RealWorldShop: Benchmarking and Improving Conversational Shopping Agents in Real-World E-commerce

Xinwei Yang[1,2,4*] **Kelong Mao**[1] **Yudong Guo**[1] **Sulong Xu**[1] **Simiu Gu**[1]
Chen Huang[2,3,4†] **Wenqiang Lei**[2,4]
[1] JD.com
[2] College of Computer Science, Sichuan University
[3] Institute of Data Science, National University of Singapore
[4] Engineering Research Center of Machine Learning and Industry Intelligence, Ministry of Education, China
xinwei_yang@stu.scu.edu.cn huangc.scu@gmail.com

Abstract

Large language models are reshaping e-commerce from static recommenders into interactive shopping assistants, yet real-world shopping requires session-level decision support: users reveal and revise constraints, coordinate multiple goals, and expect product-grounded recommendations over a full conversation. Existing benchmarks are mostly outcome-oriented or execution-oriented, leaving this evolving decision process under-evaluated. We introduce REALWORLDSHOP, a benchmark built on 3.28M grounded products, structured shopping episodes, a profile-grounded and action-controlled user simulator, and role-play evaluation. Our analysis shows that current systems produce locally plausible responses but struggle with state tracking, constraint updating, and grounded convergence, especially under ambiguous intent, bundle, and multi-intent scenarios. We further propose REALSHOP_AGENT, an executable session-control framework with explicit state management, shopping-flow control, catalog-grounded retrieval, and runtime guards. Experiments show that REALSHOP_AGENT consistently outperforms strong baselines on REALWORLDSHOP.

1 Introduction

Large language models are reshaping e-commerce systems from static recommenders into interactive shopping assistants that support purchase decisions across multi-turn conversations (Wang et al., 2026a; Lyu et al., 2025; Liu et al., 2026). However, real-world shopping is not simply a matter of returning a relevant item. Users often begin with underspecified intents, reveal constraints gradually, revise earlier preferences, compare alternatives, and sometimes shift purchase targets (Kim et al., 2025a; Huang et al., 2026; Wang et al., 2026b). Valid recommendations must also be grounded in real catalog evidence, including category, price, attributes, compatibility, and fulfillment constraints (Li et al., 2026b; Chen et al., 2025; Li et al., 2026a). Consequently, real-world shopping agents must elicit hidden needs, maintain and update user state, coordinate subgoals, and guide the interaction toward a grounded and actionable purchase throughout the interaction session.

Despite growing interest in conversational shopping agents, existing benchmarks remain insufficient for evaluating agent performance in real-world shopping environments. Specifically, outcome-oriented benchmarks primarily evaluate preference elicitation or final-item relevance, but often rely on simplified product spaces, static intents, or limited state transitions (Huang et al., 2024; Qin et al., 2024; Wang et al., 2025). In contrast, execution-oriented benchmarks emphasize filtering, sorting, and tool-use behaviors, while largely overlooking the conversational decision process behind those actions (Wang et al., 2026a; Cheng et al., 2026). Consequently, current evaluation practice provides limited insight into critical real-world capabilities, such as hidden-constraint elicitation, state updating, catalog-grounded recommendation, bundle and multi-intent coordination, and deciding when to clarify, compare, recommend, or confirm (Du et al., 2025; Chhetri et al., 2025; Zhao et al., 2026; Li et al., 2026b).

In this work, we introduce REALWORLDSHOP, a real-world benchmark for conversational shopping agents with session-level evaluation. REALWORLDSHOP operationalizes realistic shopping interactions through four coordinated components. First, a 3.28M-product inventory grounds recommendations in real catalog evidence, including prices, categories, attributes, descriptions, and fulfillment signals. Second, approximately 1,200 diverse shopping scenarios instantiate realistic de-

*This work was done during Xinwei Yang's internship at JD.com.

†Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 15533–15565
October 24-29, 2026 ©2026 Association for Computational Linguistics

cision contexts, covering explicit and ambiguous intents, single- and multi-subgoal sessions, bundle construction, upgrades, repairs, restocking and urgency-driven purchases. Third, more than 2,000 synthesized user profiles support a profile-grounded and action-controlled user simulator that gradually exposes hidden intents and constraints, enabling controlled evaluation under partial observability. Finally, a session-level role-play protocol evaluates local response quality, process quality, and final convergence, distinguishing successful purchase decisions from early exits and max-turn failures. Together, these components enable the evaluation of conversational shopping agents under realistic shopping dynamics.

Using REALWORLDSHOP, **we find that current shopping agents are substantially better at turn-level response generation than at session-level decision control.** Many systems produce fluent and plausible responses at individual turns, but their performance drops when they must preserve evolving user state, revise constraints after updates, coordinate multiple subgoals, and converge to a grounded purchase decision. These failures are especially pronounced in ambiguous-intent, bundle, multi-intent, and intent-shift scenarios, where hidden needs and changing constraints must be propagated across the full trajectory. These findings suggest that realistic conversational shopping should be evaluated under partial observability, catalog grounding, and convergence pressure, rather than treated as single-turn recommendation or isolated tool execution.

To address these failure modes, we propose REALSHOP_AGENT, an executable session-control framework for conversational shopping. Rather than treating shopping conversations as open-ended response generation, REALSHOP_AGENT models them as state-driven decision processes: it maintains evolving session state, decides when to clarify, retrieve, verify, compare, recommend, or confirm, grounds product decisions in catalog evidence, and applies runtime guards against over-retrieval, unsupported SKU-level claims, and stale-state recommendations. These capabilities are implemented through four executable modules: **Session State Manager**, **Shopping-Flow Controller**, **Catalog-Grounded Retriever**, and **Runtime Execution Guards**. Experiments on REALWORLDSHOP show that REALSHOP_AGENT consistently outperforms strong baselines, and improves the success rate from 0.71 to 0.82 over the strongest baseline. These results suggest that robust shopping agents require explicit session-level control and grounded decision making beyond local response generation. Our contributions are threefold:

- We introduce REALWORLDSHOP, a session-level benchmark grounded in a 3.28M-SKU inventory, structured episodes, an action-controlled user simulator, and role-play evaluation.
- We propose REALSHOP_AGENT, an executable session-control framework with state management, shopping-flow control, catalog-grounded retrieval, and runtime guards.
- We show that current systems struggle with evolving session-level constraints, while REALSHOP_AGENT achieves the strongest overall performance on REALWORLDSHOP.

