BIND BEFORE YOU BUY CANDIDATE-CONSISTENT CONSTRAINT LEARNING FOR INTERACTIVE SHOPPING AGENTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Interactive shopping agents must recover user requirements while navigating a catalog, then buy one product configuration that satisfies every category, attribute, option, and price constraint. Supervised fine-tuning teaches successful interaction patterns, while online reinforcement learning improves the policy from terminal outcomes. The terminal reward does not reveal which earlier clarification or product check made a trajectory useful. A second ambiguity appears when the policy tracks user needs and browser progress separately, since facts from incompatible products may appear jointly complete. We introduce ShopBind, a candidate-consistent constraint learning framework that uses one visible interaction state in both training stages. ShopBind supervises a compact binding trace that assigns every visible requirement to support, contradiction, or uncertainty for one observed product configuration. It then uses first-time contradiction-free coverage expansions to resolve terminal ties and redistribute a bounded share of trajectory advantage. The benchmark reward, action space, rollout budget, and number of policy calls remain unchanged. We evaluate Qwen3-8B in all four ShopSimulator scenarios under shared data, optimization, decoding, and selection budgets. On the completed Single-Turn evaluation over 1,459 test tasks, ShopBind reaches 65.76 R_{loose}, 50.00 R_{strict}, and 45.92 R_{succ}.

1 INTRODUCTION

Shopping agents built on large language models combine dialogue, catalog search, and structured environment actions in one policy (Wu et al., 2024; Xu et al., 2024; Yao et al., 2022; Chen et al., 2024). A useful policy must do more than retrieve a plausible item. It must recover missing requirements, interpret long-term preferences, inspect similar products, select compatible options, and purchase one configuration that satisfies the complete request. The output is therefore an interaction trajectory ending in one concrete purchase rather than a ranked list.

Current post-training recipes improve shopping policies through successful demonstrations and terminal reinforcement learning. This recipe is effective because the final evaluator directly measures purchase quality. It is also incomplete because many trajectories share the same terminal outcome even though their intermediate decisions differ sharply. A failed trajectory may clarify the decisive option and inspect a compatible product, while another may repeat searches or purchase before checking the product page. Terminal-only optimization assigns the same group-relative signal to both trajectories.

The ambiguity becomes more serious when user requirements and browsing progress are tracked as separate lists. A requested color may be confirmed for one product while the required size exists only for another. Their union appears complete, yet no purchasable configuration satisfies the request. The relevant learning object is therefore not an isolated requirement or an isolated page observation. It is the relation between every visible requirement and one concrete product with one compatible option tuple.

This observation motivates one principle. Intermediate progress should count only when one product identity and one compatible option tuple support it. We call this requirement candidate consistency. ShopBind represents visible needs and observed configurations through signed relations. SFT teaches

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.

a compact projection of this state before the next environment action. During GRPO, the state is recomputed from visible interaction prefixes. First-time increases in contradiction-free coverage resolve terminal ties and move a bounded amount of credit toward the steps that produced them. Distinct terminal outcomes always retain priority.

We study this problem in four interactive shopping settings that vary dialogue and personalization (Wang et al., 2026). The empirical study first asks whether terminal ties hide valid product specific progress and whether identity free tracking creates false progress. The method then follows directly from the answer. Experiments compare complete systems, isolate the training design with one Qwen3-8B backbone, test every claimed component, locate the redistributed credit, and record the operating cost. The completed Single-Turn evaluation gives 45.92 percent full success over all 1,459 test tasks.

Our Contributions: (1) **In-depth Investigation.** We show that terminal-equal shopping trajectories retain useful variation in product specific progress and that identity free tracking can combine requirements across incompatible configurations. (2) **Novel Method.** We introduce ShopBind, which connects binding-trace supervision with outcome-lexicographic frontier credit through one candidate-consistent state while preserving the terminal objective. (3) **State-of-the-art Performance.** We establish a shared four-scenario Qwen3-8B protocol and reach 45.92 percent full success on the completed Single-Turn evaluation. The paired study holds data, optimization, rollout, decoding, and model selection budgets constant.

2 Related Works

Interactive shopping and web agents. WebShop and ChatShop introduced grounded product search, purchase, and interactive information seeking (Yao et al., 2022; Chen et al., 2024). DeepShop, WebMall, ShoppingBench, and ShopSimulator add deep product research, cross-store browsing, fine-grained intents, personalization, and clarification (Lyu et al., 2025; Peeters et al., 2025; Wang et al., 2025a; 2026). General web benchmarks study action prediction, functional websites, visual grounding, dialogue, and computer-use tasks (Deng et al., 2023; Zhou et al., 2024a; de Chezelles et al., 2024; Koh et al., 2024; Zheng et al., 2024; He et al., 2024b; Lai et al., 2024; Lù et al., 2024; Tian et al., 2025; Pan et al., 2024; Xie et al., 2024). Our setting focuses on whether progress toward one shopping purchase remains coherent across a trajectory.

Commerce knowledge and personalization. Shopping MMLU, ChineseEcomQA, and ECKGBench evaluate commerce knowledge and reasoning, while ESCI and Amazon-M2 provide large product-query and shopping-session collections (Jin et al., 2024b; Chen et al., 2025; Liu et al., 2025; Reddy et al., 2022; Jin et al., 2024a). EshopInstruct, e-Llama, EcomGPT, and eCeLLM specialize language models for commerce tasks (Zhang et al., 2024; Herold et al., 2025; Li et al., 2024; Peng et al., 2024). Recommendation agents and user simulators further model personas, preferences, and interactive behavior (Wang et al., 2025c; Sun et al., 2025; Park et al., 2023; Sun et al., 2024; Wang et al., 2023; Nie et al., 2024; Gong et al., 2025; Yi et al., 2025; Tsai et al., 2024; Peng et al., 2026). These methods improve domain or user modeling. They do not define when several visible requirements are supported by the same product configuration.

Reasoning and interaction data. ReAct, Reflexion, WebGPT, WebDreamer, and commitment planning organize action reasoning, feedback, and revision in interactive tasks (Yao et al., 2023; Shinn et al., 2023; Nakano et al., 2021; Gu et al., 2024; Liu et al., 2026b). Learn-by-Interact, Explorer, Proposer-Agent-Evaluator, and OpenWebVoyager acquire trajectories or skills through environment interaction (Su et al., 2025; Pahuja et al., 2025; Zhou et al., 2024b; He et al., 2024a). ShopBind is narrower than a general plan or memory. Its trace records which visible need is supported, contradicted, or unresolved for the product under inspection before the next action.

Post-training and credit assignment. WebRL, DigiRL, WebAgent-R1, Shop-R1, ChatShopBuddy, and OpenWebRL improve interactive policies through supervised data and online reinforcement learning (Qi et al., 2025; Bai et al., 2024; Wei et al., 2025; Zhang et al., 2025; Cheng et al., 2026; Yang et al., 2026). PPO, DPO, GRPO, and DAPO provide common optimization foundations, and ROLL supports distributed agent training (Schulman et al., 2017; Rafailov et al., 2023; Shao et al., 2024; Yu et al., 2025; Wang et al., 2025b). RMSWeb studies action-semantic rewards for compact web agents

(Liu et al., 2026a). ShopBind keeps the released terminal reward, uses candidate-consistent progress only inside terminal ties, and redistributes a zero-mean share of advantage across time.

3 Preliminaries and Problem Formulation

3.1 Interactive Shopping

Following the ShopSimulator interaction protocol (Wang et al., 2026), let an episode contain a user instruction u_0, an optional static profile p, browser observations o_t, and, in multi-turn scenarios, shopper utterances u_t. At step t, a policy π_θ produces one environment action

$$a_t \sim \pi_\theta(\cdot \mid u_{\le t}, p, o_{\le t}, a_{<t}), \tag{1}$$

where the feasible actions search the catalog, click a displayed element or product option, ask the shopper, or purchase the current configuration. The environment returns the next browser observation and, when applicable, a shopper reply. A single-turn episode allows at most 30 actions, whereas a multi-turn episode allows at most 40 (Wang et al., 2026).

The decision process may use only information visible in the interaction prefix. In particular, the target product, hidden reward fields, private shopper state, and final evaluator output are unavailable to the policy and to all training annotations. A product configuration x denotes one displayed product identity together with its currently selected options. Facts from different product identities or incompatible option tuples cannot be merged into one configuration.

