <!-- (kept) root candidate for choice
      B after compression -->
      <div class="store-block">
        <div class="left">
          <div class="spacer"></div>
          <div class="store-entry">
            <button aria-label="my store"
            data-testid="store-locator">
              my store
              <span class="icon"></span>
            </button>
            <div class="meta">
              <small>Open until 8:00pm</
              small>
              <span>San Diego</span>
            </div>
          </div>
          <div class="right">
            <button aria-label="delivery zip
            code" data-testid="delivery-zip">
              delivery zip code 43202
            </button>
            <div class="meta">
              <small>Deliver to</small><span>
              43202</span>
            </div>
          </div>
        </div>
        <div class="toolbar">
          <!-- Toolbar (many sibling buttons;
          only a few become candidate roots) -->
          <section class="toolbar">
            <div class="toolbar-inner">
              <!-- (kept) choice C after
              compression -->
              <button id="sortBtn" aria-label="
              show sorting options modal">
                <span>Sort</span><span class="
                icon"></span>
              </button>
              <!-- mostly irrelevant filters for
              this step -->
              <button aria-label="show filter
              modal color">color</button>
              <button aria-label="filter products
              by in stock">in stock</button>
              <button aria-label="show filter
              modal material">material</button>
              <button aria-label="filter products
              by top seller">top seller</button>
              <button aria-label="filter products
              by last chance">last chance</button>
              <button aria-label="filter products
              by new products">new</button>
              <!-- (kept) choice D after
              compression -->
              <button id="allFiltersBtn" aria-
              label="show more filters modal">
                <span>All filters</span>
              </button>
            </div>
          </section>
          <!-- Product grid (deep nesting,
          repeated wrappers, ads, rec widgets,
          etc.) -->
          <section class="grid">
            <div class="grid-row">
              <div class="sponsored-slot">
                <div class="ad">ADVERTISEMENT</
                div>
                <script>/* ad network injection */
                </script>
              </div>
              <!-- (kept) choice E after
              compression -->
              <div class="product-tile" data-sku=
              "FRNUFT-20">
                <a href="/p/frnuft-20-piece-
                flatware-set" class="product-link">
                  frnuft - 20-piece flatware set
                  
                </a>
                <div class="details">
                  <div class="names"><span>FRNUFT
                  </span><span>20-piece flatware set</
                  span></div>
                  <div class="price"><span>$</
                  span><span>12</span><span>.99</span></
                  div>
                  <div class="reviews">review:
                  4.7 out of 5 <span>(407)</span></div>
                </div>
              </div>
            </div>
          </section>
        </div>
      </div>
    </section>
  </main>
</body>
</html>
```

```

        <div class="cta">
          <button aria-label="add to
bag">add to bag</button>
          <button aria-label="save to
igift">save to igift</button>
          <button aria-label="save to
favorites">save to favorites</button>
        </div>
        <div class="availability">
          <input type="checkbox" aria-
label="select for product comparison"/>
          <!-- (kept) choice F after
compression -->
          <span class="delivery">
Available for delivery</span>
          <span class="store-stock">In
stock at San Diego, CA</span>
        </div>
        <div>
        <aside class="recommendations">
          <h3>Recommended for you</h3>
          <div class="carousel">...</div>
        </aside>
        </div>
      </section>
    </main>
    <footer>
      <div class="legal">Terms Privacy
Accessibility ...</div>
      <script>/* more telemetry & hydration
scripts */</script>
    </footer>
  </body>
</html>

```

B.2 Compressed DOM

```

(html
  (body
    (div id=0
      (div button my store open store
locator
        (div (small Open until 8:00pm) (
span San Diego)))
      (div button delivery zip code 43202
open
        (div (small Deliver to) (span 43202)
)))
    (div
      (div
        (button id=1 show sorting options
modal (span Sort))
        (button show filter modal color)
        (button filter products by in stock)

        (button show filter modal material)
        (button filter products by top
seller)
        (button filter products by last
chance)
        (button filter products by new
products)
        (button id=2 show more filters
modal (span All filters))))
      (div
        (div id=3
          (a frnuft - 20-piece flatware set

```

```

        (img frnuft 20-piece flatware
set, stainless))
      (div
        (div
          (div (span FRNUFT) (span 20-
piece flatware set))
          (span (span $) (span 12) (span
(span .) 99))
          (span review: 4.7 out of 5 (
span (407))))
          (div
            (button button add to bag,
frnuft, 20-piece)
            (button button save to igift,
frnuft, 20-piece)
            (button button save to
favorites, frnuft, 20-piece))))
          (div
            (input checkbox select for product
comparison, 20-piece checkboxname on)
            (span id=4 Available for delivery)
            (span In stock at San Diego, CA))))))

# Candidate roots (A--F) used by the policy
for action selection:
# A: None of the above
# B: subtree rooted at (div id=0 ...)
# C: subtree rooted at (button id=1 ...)
# D: subtree rooted at (button id=2 ...)
# E: subtree rooted at (div id=3 ...)
# F: subtree rooted at (span id=4 ...)

```

Raw HTML vs. compressed DOM. A straight-forward way to represent webpage state is to serialize the raw page HTML (Raw HTML), which maximizes fidelity but introduces substantial boilerplate and weakly action-related tokens (e.g., layout wrappers, UI chrome, and auxiliary modules). To understand whether providing HTML context is beneficial in principle, we include a Raw-HTML variant trained with the same supervision format. As shown in Table 1, adding raw HTML context already improves over the base model (Qwen3-32B), indicating that webpage HTML does contain useful grounding cues for action prediction. However, this gain comes with a clear efficiency drawback: the raw serialization is unnecessarily long and noisy, making the prompt budget dominated by irrelevant content and consequently limiting the model’s ability to focus on discriminative evidence.

In contrast, our compressed DOM representation preserves the actionable structure and local evidence while aggressively removing non-essential tokens, and it explicitly aligns candidate subtrees to the decision options. This design improves the information-to-noise ratio under a fixed context budget, which translates into consistently stronger performance across splits and metrics in Table 1. In particular, our FailAlign-based training with com-

pressed inputs (SFT/DPO/RL) yields further gains beyond the Raw-HTML-trained variant, supporting the claim that *how* we represent HTML context matters: naively providing more HTML is helpful but inefficient, whereas targeted compression retains the useful grounding signal and enables more effective learning.