2 Related Work

Conversational Shopping Agent Benchmark. Existing benchmarks for conversational shopping agents mainly follow two lines. *Outcome-oriented* benchmarks evaluate preference elicitation and final-item relevance (Liang et al., 2024; Chhetri et al., 2025; Huang et al., 2024; Kim et al., 2025a; Vlachou, 2025; Chen et al., 2025), while *execution-oriented* benchmarks test shopping actions, tool calls, or task-stage completion (Liu et al., 2026; Lyu et al., 2025; Zhao et al., 2026; Wang et al., 2026b). However, real-world shopping requires agents to operate over full sessions, where users reveal needs gradually, revise constraints, reject candidates, compare alternatives, and shift or combine purchase goals. REALWORLDSHOP addresses this gap by evaluating complete multi-turn shopping trajectories under catalog grounding and partial observability, with metrics covering turn-level response quality, session-level process quality, holistic outcomes, and early-exit behavior. A detailed comparison with representative benchmarks is provided in Table 5.

Improving Conversational Shopping Agents. Recent methods improve conversational shopping agents through user profiling, strategic action selection, feedback modeling, tool-augmented reasoning, reinforcement learning, and recommendation representations (Shi et al., 2025; Kim et al., 2025b; Cai et al., 2025; Zhang et al., 2026a; Cheng et al., 2026; Zhao et al., 2025; Zhang et al., 2026b). These advances are complementary to our setting

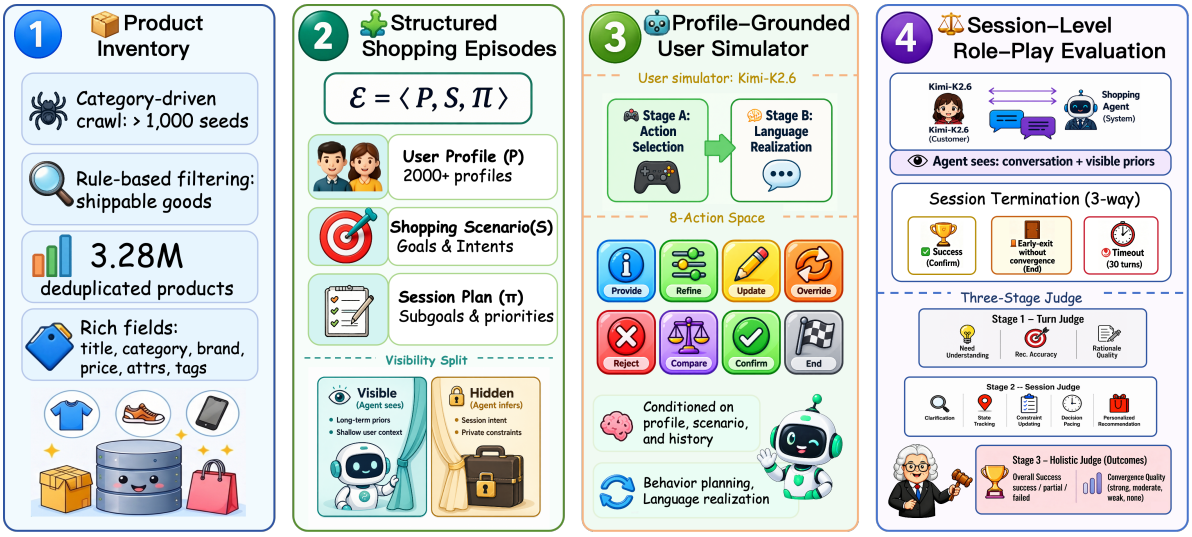

Figure 1: Overview of REALWORLDSHOP. The benchmark integrates a 3.28M-SKU product inventory, approximately 1,200 structured shopping scenarios, and over 2,000 synthesized user profiles to construct profile-grounded user simulators, together with a session-level role-play protocol for evaluating conversational shopping agents.

Component	Illustrative Episode
P	A budget-conscious, low-patience user preferring compact products, with latent constraints on counter space, noise, and household size.
S	Ambiguous bundle shopping for a small-apartment kitchen, starting from a vague request for "something useful for my kitchen."
Π	The simulator reveals space constraints, rejects an oversized appliance, updates the budget, and requests a compatible bundle.
Target	Clarification, state tracking, constraint updating, bundle coordination, and grounded convergence.

Table 1: Shopping episode illustration.

and provide useful building blocks for more capable shopping agents. However, they less directly operationalize conversational shopping as a session-control problem, where an agent must maintain evolving user state, decide when to clarify, retrieve, verify, or recommend, ground product decisions in current catalog evidence, and prevent responses that conflict with revised constraints or unsupported product claims. REALSHOP_AGENT addresses this perspective through four executable modules: a Session State Manager, a Shopping-Flow Controller, a Catalog-Grounded Retriever, and Runtime Execution Guards.

3 RealWorldShop Benchmark

Figure 1 illustrates the construction and evaluation pipeline of REALWORLDSHOP. Rather than treating conversational shopping as single-turn recommendation, REALWORLDSHOP evaluates agents as session-level decision makers through four components: (1) a real-product inventory for catalog-faithful grounding, (2) structured shopping episodes for realistic and evolving shopping contexts, (3) a profile-grounded user simulator for controlled multi-turn interaction, and (4) a role-play evaluation protocol for measuring turn quality, session process, and final convergence.

3.1 Product Inventory

We construct the product pool from authorized category-driven crawls over a large real-world e-commerce platform. After rule-based filtering and deduplication, the final inventory contains 3.28M deduplicated products across major product groups. Each record retains catalog fields such as title, price, attributes, category labels, descriptions, and fulfillment signals, enabling catalog-faithful grounding. These fields allow the benchmark to verify whether recommendations are not only preference-matched, but also faithful to real product evidence with respect to category, price, attributes, and fulfillment constraints. Further details are provided in Appendix C.1.

3.2 Shopping Episode Construction

Each evaluation instance in REALWORLDSHOP is a structured shopping episode $E = \langle P, S, \Pi \rangle$, where P specifies the user profile, S defines the shopping scenario, and Π controls how the session unfolds. This formulation separates the latent shop-

ping state from the agent-visible context, allowing the benchmark to test whether an agent can elicit, maintain, and update user needs through interaction. Episodes are synthesized through a category-anchored pipeline rather than as fixed dialogues. Starting from product category queries and source paths, the pipeline instantiates task archetypes such as bundle construction, upgrades, repairs, restocking, mixed errands, and urgency-driven purchases, together with constraints involving compatibility, budget, risk, logistics, and bundle splitting. The session plan Π specifies whether the intent is explicit or ambiguous, whether one or multiple subgoals are involved, and when hidden constraints, preference changes, rejections, or intent shifts are revealed. This enables controlled evaluation of clarification, state tracking, constraint updating, and multi-subgoal coordination. Table 1 provides an example, with further details in Appendix C.2.