3.2 Terminal Objectives

Let U_{att}, U_{opt}, and U_{price} denote the required attributes, options, and price bound. Let Y_{att}, Y_{opt}, and Y_{price} describe the purchased configuration. ShopSimulator uses three terminal metrics (Wang et al., 2026). The loose score averages partial constraint satisfaction after category matching. Its per-episode value is

$$R_{\text{loose}} = R_{\text{cat}} \frac{|U_{\text{att}} \cap Y_{\text{att}}| + |U_{\text{opt}} \cap Y_{\text{opt}}| + \mathbb{I}[Y_{\text{price}} \le U_{\text{price}}]}{|U_{\text{att}}| + |U_{\text{opt}}| + 1}. \tag{2}$$

The strict score multiplies the category, attribute, option, and price terms so that a weak dimension suppresses the total. Its per-episode value is

$$R_{\text{strict}} = R_{\text{cat}} \frac{|U_{\text{att}} \cap Y_{\text{att}}|}{|U_{\text{att}}|} \frac{|U_{\text{opt}} \cap Y_{\text{opt}}|}{|U_{\text{opt}}|} \mathbb{I}[Y_{\text{price}} \le U_{\text{price}}]. \tag{3}$$

Full success R_{succ} is binary and requires the purchase to satisfy every required dimension. We report dataset means as percentages, including $\text{SR} = |\mathcal{D}|^{-1} \sum_{i \in \mathcal{D}} R^{(i)}_{\text{succ}}$. Missing episodes, invalid outputs, timeouts, and trajectories without a purchase contribute zero.

Unlike static product ranking or one-shot retrieval, the policy does not receive a complete query and return a static list. Each environment action may reveal a new shopper requirement, a new product attribute, or a different selected option, thereby changing the visible information used by the next decision. The output is therefore one purchased configuration reached through an ordered interaction, rather than a score assigned independently to catalog items.

We study post-training under the same information and interaction budget. Both the baseline and ShopBind use the same Qwen3-8B backbone, the same ordered demonstrations, the same rollout prompts, and the same environment and shopper randomness. The baseline performs action-only SFT followed by GRPO on terminal R_{strict}. ShopBind may change the supervised target and training-time credit, but it cannot add a verifier, reranker, tool call, policy call, rollout, or hard purchase gate.

4 Empirical Study

4.1 Where the Terminal Signal Loses Resolution

We begin from a recurring failure in interactive shopping. A policy can discover a useful requirement, inspect a relevant product, or recover from a poor candidate and still receive the same terminal

outcome as a trajectory that made no such progress. Error analysis in the environment reinforces this concern. Search actions omit known attributes, product clicks violate explicit needs, and many purchases occur before details are confirmed (Wang et al., 2026). Personalized tasks add another tension. Useful profile fields may be ignored, while unrelated fields may be treated as active requirements.

These failures suggest two connected questions. First, how much training resolution disappears when several rollouts receive equal terminal advantages. Second, whether a progress score remains valid when the policy moves between similar products and option tuples. A score based only on the number of known needs and observed attributes can answer the first question while failing the second. It may reward an impossible union assembled from different products.

4.2 A Prefix Level Probe

We examine visible rollout prefixes from the final 20 optimizer updates. For every GRPO group, the probe records whether terminal advantages are tied and whether the tied trajectories still differ in contradiction free coverage. Coverage is computed separately for each product identifier and selected option tuple. A second readout compares this value with a separable score that tracks user needs and product facts independently. Their disagreement counts a progress event that disappears once product identity is restored. The probe uses only the instruction, profile, shopper replies, browser observations, and earlier actions.

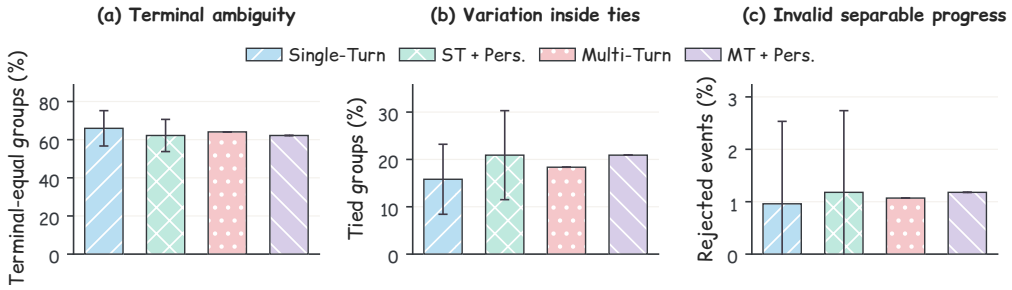

Figure 1: **Progress inside terminal ties.** Prefix statistics across four interaction settings.

4.3 Observation and Design Requirement

Figure 1 shows frequent terminal ties in every setting. Their mean share ranges from 62.19 to 65.94 percent. Candidate-consistent coverage still varies inside 15.81 to 20.90 percent of tied groups. Identity checks reject 0.96 to 1.18 percent of separable progress events. The first two patterns show that terminal-equal rollouts often remain distinguishable through visible interaction quality. The third pattern is smaller but changes the meaning of the signal. Even an occasional cross-product union can reward progress that no single product supports.

The empirical study yields one method requirement. Intermediate credit should distinguish terminal-equal trajectories only when visible requirements become jointly supported by one observed configuration. The state used to define that progress should also be learnable by the policy, so the same product and requirement relation guides supervised initialization and reinforcement learning. This requirement motivates the candidate-consistent state in the next section.

5 Methodology

5.1 Overview

ShopBind uses one candidate-consistent constraint state across supervised initialization and online RL. Each visible prefix provides user needs supported by the interaction and product configurations that have already been observed. SFT teaches a compact binding trace followed by one environment action. RL then detects the first improvements in the same state, resolves terminal ties, and moves a bounded amount of credit toward the responsible trace and action tokens. Figure 2 shows the complete flow.

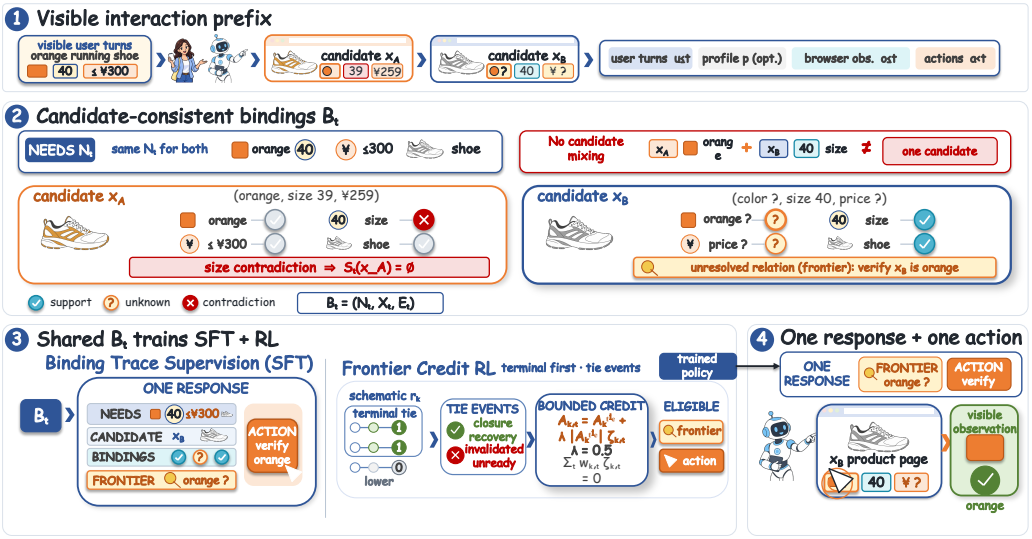

Figure 2: **ShopBind overview.** One candidate state connects binding-trace supervision with frontier credit.

5.2 Candidate Consistent Constraint State

At prefix t, let $\mathcal{N}_t$ contain the needs supported by visible user information and let $\mathcal{X}_t$ contain the observed product configurations. Each configuration identifies one displayed product and its selected compatible options. The relation $e_t(x, j) \in \{+1, 0, -1\}$ marks visible support, an unknown relation, or a contradiction. We write the visible relation graph as $\mathcal{B}_t = (\mathcal{N}_t, \mathcal{X}_t, \mathcal{E}_t)$. The valid support of one configuration is formally defined as

$$\mathcal{S}_t(x) = \begin{cases} \{j \in \mathcal{N}_t \mid e_t(x, j) = +1\}, & \nexists j \in \mathcal{N}_t \text{ with } e_t(x, j) = -1, \\ \emptyset, & \text{otherwise.} \end{cases} \tag{4}$$

Missing product text remains unknown and relations never transfer across product configurations. Let $M_t = \max\big(\{0\} \cup \{|\mathcal{S}_t(x)| \mid x \in \mathcal{X}_t\}\big)$ denote the best valid support at the current prefix. A frontier event is then defined as

$$d_t = \mathbb{I}\Big[M_{t+1} > \max_{s \leq t} M_s\Big]. \tag{5}$$

Repeated questions, revisits, merely displayed options, cross-product unions, and the purchase action itself do not increase the historical maximum. Thus d_t marks only the first visible expansion of valid support for one configuration.

5.3 Binding Trace Supervision

The binding frontier is one unresolved need and configuration relation that can improve coherent coverage after clarification or verification. The supervised response places a compact state projection before exactly one environment action.

`NEEDS` visible need, value, and origin
`CANDIDATE` product identifier and selected options
`BINDINGS` support, contradiction, or unknown
`FRONTIER` one unresolved relation
`ACTION` one environment action

A deterministic procedure constructs this trace from the visible transcript and links each known relation to a displayed span. Ambiguous mappings remain unknown. For target tokens $y_{1:L}$, the shared trace and action objective is

$$\mathcal{L}_{\text{SFT}} = -\sum_{\ell=1}^{L} \log \pi_\theta(y_\ell \mid y_{<\ell}, u_{\leq t}, p, o_{\leq t}, a_{<t}). \tag{6}$$

The policy thus learns the product-specific state that should precede each action without adding a second inference round.

5.4 Outcome Lexicographic Frontier Credit

Consider a GRPO group of K trajectories with terminal rewards r_k and ordinary group advantages A_k^G. A training-time scanner assigns each step a lifecycle score $v_{k,t} \in \{-1, 0, +1\}$. First coherent closures and valid recoveries receive positive values. Candidate invalidation and visibly unready purchases receive negative values. Repeated events and ready purchases receive zero. The trajectory quality is $q_k = \sum_t v_{k,t}$. Within each tie set $\mathcal{T}(a) = \{k \mid A_k^G = a\}$, we center and normalize q_k as $\eta_k \in [-1, 1]$, where $\sum_{k \in \mathcal{T}(a)} \eta_k = 0$. Tie resolution is formally defined as

$$A_k^{\text{tie}} = A_k^G + \rho_G \eta_k. \tag{7}$$

The bound ρ_G is at most half the smallest positive gap between distinct group advantages. A fully tied group uses the unit anchor and a tie set with no lifecycle variation receives no added signal. This bound preserves the ordering of distinct group advantages and lets visible candidate-consistent progress separate only terminal-equal trajectories.

We next center the lifecycle scores over eligible frontier and action steps as $\zeta_{k,t} \in [-1, 1]$. With masked response-token count $w_{k,t}$, the centering satisfies $\sum_t w_{k,t}\zeta_{k,t} = 0$. Using $\lambda = 0.5$, temporal credit is formally defined as

$$A_{k,t} = A_k^{\text{tie}} + \lambda |A_k^{\text{tie}}| \zeta_{k,t}. \tag{8}$$

All remaining tokens retain A_k^{tie}. Equations 7 and 8 preserve the mean inside each tie set and the masked-token-count weighted mean inside each trajectory before clipping. Distinct terminal outcomes therefore retain priority.

5.5 Information and Computational Boundaries

Both stages use visible prefixes only. Training annotations are recomputed from the environment transcript and are not copied from model-generated traces. During inference the model emits the trace and action in one response. ShopBind changes response length but adds no model invocation, tool, environment action, rollout, verifier, or hidden field access.

6 Experiments

The experiments address four questions. **Q1** measures full success across dialogue and personalization settings. **Q2** studies the post-training gain with one Qwen3-8B backbone. **Q3** tests the trace, tie rule, temporal redistribution, and candidate identity. **Q4** studies where frontier credit acts and whether the external interaction protocol changes.

6.1 Experimental Setup

We use the released ShopSimulator environment, Catalog Fine interface, evaluator, and four dialogue and profile scenarios (Wang et al., 2026). Evaluation contains 1,459 tasks in each nonpersonalized scenario and 1,343 tasks in each personalized scenario. Missing, invalid, timed-out, and no-purchase episodes receive zero. Greedy decoding uses one policy call per action.

Both paired arms start from the same Qwen3-8B revision (Yang et al., 2025) and consume the same ordered cohort of 6,000 successful GPT-5.5 trajectories. Each scenario contributes 1,500 trajectories. SFT uses 750 updates. Scenario specific GRPO uses 200 steps with 32 prompts and eight rollouts per step, learning rate 10^{-6}, and the released ROLL stack (Wang et al., 2025b; Shao et al., 2024). The terminal-only arm uses action-only SFT and terminal R_{strict}. ShopBind changes only the supervised target and training-time credit. Tables use point estimates because the ShopSimulator source reports no dispersion. Repeated-run vectors remain available for diagnostic plots and statistical analysis.

6.2 Main Results Across Interaction Settings

Table 1 gives two complementary readings of full success. The general-agent block locates the task difficulty across several model families (Wang et al., 2026). The Qwen3-8B block follows one

Table 1: **Full success across interaction settings.** ShopSimulator systems and ShopBind.

System	R_{succ}				Average
	Single-Turn	Single-Turn + Pers.	Multi-Turn	Multi-Turn + Pers.	
ShopSimulator agent results					
GPT-5	**40.78**	**36.41**	29.29	24.13	**32.65**
Claude-4-Sonnet	30.29	32.17	**38.25**	25.39	31.52
DeepSeek-V3.1 (671B-A37B)	31.86	35.69	31.62	**28.07**	31.81
Qwen3-235B-A22B	27.96	26.81	24.29	23.60	25.66
ShopSimulator Qwen3-8B training systems					
Baseline	14.13	17.24	6.48	13.63	12.87
SFT	32.47	30.98	29.27	30.49	30.80
SFT + RL w. R_{strict}	38.89	57.33	35.50	34.35	41.52
ShopBind	**45.92**	**67.70**	**41.92**	**40.56**	**49.02**

trainable backbone through supervised and reinforcement learning, then presents ShopBind in the same four interaction settings. ShopBind reaches a four-setting average of 49.02 percent and is higher than ShopSimulator SFT plus strict-reward RL in every setting. The differences range from 6.21 to 10.37 points under the same metric columns.

6.3 Completed Single-Turn Analysis

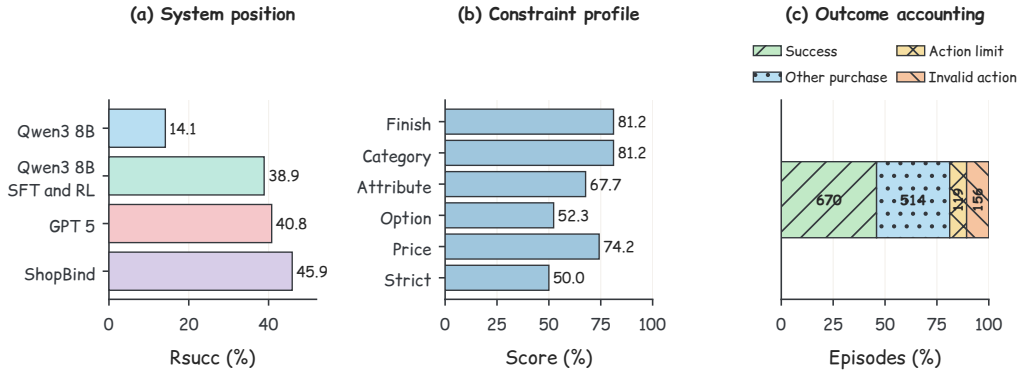

Figure 3: **Single-Turn evaluation.** System position, constraint scores, and outcomes over 1,459 tasks.

Figure 3 reports the completed evaluation at 45.92 on R_{succ} with 670 exact successes and 1,184 purchases over 1,459 tasks. Option satisfaction is the weakest purchased-product dimension at 52.30, followed by attribute satisfaction at 67.69. The outcome panel separates incorrect purchases from action-limit and malformed-action failures, so full success is not read as completion alone.

6.4 Cross Setting Improvement

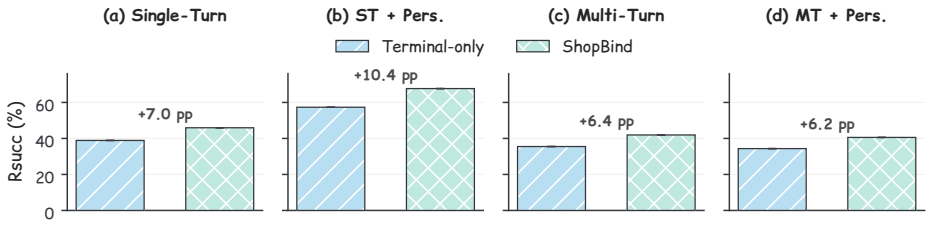

Figure 4: **Full success by interaction setting.** ShopBind and the closest Qwen3-8B training system.