3.3 User Profile Construction

Each shopping episode is grounded in a structured user profile P covering long-term preferences, historical behavior, current intent, constraints, interaction style, and optional multi-intent fields. We split each profile into an agent-visible prior $\tilde{P}$ and a latent state $P \setminus \tilde{P}$, so that agents must infer task-specific constraints through dialogue rather than assuming fully observed preferences. Profiles cover both explicit-intent cases, where the goal is directly searchable from the initial request, and ambiguous-intent cases, where goals and constraints are revealed gradually. The schema distinguishes hard, soft, and negotiable requirements over budget, compatibility, risk tolerance, logistics, brand preference, and interaction style. To improve grounding, profile synthesis can be conditioned on local catalog evidence, and unsupported constraints are downgraded to soft or negotiable preferences. All profiles are normalized and validated for schema consistency, multi-intent structure, ambiguity fields, category anchoring, and catalog grounding. Further details are provided in Appendix C.3.

3.4 Evaluation Protocol

We evaluate conversational shopping systems through a role-play-based, session-level protocol. The protocol separates three roles: the simulator generates user behavior from the hidden episode state, the evaluated agent responds using only visible context and dialogue history, and the judge evaluates the completed trajectory.

Role-play setup. For each episode, Kimi-K2.6 plays the customer, conditioned on the structured episode configuration and dialogue history. The shopping agent, i.e., the system under evaluation, observes only the conversation history and the visible prior $\tilde{P}$, and must progressively infer hidden intents and constraints through interaction.

Interaction and termination. Conversations unfold turn by turn: the simulator first generates a user utterance under its action-controlled policy, and the agent then responds. A session terminates in one of three states: SUCCESS, where the interaction reaches a grounded and actionable purchase decision; EARLY-EXIT-WITHOUT-CONVERGENCE, where the user ends the session before committing; or MAX-TURN FAILURE, where the maximum of 30 assistant turns is reached. This taxonomy separates voluntary abandonment from capacity-limited non-convergence.

Three-level Judging & Evaluation Reliability. After each episode, a three-stage GPT-5.5 judging pipeline evaluates the dialogue, tool trace, and episode configuration. The turn-level judge measures local response quality, including need understanding, recommendation accuracy, and rationale quality. The session-level judge measures process quality, including clarification, state tracking, constraint updating, decision pacing, and personalization. Finally, the holistic judge determines whether the session reaches an actionable purchase decision and assigns `overall_success` and `convergence_quality`. All turn- and session-level dimensions are scored on a $\{0, 1, 2\}$ scale. We assess judge reliability using repeated judging passes and a stratified 150-session human validation subset, where automatic judgments achieve QWK/κ scores of 0.76–0.89 and exact agreement rates of 0.88–0.97 across dimensions. Further details are provided in Appendix C.6.

4 Benchmark Analysis

4.1 Benchmark Setup

Overview. All experiments are performed on RealWorldShop, utilizing the rigorous evaluation protocol and metrics defined in Section 3.4.

Baselines. We evaluate two groups of baselines: general LLMs and recent shopping-agent methods. The LLM group includes *GPT-5* (Singh

Model	Turn-Level Quality ↑				Session-Level Process Quality ↑						Turn–Session	Holistic Outcome		User Signal
	Need	Rec. Acc.	Rationale	Turn Avg.	Clarif.	State	Constraint	Pace	Personal.	Sess. Avg.	Gap ↓	Succ. ↑	Conv. ↑	Early Exit ↓
GPT-5 (Singh et al., 2025)	**1.62**	**1.41**	**1.33**	**1.45**	**1.68**	**1.23**	**1.09**	**0.94**	**1.08**	**1.20**	0.25	**0.71**	**2.49**	**0.10**
Gemini-2.5-Flash (Comanici et al., 2025)	0.91	0.80	0.84	0.85	1.21	0.69	0.56	0.59	0.45	0.70	**0.15**	0.38	1.64	0.35
GPT-4o (Hurst et al., 2024)	1.30	0.71	0.75	0.92	0.95	0.62	0.49	0.37	0.35	0.56	0.36	0.38	1.48	0.39
DeepSeek-V3.2 (Liu et al., 2025)	1.36	0.92	0.81	1.03	1.41	0.80	0.63	0.64	0.52	0.80	0.23	0.56	1.93	0.27
Kimi-K2.6 (Team et al., 2025)	1.45	1.22	1.21	1.29	1.54	0.90	0.88	0.70	0.95	0.99	0.30	0.67	2.23	0.16
Qwen3.5-27B (Yang et al., 2025)	1.41	0.83	0.80	1.01	1.49	0.58	0.57	0.50	0.36	0.70	0.31	0.55	1.74	0.25
Qwen3.5-35B-A3B (Yang et al., 2025)	1.42	0.78	0.79	1.00	1.30	0.62	0.46	0.37	0.43	0.64	0.36	0.53	1.66	0.31
LLM Avg.	1.35	0.95	0.93	1.08	1.37	0.78	0.67	0.59	0.59	0.80	0.28	0.54	1.88	0.26
Specialized Methods (DS-V3.2 backbone)														
AgentCF (Zhang et al., 2024)	1.37	1.15	1.03	1.18	1.42	0.97	0.87	0.61	0.76	0.93	0.26	0.61	2.15	0.23
CRAVE (Zhu et al., 2025a)	1.38	1.01	0.89	1.09	1.43	0.79	0.83	0.57	0.80	0.88	0.21	0.57	1.98	0.23
PersonaX (Shi et al., 2025)	1.36	1.13	0.92	1.14	1.45	0.92	0.89	0.71	0.86	0.97	0.17	0.60	2.14	0.15
CSI (Kim et al., 2025b)	1.25	1.18	0.93	1.12	1.43	0.71	0.70	0.63	0.82	0.86	0.26	0.59	1.94	0.24
AFL (Cai et al., 2025)	1.29	1.19	0.96	1.15	1.33	0.77	0.81	0.69	0.70	0.86	0.29	0.62	1.90	0.21
Method Avg.	1.33	1.13	0.95	1.14	1.41	0.83	0.82	0.64	0.79	0.90	0.24	0.60	2.02	0.21

Table 2: Overall benchmark results on REALWORLDSHOP. Turn Avg. is the arithmetic mean of Need, Rec. Acc., and Rationale, while Sess. Avg. is the arithmetic mean of Clarif., State, Constraint, Pace, and Personal. Gap measures the degradation from turn-level quality to session-level process quality and is computed as the unrounded Turn Avg. minus the unrounded Sess. Avg. Succ. denotes success rate, Conv. denotes convergence quality, and Early Exit denotes the simulator-side early-exit rate. All aggregate metrics are computed before rounding. Best and second-best results across all evaluated systems are marked in **bold** and underline, respectively. Tied results receive the same formatting.

et al., 2025), *Gemini-2.5-Flash* (Comanici et al., 2025), *GPT-4o* (Hurst et al., 2024), *DeepSeek-V3.2* (Liu et al., 2025), *Kimi-K2.6* (Team et al., 2025), *Qwen3.5-27B* (Yang et al., 2025), and *Qwen3.5-35B-A3B* (Yang et al., 2025); all models are evaluated under the same task instructions, agent-visible inputs, tool interfaces, and judging pipeline, while their own generated responses and tool traces are evaluated independently. The shopping-agent group includes *AgentCF* (Zhang et al., 2024), *CRAVE* (Zhu et al., 2025a), *PersonaX* (Shi et al., 2025), *CSI* (Kim et al., 2025b), and *AFL* (Cai et al., 2025).

Evaluation Dataset. We construct a held-out diagnostic set from REALWORLDSHOP and evaluate all systems on the same episodes. Each episode is annotated along three axes: scenario type, interaction phenomenon, and interaction profile. For each category, we sample 100 episodes, covering six scenario types: explicit single-item, explicit bundle, ambiguous single-item, ambiguous bundle, multi-intent shopping, and long-horizon sessions; six interaction phenomena: constraint refinement, product comparison, constraint update, preference override, recommendation rejection, and bundle re-planning; and six user profiles: collaborative, high-patience, low-skepticism, efficiency-first, high-skepticism, and low-patience. Early-exit and non-exit sessions are grouped by realized outcomes rather than pre-defined sampling labels. For each system, we repeat the full evaluation five times and report the mean across the five runs.

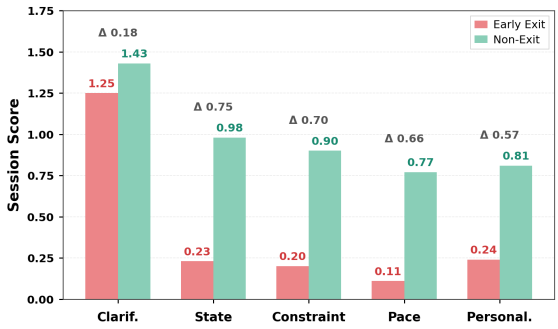

Figure 2: Early-exit sessions reveal weak session-level process control. Compared with non-exit sessions, early-exit sessions show larger degradation in state tracking, constraint updating, and decision pacing, indicating that user abandonment is closely tied to failures in maintaining and revising session state.

4.2 Benchmark Findings

Finding 1: High single-turn scores do not guarantee successful shopping sessions. Table 2 evaluates agents at two levels: *Turn Avg.* captures local response quality, whereas *Sess. Avg.* captures trajectory-level shopping competence, including state tracking, constraint updating, subgoal coordination, and grounded convergence. Across all systems, the session-level score is consistently lower than the turn-level score. The average general LLM drops from 1.08 to 0.80, and specialized shopping-agent methods drop from 1.14 to 0.90. Even GPT-5 shows the same gap, decreasing from

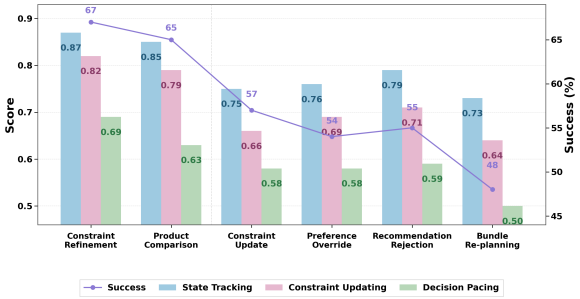

Figure 3: Interaction phenomena stress state revision and trajectory management. Performance declines under constraint updates, preference overrides, recommendation rejections, and bundle re-planning, showing that agents struggle to propagate revised user information through the full shopping trajectory.

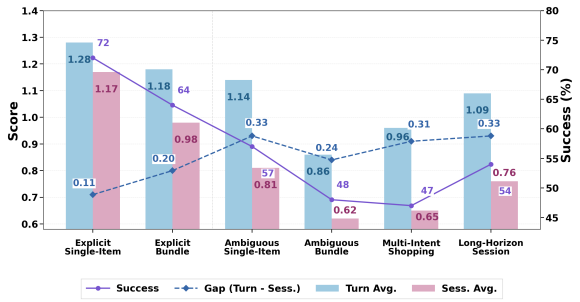

Figure 4: Scenario complexity amplifies the turn-to-session quality gap. Agents perform best in explicit single-item sessions, while ambiguous, bundle, multi-intent, and long-horizon scenarios expose larger gaps between local response quality and session-level shopping success.

1.45 to 1.20. These results indicate that locally plausible responses do not guarantee coherent shopping behavior over an entire session. An agent may handle the current user utterance well, but still fail to maintain evolving user state, coordinate multiple needs, revise constraints, and converge to a grounded purchase decision. Thus, turn-level evaluation alone risks substantially overestimating real shopping-agent competence.

Finding 2: Early exits are driven more by state-tracking failures than by poor clarification. Figure 2 shows that early-exit sessions underperform non-exit sessions across session-level dimensions, but the degradation is highly uneven. Clarification drops only moderately from 1.43 to 1.25, whereas state tracking drops sharply from 0.98 to 0.23. This contrast suggests that user abandonment is not mainly caused by insufficient questioning; rather, users are more likely to exit when the agent fails to remember, update, or act on in-

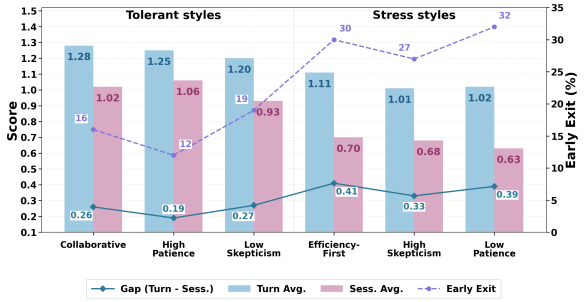

Figure 5: Demanding interaction styles increase session fragility. Efficiency-first, high-skepticism, and low-patience users lead to larger turn-to-session gaps and higher early-exit rates, highlighting the need to evaluate shopping agents under realistic convergence pressure.

formation that has already been revealed. Figure 3 further shows that failures become more severe when the dialogue requires explicit state revision. In particular, bundle re-planning is one of the most challenging cases, with success falling to 48% and process scores dropping to 0.64 for constraint updating and 0.50 for decision pacing. These results indicate that current agents struggle to propagate revised information through the session state, candidate set, and downstream recommendation plan.