Figure 4 focuses on the closest ShopSimulator Qwen3-8B training system and makes the setting-level differences visible at a glance. The repeated direction across dialogue and profile settings matters more than the largest cell because it tests whether the method is tied to one interaction mode.

6.5 Component Necessity

Table 2: **Core ablation.** Full success and average change from full ShopBind.

Variant	R_{succ}				Average	Δ vs. Full
	Single-Turn	Single-Turn + Pers.	Multi-Turn	Multi-Turn + Pers.		
Supervised initialization						
ShopSimulator SFT	32.47	30.98	29.27	30.49	30.80	−18.22
Binding trace SFT	**39.20**	**49.34**	**35.59**	**35.53**	**39.91**	−9.11
Full method and component removals						
Full ShopBind	**45.92**	**67.70**	**41.92**	**40.56**	**49.02**	0.00
w/o Binding Trace supervision	43.58	64.24	39.78	38.49	46.52	−2.50
w/o Outcome-Lexicographic Tie Credit	43.58	64.24	39.78	38.49	46.52	−2.50
w/o Frontier Redistribution	43.58	64.24	39.78	38.49	46.52	−2.50
Candidate consistency controls						
Separable need and product progress	**41.23**	**60.79**	**37.64**	**36.42**	**44.02**	−5.00
Cross product relation union	38.89	57.33	35.50	34.35	41.52	−7.51

Table 2 separates supervised initialization, component removals, and candidate consistency controls. The ShopSimulator SFT average is 30.80 and binding trace SFT reaches 39.91. Each neutral one-component removal is 46.52, below the full value of 49.02. Separable progress reaches 44.02 and cross-product union reaches 41.52. This hierarchy tests the learned state first, the credit rule second, and product identity last, so one aggregate removal cannot hide which design level is responsible.

6.6 Where Frontier Credit Acts

We inspect first coherent expansions in successful and failed trajectories, then summarize where redistributed credit appears across action types. Successful trajectories contain 1.12 to 1.28 first expansions on average, while other trajectories contain 0.86 to 1.09.

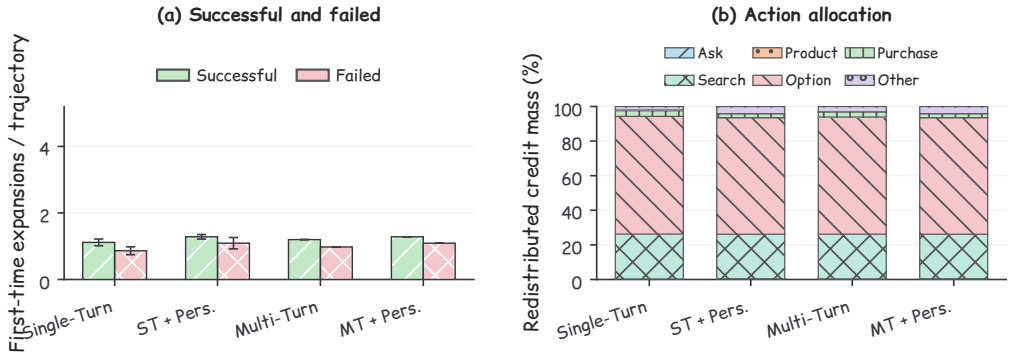

Figure 5: **Frontier credit behavior.** Events and credit destinations across four scenarios.

Figure 5 places most redistributed mass on option clicks and search actions. Shopper queries receive a larger share in the multi-turn settings, which matches their clarification action space. In Single-Turn and personalized Single-Turn, option clicks receive 68.03 ± 1.87 and 67.38 ± 3.19 percent of redistributed mass, while search receives 25.95 ± 2.05 and 25.73 ± 2.72 percent. Purchase receives only a small share. The analysis locates the training signal near requirement acquisition and product verification rather than treating the final purchase token as the sole decision point. This action-level diagnostic locates the redistributed signal, while the paired outcome and ablation tables establish the resulting task-level effect.

6.7 What Changes at Purchase

Appendix Table 6 shows that ShopBind raises R_{strict} and R_{succ} in every scenario even though it does not maximize purchase completion. The clearest change appears in option satisfaction. It changes from 47.26 to 52.30 in Single-Turn, from 63.12 to 69.85 with personalization, from 43.05 to 47.64 in Multi-Turn, and from 42.62 to 47.16 when dialogue and personalization are combined. Reading

the full evaluator profile prevents an increase in completed purchases from being mistaken for better constraint satisfaction.

6.8 Execution Boundary

Both policies retain one model call per environment action and the same 30 and 40 step limits. ShopBind emits its binding trace and action in one response, placing candidate consistency inside the existing policy call instead of adding a verifier round. The completed Single-Turn run averages 320.12 response tokens and 6.41 actions per episode. Its final 20 training updates average 49.09 response tokens per action, 2.63 seconds per action, and 5.76 actions per episode. The trace therefore changes what the policy represents while preserving the environment interaction budget. Together, the response, latency, and trajectory statistics characterize the operating profile under the reported hardware and decoding configuration. The evaluation average covers complete responses and episodes, whereas the final-update profile measures per-action training traffic. Keeping these boundaries separate makes the reported 320.12 and 49.09 token counts directly interpretable.

6.9 Interpretation

The diagnostic, paired comparison, and evaluator profile point to the same bottleneck. A shopping agent must connect requirement acquisition with product verification throughout the trajectory. A correct shopper reply can be applied to the wrong product, and a correct product check can become stale after an option change. Candidate identity makes these dependencies explicit. The main comparison measures the final task effect, the ablation identifies the contributing design level, and the diagnostic locates the added signal in the rollout. Together they connect representation, optimization, and outcome under one matched interface. ShopBind changes the state predicted by the policy and the credit assigned during training while keeping the external interaction interface intact.

This connection explains why completion and constraint satisfaction move differently. A policy can reach purchase quickly while carrying stale facts or facts pooled across products. Binding traces make that state error visible before the action, and frontier credit rewards the first candidate-consistent coverage expansion that resolves it. The evaluator profile then measures whether those local state improvements survive through the purchased configuration.

Support synthesis. The empirical probe identifies terminal ties that hide different candidate specific states. Binding trace supervision and frontier credit address this mismatch at the state and optimization levels. Held-out success then tests the method, component removals test the design, and rollout statistics locate the added signal. This chain follows one object, the active product configuration, from visible support through training credit to the final evaluator outcome.

Candidate-consistent learning turns that principle into an operational rule. Each need is keyed to the current product and option tuple, state transitions preserve uncertainty until the interface resolves it, and learning credit follows the first contradiction-free expansion of that state. The resulting policy improves final constraint satisfaction without changing the external action space, terminal reward, rollout budget, or number of policy calls.

7 Conclusion

Interactive shopping requires several user constraints to hold on one product configuration. Terminal-only RL cannot distinguish useful clarification and verification steps inside tied outcomes. Product-agnostic progress can instead combine facts from incompatible configurations. ShopBind uses one candidate-consistent state across SFT and RL. Binding-trace supervision teaches the state with the next action, while frontier credit separates terminal ties without changing the final reward or adding policy calls. The completed Single-Turn system obtains 45.92 percent full success over all 1,459 tasks. Its strongest remaining weakness is option satisfaction, which shows that successful interaction still depends on the exact purchased configuration. The four-setting comparison and component removals test the principle across dialogue, profiles, and design variants. Together, these evaluations support one principle. Useful progress should remain attached to the product configuration that makes it valid.

AI USE STATEMENT

Generative AI tools assisted with experiment implementation and execution, and with editing the manuscript for clarity and fluency. The authors reviewed the resulting code, experimental outputs, and text, and take responsibility for the final content.

REPRODUCIBILITY STATEMENT

The problem formulation, method, and appendix specify the candidate-consistent constraint state, binding-trace supervision, frontier credit, the scenario protocol matrix, evaluator dimensions, training and decoding settings, and the evaluation integrity checks. Tables are generated from the evaluation outputs, and the aggregation script rejects duplicate task identifiers, preserves failed executions as zeros, and verifies the checkpoint binding, protocol digest, scenario name, task count, and summary digest. Point estimates are reported because the source environment reports no dispersion. Anonymous code will accompany the submission.

REFERENCES

Hao Bai et al. DigiRL: Training in-the-wild device-control agents with autonomous reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024.

Haibin Chen, Kangtao Lv, Chengwei Hu, Yanshi Li, Yujin Yuan, Yancheng He, Xingyao Zhang, Langming Liu, Shilei Liu, Wenbo Su, et al. Chineseecomqa: A scalable e-commerce concept evaluation benchmark for large language models. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 5311–5321, 2025.

Sanxing Chen, Sam Wiseman, and Bhuwan Dhingra. ChatShop: Interactive information seeking with language agents. *arXiv preprint arXiv:2404.09911*, 2024.