Finding 3: Ambiguity, long horizons, and demanding user styles expose failures hidden by turn-level metrics. Figure 4 shows that agents perform best in explicit single-item sessions, achieving a success rate of 72% and the smallest local-to-session gap of 0.11. As scenarios become more complex, performance declines: ambiguous single-item sessions reduce success to 57%, while ambiguous single-item and long-horizon sessions jointly exhibit the largest gap of 0.33. This suggests that ambiguity, bundle construction, multi-intent coordination, and long-range dependencies amplify trajectory-level weaknesses that are not captured by turn-level quality alone. Figure 5 further shows that demanding interaction styles intensify these failures. Low-patience, high-skepticism, and efficiency-first users reduce session-level scores to 0.63–0.70 and raise early-exit rates to 27%–32%, compared with 12%–19% under more tolerant styles. Therefore, realistic shopping evaluation should stress-test agents under partial observability, evolving constraints, catalog grounding, and convergence pressure. Additional diagnostic analyses supporting these findings are provided in Appendix A, where we further examine early-exit behavior, interaction phenomena, scenario complex-

ity, and user interaction styles.

5 REALSHOP_AGENT

Overview & Problem Setup. The findings from REALWORLDSHOP suggest that conversational shopping is better viewed as a session-level decision-control problem than as isolated recommendation. Users may gradually reveal or revise needs, reject suggestions, compare alternatives, and shift purchase targets. An effective shopping agent must therefore maintain evolving state, choose when to clarify, retrieve, verify, compare, recommend, or confirm, and ground product claims in up-to-date catalog evidence.

REALSHOP_AGENT is a diagnosis-guided session-control framework that addresses these challenges through four executable modules: a Session State Manager for evolving user state, a Shopping-Flow Controller for action timing, a Catalog-Grounded Retriever for evidence grounding, and Runtime Execution Guards for preventing unsupported or stale recommendations. Together, these modules form an explicit control loop for session-level shopping.

Following the episode formulation in Section 3, each shopping episode is represented as $E = \langle P, S, \Pi \rangle$, where P is the structured user profile, S the shopping scenario, and Π the session plan. At turn t, the assistant observes only the visible prior $\widetilde{P}$ and dialogue history $h_{<t}$, while latent profile fields and future session dynamics remain hidden. REALSHOP_AGENT maintains an explicit session state M_t and selects among six high-level actions: CLARIFY, RETRIEVE, VERIFY, COMPARE, RECOMMEND, and CONFIRM. Depending on the selected action, it may invoke catalog tools $o_t \in \mathcal{T}$, update grounded evidence, and generate a response subject to grounding and state-consistency checks. Algorithmic details are provided in Appendix E.

5.1 M1: Session State Manager

The Session State Manager maintains the evolving state M_t, covering revealed intents, active subgoals, constraints, rejected options, confirmed preferences, and stale dependencies. Since shopping sessions are non-stationary, newly revealed constraints override long-term preferences, and dependent subgoals are marked stale when budgets, candidates, compatibility requirements, or bundle anchors change. The updated M_t guides downstream action selection and product grounding.

5.2 M2: Shopping-Flow Controller

The Shopping-Flow Controller maps the current session state M_t to one of six high-level shopping actions: CLARIFY, RETRIEVE, VERIFY, COMPARE, RECOMMEND, and CONFIRM. This module determines not only what information or evidence is needed, but also when the agent should progress from elicitation to search, verification, comparison, recommendation, and final commitment. It selects CLARIFY when blocking information is missing; RETRIEVE when the active subgoal is sufficiently specified but fresh catalog evidence is unavailable; VERIFY when retrieved candidates have not yet been checked against active constraints; COMPARE when multiple viable candidates remain or the user requests trade-off analysis; RECOMMEND when grounded candidates satisfy the active constraints and are aligned with the current subgoal; and CONFIRM when the user is ready to commit and the proposed recommendation or purchase plan is actionable. This explicit action policy reduces premature recommendations, redundant questioning, unsupported comparisons, and delayed convergence.

5.3 M3: Catalog-Grounded Retriever

The Catalog-Grounded Retriever connects the state-driven control process to real product evidence. When retrieval or verification is triggered, it returns structured evidence records $E_t = \{e_i\}$, including SKU identifiers, category labels, attributes, prices, fulfillment signals, relevance scores, and retrieval metadata. This module is necessary because even a well-timed recommendation can be invalid if its product claims are not supported by the current catalog evidence. Therefore, concrete SKU-level claims must be traceable to the latest retrieved records, and evidence tied to stale intents, rejected options, or superseded constraints is invalidated before recommendation.

5.4 M4: Runtime Execution Guards

These Guards form the final consistency layer before response generation. Since agents may still over-retrieve, overstate product properties, or recommend candidates that conflict with updated constraints, TOOLBUDGET limits redundant retrieval, FABRICATIONGUARD rewrites unsupported SKU-level claims into uncertainty-aware statements, and STATECONSISTENCYGUARD blocks recommendations tied to stale subgoals, rejected options, or superseded constraints. Together, these guards keep

the final response consistent with the latest session state and catalog evidence.

5.5 Backbone Adaptation

We adapt a Qwen3.5-27B backbone for tool-using shopping policies. Although the four session-control modules structure the interaction, the backbone still decides how to act under evolving constraints, such as when to clarify, retrieve, verify, recommend, or revise. We initialize it from an SFT checkpoint and further optimize it with reinforcement learning over tool-using shopping rollouts, improving shopping-specific action selection and catalog-grounded reasoning. Details are provided in Appendix E.

6 Experiment

6.1 Experimental Setup

We evaluate REALSHOP_AGENT using our REALWORLDSHOP. All experiments use the role-play protocol and three-stage judging pipeline described in Section 3.4. We report the same metric groups as Table 2: turn-level quality, session-level process quality, and final outcome quality.

6.2 Overall Performance

Table 3 reports the main results and ablations.

REALSHOP_AGENT closes the gap between local response quality and session-level shopping competence. The prompt-only Qwen3.5-27B Base shows a clear local-to-session mismatch, with a Turn Avg. of 1.01 but a Sess. Avg. of only 0.70. SFT/RL adaptation improves the backbone, raising Sess. Avg. to 1.00 and success rate from 0.55 to 0.65, but it still falls short of the full REALSHOP_AGENT. This suggests that improving the underlying policy helps, but does not by itself solve the session-level control problem.

Explicit session control provides gains beyond backbone adaptation. Equipping the original Qwen3.5-27B Base with REALSHOP_AGENT raises Sess. Avg. from 0.70 to 1.16 and success rate from 0.55 to 0.71, while reducing the local-to-session Gap from 0.31 to 0.16. The full system further achieves the best overall results, reaching a Sess. Avg. of 1.39, success rate of 0.82, the lowest Gap of 0.07, and the lowest Early Exit rate of 0.09. These gains indicate that backbone adaptation and explicit control are complementary: adaptation improves the policy, while the control framework stabilizes state tracking, action timing, grounding, and convergence over long shopping sessions.