Yiruo Cheng, Kelong Mao, Tianhao Li, Jiejun Tan, Ji-Rong Wen, and Zhicheng Dou. ChatShopBuddy: Towards reliable conversational shopping agents via reinforcement learning. *arXiv preprint arXiv:2603.06065*, 2026.

Thibault Le Sellier de Chezelles et al. The BrowserGym ecosystem for web agent research. *arXiv preprint arXiv:2412.05467*, 2024.

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

Ming Gong, Xucheng Huang, Chenghan Yang, Xianhan Peng, Haoxin Wang, Yang Liu, and Ling Jiang. Mindflow: Revolutionizing e-commerce customer support with multimodal llm agents. *arXiv preprint arXiv:2507.05330*, 2025.

Yu Gu et al. Is your LLM secretly a world model of the internet? Model-Based planning for web agents. *arXiv preprint arXiv:2411.06559*, 2024.

Hongliang He et al. OpenWebVoyager: Building multimodal web agents via iterative real-world exploration, feedback and optimization. *arXiv preprint arXiv:2410.19609*, 2024a.

Hongliang He et al. WebVoyager: Building an end-to-end web agent with large multimodal models. In *Annual Meeting of the Association for Computational Linguistics*, 2024b.

Christian Herold, Michael Kozielski, Tala Bazazo, Pavel Petrushkov, Patrycja Cieplicka, Dominika Basaj, Yannick Versley, Seyyed Hadi Hashemi, and Shahram Khadivi. Domain adaptation of foundation llms for e-commerce. *arXiv preprint arXiv:2501.09706*, 2025.

Wei Jin, Haitao Mao, Zheng Li, Haoming Jiang, Chen Luo, Hongzhi Wen, Haoyu Han, Hanqing Lu, Zhengyang Wang, Ruirui Li, et al. Amazon-m2: A multilingual multi-locale shopping session dataset for recommendation and text generation. *Advances in Neural Information Processing Systems*, 36, 2024a.

Yilun Jin, Zheng Li, Chenwei Zhang, Tianyu Cao, Yifan Gao, Pratik Jayarao, Mao Li, Xin Liu, Ritesh Sarkhel, Xianfeng Tang, et al. Shopping mmlu: A massive multi-task online shopping benchmark for large language models. *Advances in Neural Information Processing Systems*, 37:18062–18089, 2024b.

Jing Yu Koh et al. VisualWebArena: Evaluating multimodal agents on realistic visual web tasks. In *Annual Meeting of the Association for Computational Linguistics*, 2024.

Hanyu Lai et al. AutoWebGLM: A large language model-based web navigating agent. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024.

Yangning Li, Shirong Ma, Xiaobin Wang, Shen Huang, Chengyue Jiang, Hai-Tao Zheng, Pengjun Xie, Fei Huang, and Yong Jiang. Ecomgpt: Instruction-tuning large language models with chain-of-task tasks for e-commerce. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 18582–18590, 2024.

Chengbo Liu, Lifang Zhou, Ruijie Yan, Pei Tan, Ao Sun, Haojun Huang, Guichun Hua, Sining Wei, Yining Chen, Yingying He, and Yutao Xie. RMSWeb: Reflection, failure-mode mining, and Salvage-DS for web agent reinforcement learning. *arXiv preprint arXiv:2608.00335*, 2026a.

Guangyi Liu, Huan Zhao, and Quanming Yao. Falsifiable commitment planning for self-correcting web agents. *arXiv preprint arXiv:2607.24167*, 2026b.

Langming Liu, Haibin Chen, Yuhao Wang, Yujin Yuan, Shilei Liu, Wenbo Su, Xiangyu Zhao, and Bo Zheng. Eckgbench: Benchmarking large language models in e-commerce leveraging knowledge graph. *arXiv preprint arXiv:2503.15990*, 2025.

Xing Han Lù et al. WebLINX: Real-world website navigation with multi-turn dialogue. In *International Conference on Machine Learning*, 2024.

Yougang Lyu, Xiaoyu Zhang, Lingyong Yan, Maarten de Rijke, Zhaochun Ren, and Xiuying Chen. DeepShop: A benchmark for deep research shopping agents. *arXiv preprint arXiv:2506.02839*, 2025.

Reiichiro Nakano et al. WebGPT: Browser-assisted question-answering with human feedback. *arXiv preprint arXiv:2112.09332*, 2021.

Guangtao Nie, Rong Zhi, Xiaofan Yan, Yufan Du, Xiangyang Zhang, Jianwei Chen, Mi Zhou, Hongshen Chen, Tianhao Li, Ziguang Cheng, et al. A hybrid multi-agent conversational recommender system with llm and search engine in e-commerce. In *Proceedings of the 18th ACM Conference on Recommender Systems*, pp. 745–747, 2024.

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.

Yichen Pan, Dehan Kong, Sida Zhou, Cheng Cui, Yifei Leng, Bing Jiang, Hangyu Liu, Yanyi Shang, Shuyan Zhou, Tongshuang Wu, et al. Webcanvas: Benchmarking web agents in online environments. *arXiv preprint arXiv:2406.12373*, 2024.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Ralph Peeters, Aaron Steiner, Luca Schwarz, Julian Yuya Caspary, and Christian Bizer. WebMall–a multi-shop benchmark for evaluating web agents. *arXiv preprint arXiv:2508.13024*, 2025.

Bo Peng, Xinyi Ling, Ziru Chen, Huan Sun, and Xia Ning. ecellm: Generalizing large language models for e-commerce from large-scale, high-quality instruction data. *arXiv preprint arXiv:2402.08831*, 2024.

Zhiyuan Peng, Xuyang Wu, Huaixiao Tou, Yi Fang, and Yu Gong. MemRerank: Preference memory for personalized product reranking. *arXiv preprint arXiv:2603.29247*, 2026.

Zehan Qi et al. WebRL: Training LLM web agents via self-evolving online curriculum reinforcement learning. In *International Conference on Learning Representations*, 2025.

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

Chandan K Reddy, Lluís Màrquez, Fran Valero, Nikhil Rao, Hugo Zaragoza, Sambaran Bandyopadhyay, Arnab Biswas, Anlu Xing, and Karthik Subbian. Shopping queries dataset: A large-scale esci benchmark for improving product search. *arXiv preprint arXiv:2206.06588*, 2022.

John Schulman et al. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, and Y. Wu. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Noah Shinn et al. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

Hongjin Su et al. Learn-by-Interact: A data-centric framework for self-adaptive agents. *arXiv preprint arXiv:2501.10893*, 2025.

Chenkai Sun, Ke Yang, Revanth Gangi Reddy, Yi R Fung, Hou Pong Chan, ChengXiang Zhai, and Heng Ji. Persona-db: Efficient large language model personalization for response prediction with collaborative data refinement. *arXiv preprint arXiv:2402.11060*, 2024.

Lu Sun, Shihan Fu, Bingsheng Yao, Yuxuan Lu, Wenbo Li, Hansu Gu, Jiri Gesi, Jing Huang, Chen Luo, and Dakuo Wang. Llm agent meets agentic ai: Can llm agents simulate customers to evaluate agentic-ai-based shopping assistants? *arXiv preprint arXiv:2509.21501*, 2025.

Shulin Tian, Ziniu Zhang, Liangyu Chen, and Ziwei Liu. MMInA: Benchmarking multihop multimodal Internet agents. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 13682–13697, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-256-5. doi: 10.18653/v1/2025.findings-acl.703. URL https://aclanthology.org/2025.findings-acl.703/.

Alicia Tsai, Adam Kraft, Long Jin, Chenwei Cai, Anahita Hosseini, Taibai Xu, Zemin Zhang, Lichan Hong, Ed Chi, and Xinyang Yi. Leveraging LLM reasoning enhances personalized recommender systems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Findings of the Association for Computational Linguistics ACL 2024*, pp. 13176–13188, Bangkok, Thailand and virtual meeting, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-acl.780. URL https://aclanthology.org/2024.findings-acl.780.

Jiangyuan Wang, Kejun Xiao, Qi Sun, Huaipeng Zhao, Tao Luo, Jian Dong Zhang, and Xiaoyi Zeng. ShoppingBench: A real-world intent-grounded shopping benchmark for LLM-based agents. *arXiv preprint arXiv:2508.04266*, 2025a.

Pei Wang, Yanan Wu, Xiaoshuai Song, Weixun Wang, Gengru Chen, Zhongwen Li, Kezhong Yan, Qi Liu, Ken Deng, Shuaibing Zhao, Shaopan Xiong, Xuepeng Liu, Xuefeng Chen, Wanxi Deng, Wenbo Su, and Bo Zheng. ShopSimulator: Evaluating and exploring RL-driven LLM agent for shopping assistants. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 35600–35619, San Diego, California, United States, July 2026. Association for Computational Linguistics. ISBN 979-8-89176-390-6. doi: 10.18653/v1/2026.acl-long.1647. URL https://aclanthology.org/2026.acl-long.1647/.

Weixun Wang, Shaopan Xiong, Gengru Chen, Wei Gao, Sheng Guo, Yancheng He, Ju Huang, Jiaheng Liu, Zhendong Li, Xiaoyang Li, et al. Reinforcement learning optimization for large-scale learning: An efficient and user-friendly scaling library. *arXiv preprint arXiv:2506.06122*, 2025b.

Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Yingzhen Yang. Recmind: Large language model powered agent for recommendation. *arXiv preprint arXiv:2308.14296*, 2023.

Ziyi Wang, Yuxuan Lu, Wenbo Li, Amirali Amini, Bo Sun, Yakov Bart, Weimin Lyu, Jiri Gesi, Tian Wang, Jing Huang, et al. Opera: A dataset of observation, persona, rationale, and action for evaluating llms on human online shopping behavior simulation. *arXiv preprint arXiv:2506.05606*, 2025c.

Zhepei Wei et al. WebAgent-R1: Training web agents via end-to-end multi-turn reinforcement learning. In *Conference on Empirical Methods in Natural Language Processing*, 2025.

Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, et al. A survey on large language models for recommendation. *World Wide Web*, 27(5):60, 2024.

Tianbao Xie et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems*, 2024.

Da Xu, Danqing Zhang, Guangyu Yang, Bo Yang, Shuyuan Xu, Lingling Zheng, and Cindy Liang. Survey for landing generative AI in social and e-commerce recsys–the industry perspectives. *arXiv preprint arXiv:2406.06475*, 2024.

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

Rui Yang, Qianhui Wu, Yuxi Chen, Hao Bai, Wenlin Yao, Hao Cheng, Baolin Peng, Huan Zhang, Tong Zhang, and Jianfeng Gao. OpenWebRL: Demystifying online multi-turn reinforcement learning for visual web agents. *arXiv preprint arXiv:2606.02031*, 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*, volume 35, pp. 20744–20757, 2022.

Shunyu Yao et al. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations*, 2023.

Chao Yi, Dian Chen, Gaoyang Guo, Jiakai Tang, Jian Wu, Jing Yu, Mao Zhang, Sunhao Dai, Wen Chen, Wenjun Yang, et al. Recgpt technical report. *arXiv preprint arXiv:2507.22879*, 2025.

Qiying Yu et al. DAPO: An open-source LLM reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Shuo Zhang, Boci Peng, Xinping Zhao, Boren Hu, Yun Zhu, Yanjia Zeng, and Xuming Hu. Llasa: Large language and e-commerce shopping assistant. In *Amazon KDD Cup 2024 Workshop*, 2024.

Yimeng Zhang, Tian Wang, Jiri Gesi, Ziyi Wang, Yuxuan Lu, Jiacheng Lin, Sinong Zhan, Kun Qian, Yuxin Tang, Vianne Gao, Ruochen Jiao, Junze Liu, Ran Xue, Houyu Zhang, Qingjun Cui, Yufan Guo, and Dakuo Wang. Shop-R1: Rewarding LLMs to simulate human behavior in online shopping via reinforcement learning. *arXiv preprint arXiv:2507.17842*, 2025.

Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. GPT-4V(ision) is a generalist web agent, if grounded. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61349–61385. PMLR, 2024. URL https://proceedings.mlr.press/v235/zheng24e.html.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A realistic web environment for building autonomous agents. In *The Twelfth International Conference on Learning Representations*, 2024a. URL https://openreview.net/forum?id=oKn9c6ytLx.

Yifei Zhou et al. Proposer-agent-evaluator (PAE): Autonomous skill discovery for foundation model internet agents. *arXiv preprint arXiv:2412.13194*, 2024b.

Table 3: **Evaluation scenarios.** Task counts and interaction limits.

Scenario	Tasks	Max steps	Profile	Shopper dialogue
Single-Turn	1,459	30	No	No
Single-Turn with personalization	1,343	30	Yes	No
Multi-Turn	1,459	40	No	Yes
Multi-Turn with personalization	1,343	40	Yes	Yes

A Additional Method Details

A.1 Worked Shopping Example

Consider a request for a black running shoe in size 40 below a price limit. The agent first views product x_1, whose page confirms the color and price but shows that size 40 is unavailable. It then views product x_2, whose page confirms size 40 but does not establish the requested color. Independent requirement tracking would mark color, size, and price as covered. That union cannot support a purchase because no observed configuration satisfies all three requirements.

ShopBind assigns each relation to one configuration. Product x_1 receives support for color and price together with a contradiction for size. Its valid support set is therefore empty. Product x_2 receives support for size and uncertainty for color. Its valid support contains only the size requirement. If a later page for x_2 confirms black and the price bound, its valid support expands for the first time and creates one frontier event. Revisiting the same page does not create another event. This example shows why candidate identity and first-time expansion are both needed.

A.2 Ordering and Mean Preservation

The tie rule has three useful properties. They define what the method preserves and avoid a stronger claim about token-level ordering.

Terminal ordering. Let $A_i^G < A_j^G$ and let $\Delta = A_j^G - A_i^G$ be the corresponding positive group gap. Since $\eta_i, \eta_j \in [-1, 1]$ and $\rho_G \leq \Delta/2$,

$$A_i^{\text{tie}} - A_j^{\text{tie}} = -\Delta + \rho_G(\eta_i - \eta_j) \leq 0. \tag{9}$$

The lower terminal advantage can meet but cannot exceed the higher terminal advantage.

Tie-set mean. For every tie set $\mathcal{T}(a)$, the normalized residuals sum to zero. The group mean is therefore unchanged within that set.

$$\frac{1}{|\mathcal{T}(a)|} \sum_{k \in \mathcal{T}(a)} A_k^{\text{tie}} = a + \frac{\rho_G}{|\mathcal{T}(a)|} \sum_{k \in \mathcal{T}(a)} \eta_k = a. \tag{10}$$

Temporal mean. The token-weighted redistribution also has zero mean over eligible tokens. Substituting the definition of $A_{k,t}$ gives

$$\frac{\sum_t w_{k,t} A_{k,t}}{\sum_t w_{k,t}} = A_k^{\text{tie}} + \lambda |A_k^{\text{tie}}| \frac{\sum_t w_{k,t} \zeta_{k,t}}{\sum_t w_{k,t}} = A_k^{\text{tie}}. \tag{11}$$

The trajectory mean is preserved before objective clipping. Individual token advantages may still change their ordering after redistribution and clipping.

A.3 Scenario Protocol Matrix

Table 3 fixes the denominator and interaction boundary for every setting. The policy receives one visible browser observation at each step. Personalized scenarios also provide a static profile. Multi-Turn scenarios allow the policy to ask the shopper for missing requirements. Evaluation ends after a purchase or after the scenario action limit.

Table 4: **Evaluator dimensions.** Definitions used throughout the paper.

Metric	Meaning
R_{loose}	Additive partial satisfaction after category matching
R_{strict}	Multiplicative satisfaction over category, attributes, options, and price
R_{succ}	Exact success on every required dimension
R_{finish}	Whether the trajectory completes a purchase
R_{cat}	Match quality for the product category
R_{att}	Fraction of required product attributes satisfied
R_{opt}	Fraction of required product options satisfied
R_{price}	Whether the purchased configuration respects the price bound

A.4 Statistical Reporting

Each complete test-set run contributes one value per metric. For n repeated runs with values $z_1, \ldots, z_n$, tables report the mean and population standard deviation

$$\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i \qquad \sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(z_i - \bar{z})^2}. \tag{12}$$

A complete single run is displayed as one value because a dispersion estimate cannot be obtained from one observation. All metrics use the declared task count. Failed execution, missing output, timeout, and no-purchase cases remain in the denominator with zero reward.

A.5 Evaluator Dimensions

Table 4 summarizes the quantities returned by the ShopSimulator evaluator. The first three are the headline task metrics. The remaining terms explain which part of a purchase succeeded or failed.

B Training Procedure

B.1 Shared Supervised Initialization

The paired experiment begins from the same 6,000-trajectory cohort and one Qwen3-8B revision. The action-only target contains the next environment action. The ShopBind target contains the visible needs, one observed configuration, their signed relations, one unresolved frontier, and the same next action. Both targets preserve the recorded trajectory order. Both SFT runs use four epochs, batch size 32, learning rate 10^{-5}, and 750 optimizer updates.

B.2 Scenario-Wise Reinforcement Learning

Each scenario then follows the same ordered procedure.