6.3 Diagnosis-Oriented Evaluation

Beyond aggregate scores, Appendix Table 4 evaluates whether REALSHOP_AGENT addresses the failure modes identified in Section 4.2. Here, Base denotes the prompt-only Qwen3.5-27B backbone, +SFT/RL denotes the adapted backbone, and Full Agent denotes the complete REALSHOP_AGENT.

REALSHOP_AGENT reduces early exits by improving session-state control. The overall Early Exit rate decreases from 0.25 to 0.09, while Sess. Avg. on the metadata-defined early-exit-risk subset increases from 0.60 to 1.21. Substantial gains are observed in State Tracking, which improves from 0.51 to 1.33, and Constraint Updating, which improves from 0.47 to 1.24. These results confirm that early exits are associated not only with weak local responses, but also with failures to maintain and revise the evolving session state.

REALSHOP_AGENT improves recovery from state revisions and recommendation rejections. Across revision-heavy interaction phenomena, including constraint updates, preference overrides, recommendation rejections, and bundle replanning, the full agent consistently outperforms both the prompt-only and adapted backbones. The largest success-rate improvements over the prompt-only backbone occur under constraint updates and recommendation rejections, with success increasing from 0.55 to 0.83 and from 0.50 to 0.78, respectively. These results suggest that the proposed modules help the agent incorporate user feedback, propagate revised constraints through the maintained session state and downstream recommendation plan, and avoid recommendations based on stale assumptions.

REALSHOP_AGENT remains more stable under complex scenarios and demanding users. In ambiguous-bundle sessions, success increases from 0.50 to 0.74 and the Gap decreases from 0.34 to 0.11; in multi-intent shopping, success increases from 0.49 to 0.76 and the Gap decreases from 0.35 to 0.07. The agent also reduces early exits for efficiency-first, high-skepticism, and low-patience users. Further diagnostic breakdowns are provided in Appendix A, with detailed results in Table 4.

6.4 Ablation Study

All modules provide complementary gains to REALSHOP_AGENT. As shown in Table 3, removing any module degrades performance under the same

Variant	Turn-Level Quality ↑				Session-Level Process Quality ↑						Turn–Session	Holistic Outcome		User Signal
	Need	Rec. Acc.	Rationale	Turn Avg.	Clarif.	State	Constraint	Pace	Personal.	Sess. Avg.	Gap ↓	Succ. ↑	Conv. ↑	Early Exit ↓
Prompt-only Backbones														
Qwen3.5-27B Base (Yang et al., 2025)	1.41	0.83	0.80	1.01	1.49	0.58	0.57	0.50	0.36	0.70	0.31	0.55	1.74	0.25
Qwen3.5-27B + SFT/RL	1.67	1.09	0.95	1.24	1.54	0.92	0.93	0.82	0.78	1.00	0.24	0.65	2.19	0.20
Module Ablations on SFT/RL Backbone														
w/o Session State Manager	1.53	1.27	1.06	1.29	1.58	1.02	0.87	0.83	0.93	1.05	0.24	0.61	2.18	0.22
w/o Shopping-Flow Controller	1.57	1.21	1.18	1.32	1.32	1.23	1.18	1.02	1.09	1.17	0.15	0.66	2.11	0.19
w/o Catalog-Grounded Retriever	1.55	1.09	1.00	1.21	1.60	1.11	1.07	1.03	0.85	1.13	0.08	0.65	2.13	0.13
w/o Runtime Execution Guards	1.65	1.26	1.16	1.36	1.59	1.22	1.24	1.06	1.05	1.23	0.12	0.68	2.24	0.15
Backbone Swap and Full System														
REALSHOP_AGENT (Qwen3.5-27B Base)	1.60	1.24	1.11	1.32	1.51	1.08	1.19	0.98	1.02	1.16	0.16	0.71	2.36	0.13
REALSHOP_AGENT Full	**1.76**	**1.31**	**1.31**	**1.46**	**1.65**	**1.45**	**1.34**	**1.27**	**1.22**	**1.39**	**0.07**	**0.82**	**2.49**	**0.09**

Table 3: Ablation study of REALSHOP_AGENT under the same three-level metric structure as Table 2. Turn-level and session-level dimensions are scored on a $\{0, 1, 2\}$ scale. **Turn Avg.** is the macro-average of Need, Recommendation Accuracy, and Rationale, while **Sess. Avg.** is the macro-average of Clarification, State Tracking, Constraint Updating, Decision Pacing, and Personalization. **Gap** is computed as Turn Avg. minus Sess. Avg. All aggregate metrics are computed before final rounding. **Succ.** and **Conv.** are holistic-judge outcome metrics, while **Early Exit** is a simulator-side user signal. Higher values are better for all metrics except **Gap** and **Early Exit**. Best results are shown in bold, and second-best results are underlined. Tied second-best results are underlined simultaneously.

backbone, tools, and protocol, with different ablations affecting state preservation, action timing, product grounding, and long-horizon robustness.

State management preserves evolving user needs. Removing the Session State Manager causes the largest local-to-session mismatch, increasing the Gap from 0.07 to 0.24, while reducing State Tracking from 1.45 to 1.02 and Constraint Updating from 1.34 to 0.87. This indicates that revealed preferences, revised constraints, rejected options, and stale dependencies must be explicitly carried across the session.

Flow control stabilizes action timing. Without the Shopping-Flow Controller, Sess. Avg. drops from 1.39 to 1.17 and success decreases from 0.82 to 0.66. This shows that free-form action selection is insufficient for deciding when to clarify, retrieve, verify, recommend, or confirm.

Catalog grounding supports product-level decisions. Removing the Catalog-Grounded Retriever reduces Rec. Acc. from 1.31 to 1.09 and Personalization from 1.22 to 0.85. This suggests that session control must be paired with retrieval and verification against the current constraint stack.

Runtime guards improve robustness. Removing Runtime Execution Guards lowers Sess. Avg. from 1.39 to 1.23 and increases Early Exit from 0.09 to 0.15. This indicates that unsupported claims, redundant tool use, and stale-state recommendations can still destabilize long-horizon sessions.

7 Conclusion

In this work, we introduced REALWORLDSHOP, the first large-scale benchmark for evaluating conversational shopping agents under realistic shopping conditions. It integrates a 3.28M-SKU product inventory, ∼1,200 structured shopping scenarios, over 2,000 synthesized user profiles, and a session-level role-play protocol. Our benchmark reveals a substantial gap between current conversational recommendation systems and the requirements of real-world shopping. To address this gap, we proposed REALSHOP_AGENT, a diagnosis-guided session-control framework that improves task success, reduces early exits, and narrows the gap between local response quality and session-level effectiveness. More broadly, our findings suggest that future conversational shopping agents require explicit mechanisms for state tracking, planning, and session-level control. We hope REALWORLDSHOP can serve as a realistic testbed for advancing grounded, long-horizon, and decision-oriented conversational shopping agents.

Limitations

LLM-as-judge reliability and bias. Our evaluation relies on LLM-based judges. Although we include a stratified human validation subset and repeated judging passes to assess agreement and stability, LLM judges may still be sensitive to surface fluency, response style, and prompt wording. Such biases may affect absolute scores and, in some cases, relative comparisons across model families. Therefore, the reported results should be in-

terpreted with appropriate caution, and future work could further expand human evaluation and cross-judge validation (Huang et al., 2024; Zhu et al., 2026; Tian et al., 2025b, 2026a).

Simulator-based evaluation. REALWORLDSHOP uses a profile-grounded and action-controlled user simulator to create reproducible multi-turn shopping interactions. Although this allows controlled evaluation of hidden constraints, preference updates, and intent shifts, simulated users may not fully capture the behavioral diversity of real shoppers, such as emotional reactions, hesitation, impulsive decisions, or highly personalized brand preferences. Therefore, results on REALWORLDSHOP should be viewed as a controlled approximation of real-world shopping behavior rather than a complete substitute for live user studies (Kim et al., 2025a; Chen et al., 2025; Yang et al., 2026).

Benchmark coverage. Although REALWORLDSHOP covers diverse shopping scenarios, including ambiguous intent, bundle construction, multi-intent shopping, constraint updates, and urgency-driven purchases, it does not exhaust all forms of real-world e-commerce interaction. For example, after-sales service, returns and refunds, promotion-driven shopping, cross-platform price comparison, long-term customer lifecycle modeling, and live-commerce settings are not fully covered. Extending the benchmark to these scenarios would provide a more comprehensive evaluation of conversational shopping agents in future work.

Ethical Considerations

Data collection and privacy. The data collection and benchmark construction procedures for REALWORLDSHOP were conducted under an IRB-approved protocol. Product inventory construction and post-processing follow the approved protocol, applicable access permissions, and data-use requirements. During data collection and organization, we retain only product-related information required for benchmark construction, such as product titles, categories, attributes, prices, descriptions, and fulfillment signals. Any personally identifying or sensitive information that may be associated with users or sellers is removed, anonymized, or desensitized during preprocessing. We do not use personally identifiable user information for benchmark construction or evaluation.

Use of simulated users. REALWORLDSHOP uses profile-grounded and action-controlled simulated users for role-play evaluation. The simulated profiles are constructed for research evaluation purposes and do not represent real individuals. We validate the simulator for profile faithfulness and action controllability, and the benchmark is intended to evaluate shopping-agent behavior rather than to infer or reproduce real user identities.

Potential misuse and consumer protection. Conversational shopping agents may be misused to generate unsupported product claims, overly persuasive recommendations, or suggestions that violate user constraints. REALWORLDSHOP is intended for research evaluation rather than deployment, and emphasizes grounded recommendation, constraint satisfaction, and convergence quality. REALSHOP_AGENT further includes runtime guards against unsupported SKU-level claims and stale-state recommendations, while real deployed systems should still incorporate platform-specific safety policies, consumer-protection requirements, and human oversight where appropriate.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No. U25B201508 and No. U24A20328).

References

Shihao Cai, Jizhi Zhang, Keqin Bao, Chongming Gao, Qifan Wang, Fuli Feng, and Xiangnan He. 2025. Agentic feedback loop modeling improves recommendation and user simulation. In *Proceedings of the 48th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 2235–2244. ACM.

Luyu Chen, Quanyu Dai, Zeyu Zhang, Xueyang Feng, Mingyu Zhang, Pengcheng Tang, Xu Chen, Yue Zhu, and Zhenhua Dong. 2025. Recusersim: A realistic and diverse user simulator for evaluating conversational recommender systems. In *The Web Conference*, pages 133–142. ACM.

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

Vinaik Chhetri, Yousaf Reza, Moghis Fereidouni, Srijata Maji, Umar Farooq, and AB Siddique. 2025. A framework for generating conversational recommendation datasets from behavioral interactions. *arXiv.org*.

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

Hanwen Du, Bo Peng, and Xia Ning. 2025. Sapient: Mastering multi-turn conversational recommendation with strategic planning and monte carlo tree search. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2629–2648. Association for Computational Linguistics.

Chen Huang, Peixin Qin, Yang Deng, Wenqiang Lei, Jiancheng Lv, and Tat-Seng Chua. 2024. Concept–an evaluation protocol on conversational recommender systems with system-centric and user-centric factors. *arXiv.org*.

Jiani Huang, Shijie Wang, Liangbo Ning, Wenqi Fan, Shuaiqiang Wang, Dawei Yin, and Qing Li. 2026. Towards next-generation recommender systems: A benchmark for personalized recommendation assistant with llms. In *Web Search and Data Mining*, pages 217–226. ACM.

Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*.

SungHwan Kim, Kwangwook Seo, Tongyoung Kim, Jinyoung Yeo, and Dongha Lee. 2025a. Stop playing the guessing game! evaluating conversational recommender systems via target-free user simulation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 19588–19605, Suzhou, China. Association for Computational Linguistics.

Tongyoung Kim, Jeongeun Lee, SooJin Yoon, SungHwan Kim, and Dongha Lee. 2025b. Towards personalized conversational sales agents: Contextual user profiling for strategic action. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 5131–5154, Suzhou, China. Association for Computational Linguistics.

Adrian Li, Kelong Mao, Yudong Guo, Heming Xia, Xinwei Yang, Lirui Luo, Jace Wong, Pu Yao, Sulong Xu, and Simiu Gu. 2026a. Comboshoppingbench: Evaluating llm agents for budget-constrained basket shopping with coupons. *arXiv*.

Mengfan Li, Xuanhua Shi, and Yang Deng. 2026b. Rectom: A benchmark for evaluating machine theory of mind in llm-based conversational recommender systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 31636–31644. Association for the Advancement of Artificial Intelligence (AAAI).

Tingting Liang, Chenxin Jin, Lingzhi Wang, Wenqi Fan, Congying Xia, Kai Chen, and Yuyu Yin. 2024. Llm-redial: A large-scale dataset for conversational recommender systems created from user behaviors with llms. In *Annual Meeting of the Association for Computational Linguistics*, pages 8926–8939. Association for Computational Linguistics.

Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, et al. 2025. Deepseek-v3.2: Pushing the frontier of open large language models. *arXiv preprint arXiv:2512.02556*.