1. Load the scenario-qualified SFT checkpoint and the predefined task schedule.
2. Sample 32 prompts and eight rollouts per prompt.
3. Execute every rollout through the ShopSimulator environment under the scenario action limit.
4. Compute the terminal R_{strict} score for every trajectory.
5. Standardize terminal scores within each GRPO group to obtain A_k^G.
6. Recompute ShopBind relations from each visible prefix for the ShopBind arm.
7. Resolve equal terminal advantages with the bounded zero-mean lifecycle residual.
8. Redistribute a bounded fraction of the resulting advantage over eligible trace and action tokens.
9. Apply the same clipped GRPO objective and optimizer settings to both arms.
10. Save periodic checkpoints and apply the same checkpoint-selection rule.

Table 5: **Single-Turn evaluation accounting.** Counts over the complete test set.

Quantity	Count
Test tasks	1,459
Exact successes	670
Completed purchases	1,184
Action-limit failures	119
Malformed or unavailable actions	156
API failures	0
Empty model responses	0
Environment termination failures	0

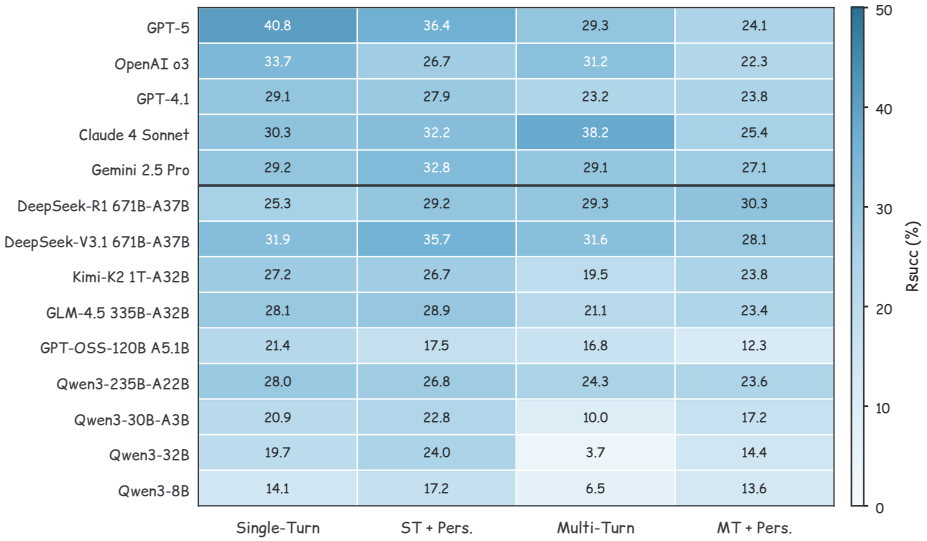

Figure 6: Full success of ShopSimulator agents across four interaction settings.

The terminal-only arm skips steps six through eight and assigns A_k^G to all response tokens. The two arms otherwise use the same task schedule, rollout count, optimizer budget, and decoding settings.

B.3 Checkpoint and Evaluation Integrity

Each evaluation checkpoint is identified before the test tasks are opened. A complete run must contain one receipt for every task in the scenario denominator. The aggregation script rejects duplicate task identifiers and preserves failed executions as zeros. It also checks the checkpoint binding, protocol digest, scenario name, task count, and summary digest before a scalar result can enter a paper table.

Table 5 gives the accounting for the completed Single-Turn ShopBind run. The success count and purchase count are distinct because an episode can complete a purchase without satisfying every required dimension.

C Complete Environment Results

Figure 6 condenses the evaluated agent range to the primary full-success metric across all four scenarios (Wang et al., 2026). It shows that model scale alone does not determine interaction quality. Several smaller systems remain competitive in particular settings, while every agent leaves substantial room below full task success.

Figure 7 reorganizes the complete Qwen3-8B training matrix by scenario and evaluator dimension. Across all four scenarios, SFT followed by strict-reward RL gives the strongest full-success score. The multi-turn settings remain the hardest even when completion and category scores are high. This gap motivates candidate-bound progress signals rather than a larger terminal reward alone.

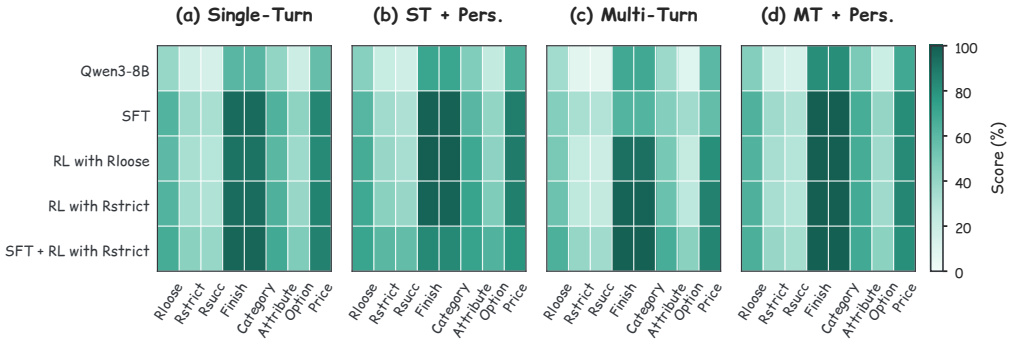

Figure 7: **ShopSimulator training profiles.** Evaluator scores for five Qwen3-8B training systems.

D More Experiments

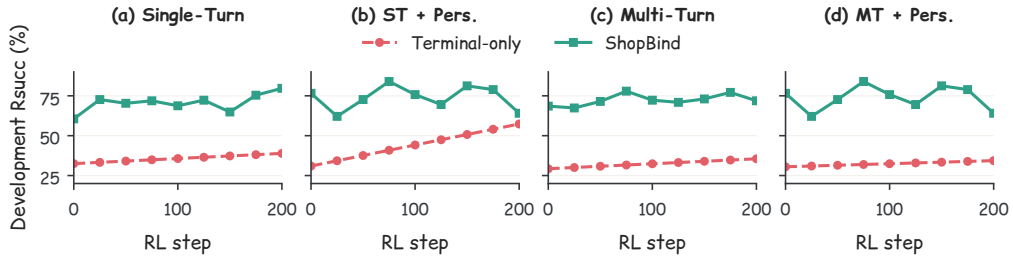

Figure 8: **Training stability.** Development success across the fixed RL budget.

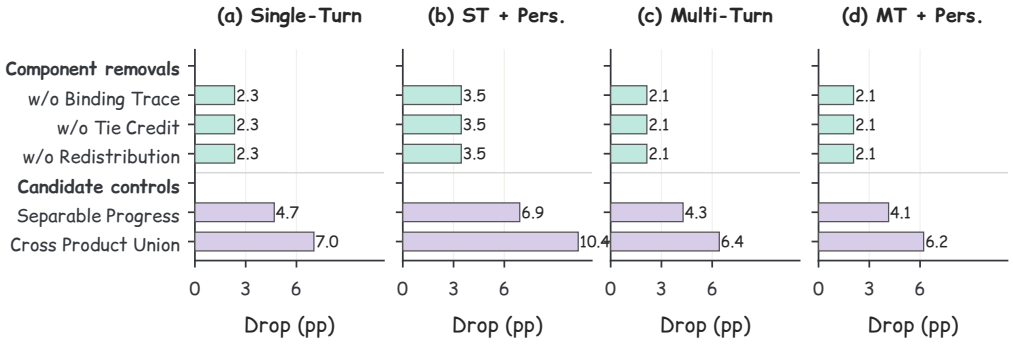

Figure 9: **Ablation profile.** Full success drops under component and identity controls.

Figures 8 and 9 separate optimization dynamics from component necessity. The first tracks development success across the fixed update budget. The second groups the full method, single component removals, and candidate identity controls under the same four settings.

D.1 Full Evaluator Breakdown

Table 6 reports every component exposed by the released evaluator for the two paired methods. This breakdown is explanatory. It identifies whether any success-rate difference coincides with task completion, category, attribute, option, or price satisfaction, but it does not replace R_{succ} as the primary metric.

D.2 Mechanism and Identity Controls

The mechanism analysis compares candidate-consistent progress with separable need/product progress, cross-product relation union, candidate-identity shuffling, temporal-event placebos, and

Table 6: **Evaluator breakdown.** Constraint scores in four settings.

Method	R_{loose}	R_{strict}	R_{succ}	R_{finish}	R_{cat}	R_{att}	R_{opt}	R_{price}
Single-Turn								
ShopSimulator SFT + RL w. R_{strict}	67.01	43.21	38.89	96.79	96.79	68.27	47.26	85.99
ShopBind	65.76	50.00	45.92	81.15	81.15	67.69	52.30	74.23
Single-Turn + Pers.								
ShopSimulator SFT + RL w. R_{strict}	71.48	60.18	57.33	83.08	83.08	72.23	63.12	78.22
ShopBind	70.14	69.63	67.70	69.66	69.66	71.62	69.85	67.52
Multi-Turn								
ShopSimulator SFT + RL w. R_{strict}	64.51	39.34	35.50	98.11	98.11	66.02	43.05	84.63
ShopBind	63.30	45.52	41.92	82.26	82.26	65.46	47.64	73.05
Multi-Turn + Pers.								
ShopSimulator SFT + RL w. R_{strict}	65.06	38.50	34.35	99.01	99.01	68.08	42.62	80.49
ShopBind	63.84	44.55	40.56	83.01	83.01	67.51	47.16	69.48

generic process-progress credit. These comparisons use the same visible information, task schedule, tuning opportunity, and rollout budget as ShopBind.

E More Experimental Setups and Implementation Details

Shared demonstrations. We collect the first 1,500 successful GPT-5.5 trajectories in each scenario. The resulting $4 \times 1{,}500$ cohort is interleaved in a predefined rank-major order across Single-Turn, Multi-Turn, Single-Turn with personalization, and Multi-Turn with personalization. Both SFT arms consume the same ordered trajectory identifiers. At every step the teacher sees the user request, the profile when present, the visible browser observation, the available environment actions, and the interaction history. It emits one `search[...]`, `click[...]`, or `ask_shopper[...]` action. Hidden targets, reward fields, and evaluator outputs are excluded from the teacher prompt. The action-only arm supervises the environment action. The structured arm supervises a binding trace followed by the same action.

E.1 Binding Annotation Rules

The annotation procedure applies five ordered rules.

1. A need enters $\mathcal{N}_t$ only after its value appears in the visible instruction, profile, or shopper reply.
2. A configuration enters $\mathcal{X}_t$ only after the corresponding product identifier and selected options appear in a browser observation.
3. Support and contradiction require a matching visible text span. Missing product text is recorded as unknown.
4. Relations never transfer between product identifiers or incompatible option tuples.
5. A frontier event is counted once when contradiction-free coverage exceeds its previous high-water mark. Repeated views and the purchase action do not create new events.

These rules keep training annotations within the policy information boundary and make the SFT and RL stages use the same state definition.

E.2 Training and Decoding Settings

Table 7 records the shared training and decoding configuration.

Paired RL. Each scenario is trained separately from its scenario-qualified SFT binding. The terminal-only and ShopBind arms use identical task schedules, training seeds, rollout seeds, shopper configuration, optimizer budgets, and checkpoint-selection rules. Neither arm receives additional rollout or selection opportunities.

Evaluation accounting. The released evaluator is applied to every task in the declared denominator. A missing output is not dropped from aggregation. This prevents runtime failures or selective completion from inflating success.

Table 7: **Training and decoding settings.** Shared values for the paired experiment.

Setting	Value
Trainable backbone	Qwen3-8B
SFT trajectories	6,000
Trajectories per scenario	1,500
SFT epochs	4
SFT batch size	32
SFT learning rate	10^{-5}
SFT optimizer updates	750
RL algorithm	GRPO
RL steps per scenario	200
Prompts per RL step	32
Rollouts per prompt	8
RL learning rate	10^{-6}
KL penalty	None
Maximum context	32K tokens
Evaluation decoding	Greedy
Policy calls per action	1
Maximum actions	30 for single turn and 40 for multi turn

For the completed Single-Turn evaluation, the denominator is 1,459. The policy succeeds on 670 tasks and completes a purchase on 1,184 tasks. The mean trajectory length is 6.41 actions. There are 119 action-cap failures and 156 malformed or unavailable-action failures. API errors, empty model responses, and environment termination errors are all zero.

E.3 Measured Training Profile

The completed Single-Turn ShopBind run uses one policy call per environment action and the same 30 and 40 step limits declared for evaluation. Table 8 reports the pooled final 20 training updates. The response length, action latency, and trajectory length are measured from the same update window, which keeps the operating profile internally consistent. The table reports the operating profile of the completed ShopBind run under the stated hardware and decoding configuration.

Table 8: **Measured training profile.** Single-Turn statistics over the final 20 updates.

System	Calls/action	Step cap ST/MT	Response tokens/action	Seconds/action	Actions/episode
ShopBind	1	30 / 40	49.09	2.63	5.76

F Reporting and Result Validation

Every aggregate is computed from episode records that retain the scenario, task identifier, evaluation seed, policy checkpoint, ordered actions, terminal state, and evaluator components. The declared task count is the denominator. Missing episodes, malformed actions, action-limit failures, and runs without a completed purchase remain in that denominator with zero full success. This accounting prevents selective completion from improving an aggregate.

Repeated runs are summarized by the arithmetic mean and population standard deviation. A complete single run is reported as one value and is not assigned an artificial variance. Paired comparisons reuse scenario tasks and seeds whenever the environment supports deterministic replay. Statistical tests operate on task-level outcomes within a scenario, while the four-scenario average remains a descriptive summary rather than an additional test set.

Checkpoint selection occurs before test evaluation and uses development records only. The same selection rule applies to terminal-only GRPO and ShopBind. Test results are not used to choose a checkpoint, extend a training schedule, or allocate extra rollouts to one arm. Each reported table is generated from a structured result record so that manuscript values can be regenerated without manual transcription.

The complete agent landscape characterizes the benchmark range across model and training choices. The paired Qwen3-8B comparison then attributes the method effect under shared data, optimization,

interaction, and selection budgets. This separation keeps system position and method attribution distinct throughout the paper.

G Trace Construction and Review

G.1 Visible Prefix Scan

State construction begins after each environment response. The scanner first extracts user needs from the initial request, the optional profile, and shopper replies already present in the visible history. A profile field enters the active need set only when its relation to the current request is clear. The scanner then enumerates displayed products and the selected options attached to each product identifier. Product text that has not appeared in the interaction stays unknown.

Relations are added one product configuration at a time. An explicit match creates support, an explicit conflict creates a contradiction, and missing or ambiguous text remains unknown. A change of product identifier starts a new configuration. An incompatible option change also starts a new configuration. This rule prevents the state from combining a favorable attribute from one product with a favorable option from another product.

G.2 Common Interaction Cases

Table 9: **Trace checks.** State updates for common interaction cases.

Observed case	State update	Training treatment
Missing product attribute	Keep the relation unknown	No coverage increase
Explicit product conflict	Add a contradiction	Negative lifecycle value when newly created
New compatible option	Create the updated configuration	Recompute coherent coverage
Product identifier change	Start a separate configuration	Never transfer prior relations
Repeated clarification	Preserve the current state	No repeated frontier event
Purchase before readiness	Preserve the visible state	Negative lifecycle value

Table 9 makes the state transition for each common case explicit. The checks keep annotations deterministic and reviewable. Every support or contradiction relation must point to a visible span. Ambiguous language remains unknown rather than being forced into a positive or negative relation.

G.3 Manual Review Protocol

The annotation audit samples prefixes from every scenario, action type, and lifecycle value. It also covers several displayed products, option changes, profile fields, and shopper corrections. Two reviewers inspect the visible interaction and record support, contradiction, or unknown for each active need on one product configuration. They verify the frontier and lifecycle value by locating the displayed spans that justify each relation. Unknown is required when the text does not settle a relation. The audit stores the scanner version, annotation hashes, reviewer decisions, agreement rates, and adjudication notes.

G.4 Scenario-Specific Interpretation

The four scenarios separate dialogue access from profile access. Single-Turn tasks test product and option consistency under a complete request. Multi-Turn tasks also test whether the policy asks for missing requirements and updates the active need set after the shopper responds. Personalized tasks test whether relevant profile fields are used without overriding the current request. Every paired method sees the same profile and shopper configuration. Reporting all four cells prevents one interaction mode from hiding a failure in another.

G.5 Failure Taxonomy

Clarification failures ask for visible information, omit a missing hard requirement, or apply an irrelevant profile field. Search failures omit an active need or abandon a strong match without a

visible reason. Inspection failures transfer attributes or options across products, while purchase failures act on unknown or contradicted relations. This sequence separates errors in the ShopBind state from errors in the environment action. Qualitative cases explain the failure loci summarized by the complete task metrics.

H LIMITATIONS AND BROADER IMPACT

ShopBind currently reads interface text. Image and video observations require additional binding operators. The study covers Qwen3-8B in one Chinese environment, motivating evaluation across models, languages, catalogs, and live sites. Deployment combines candidate-consistent state with user confirmation, catalog verification, and platform safeguards for consequential purchases.