Fei Liu, Xinyu Lin, Hanchao Yu, Mingyuan Wu, Jianyu Wang, Qiang Zhang, Zhuokai Zhao, Yinglong Xia, Yao Zhang, Weiwei Li, et al. 2026. Recoworld: Building simulated environments for agentic recommender systems. In *Companion Proceedings of the ACM Web Conference 2026*, pages 650–659.

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

Peixin Qin, Chen Huang, Yang Deng, Wenqiang Lei, and Tat-Seng Chua. 2024. Beyond persuasion: Towards conversational recommender system with credible explanations. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 4264–4282. Association for Computational Linguistics.

Yunxiao Shi, Wujiang Xu, Zeqi Zhang, Xing Zi, Qiang Wu, and Min Xu. 2025. Personax: A recommendation agent-oriented user modeling framework for long behavior sequence. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5764–5787. Association for Computational Linguistics.

Aaditya Singh, Adam Fry, Adam Perelman, Adam Tart, Adi Ganesh, Ahmed El-Kishky, Aidan McLaughlin, Aiden Low, AJ Ostrow, Akhila Ananthram, et al. 2025. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*.

Kimi Team, Yifan Bai, Yiping Bao, Y Charles, Cheng Chen, Guanduo Chen, Haiting Chen, Huarong Chen, Jiahao Chen, Ningxin Chen, et al. 2025. Kimi k2: Open agentic intelligence. *arXiv preprint arXiv:2507.20534*.

Shi-Yu Tian, Zhuo-Xia Wang, Xuan-Yi Zhu, Zhi Zhou, Xinwei Yang, Kun-Yang Yu, Ming Yang, Yang Chen, and Yu-Feng Li. 2026a. Self-evolving neuro-symbolic skills for tool-augmented spatial reasoning. *arXiv preprint arXiv:2608.07955*.

Shi-Yu Tian, Zhi Zhou, Wei Dong, Kun-Yang Yu, Ming Yang, Zi-Jian Cheng, Lan-Zhe Guo, and Yu-Feng Li. 2026b. Tabularmath: Understanding math reasoning over tables with large language models. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 15469–15498.

Shi-Yu Tian, Zhi Zhou, Xin Su, and Yu-Feng Li. 2025a. Rethinking evaluation for multi-label drug-drug interaction prediction. *Frontiers of Computer Science*, 19(9):199358.

Shi-Yu Tian, Zhi Zhou, Kun-Yang Yu, Ming Yang, Lin-Han Jia, Lan-Zhe Guo, and Yu-Feng Li. 2025b. Vcsearch: Bridging the gap between well-defined and ill-defined problems in mathematical reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 12710–12731.

Maria Vlachou. 2025. Fashion-altereval: A dataset for improved evaluation of conversational recommendation systems with alternative relevant items. In *ACM Conference on Recommender Systems*, pages 755–763. ACM.

Jiangyuan Wang, Kejun Xiao, Qi Sun, Huaipeng Zhao, Tao Luo, Jian Dong Zhang, and Xiaoyi Zeng. 2026a. Shoppingbench: A real-world intent-grounded shopping benchmark for llm-based agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pages 33521–33529. Association for the Advancement of Artificial Intelligence (AAAI).

Xiaolei Wang, Chunxuan Xia, Junyi Li, Fanzhe Meng, Lei Huang, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. 2025. Search-based interaction for conversation recommendation via generative reward model based simulated user. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 75–84. ACM.

Ziyi Wang, Yuxuan Lu, Wenbo Li, Amirali Amini, Bo Sun, Yakov Bart, Weimin Lyu, Jiri Gesi, Tian Wang, Jing Huang, et al. 2026b. Opera: A dataset of observation, persona, rationale, and action for evaluating llms on human online shopping behavior simulation. *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 43942–43960.

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

Xinwei Yang, Junyi Fan, Yuqing Liu, Jiaxuan Wang, Jiashuai Zhang, Hongru Liang, Wenqiang Lei, and Yao Song. 2026. Empathy in diversity: Personalized depression and anxiety therapy via dialogue state tracking and patient-aware planning. In *Annual Meeting of the Association for Computational Linguistics*, pages 9370–9416. Association for Computational Linguistics.

Haobo Zhang, Yutao Zhu, Kelong Mao, Tianhao Li, and Zhicheng Dou. 2026a. Recthinker: An agentic framework for tool-augmented reasoning in recommendation. *arXiv.org*.

Junjie Zhang, Yupeng Hou, Ruobing Xie, Wenqi Sun, Julian McAuley, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2024. Agentcf: Collaborative learning with autonomous language agents for recommender systems. In *Proceedings of the ACM Web Conference 2024*, pages 3679–3689.

Zhiyang Zhang, Junda She, Kuo Cai, Bo Chen, Shiyao Wang, Xinchen Luo, Qiang Luo, Ruiming Tang, Han Li, Kun Gai, et al. 2026b. Unleashing the native recommendation potential: Llm-based generative recommendation via structured term identifiers. *Findings of the Association for Computational Linguistics: ACL 2026*, pages 19659–19673.

Kaiyan Zhao, Zijie Meng, Zheyong Xie, Jin Duan, Yao Hu, Zuozhu Liu, and Shaosheng Cao. 2026. Ecomstage: Stage-wise and orientation-specific benchmarking for large language models in e-commerce. *arXiv.org*.

Keyu Zhao, Fengli Xu, and Yong Li. 2025. Reason-to-recommend: Using interaction-of-thought reasoning to enhance llm recommendation. *arXiv.org*.

Jiayin Zhu, Kelong Mao, Yudong Guo, Dengbo He, Sulong Xu, Simiu Gu, and Yutao Yue. 2026. Skillcoach: Self-evolving rubrics for evaluating and enhancing agentic skill-use. *arXiv*.

Yaochen Zhu, Harald Steck, Dawen Liang, Yinhan He, Nathan Kallus, and Jundong Li. 2025a. LLM-based conversational recommendation agents with collaborative verbalized experience. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pages 2207–2220, Suzhou, China. Association for Computational Linguistics.

Zilin Zhu, Chengxing Xie, Xin Lv, and slime Contributors. 2025b. slime: An llm post-training framework for rl scaling. `https://github.com/THUDM/slime`. GitHub repository. Corresponding author: Xin Lv.

A Additional Results

This appendix provides additional empirical analyses that complement the main findings in Section 4.2. Figures 2, 3, 4, and 5 provide fine-grained diagnostic slices over early-exit behavior, interaction phenomena, scenario types, and user interaction styles. Together, these results show that the turn-to-session gap is not confined to a single subset, but consistently emerges when agents must preserve evolving state, revise constraints, coordinate multiple subgoals, and converge under catalog-grounded decision pressure.

Early-exit behavior. Figure 2 compares early-exit and non-exit sessions across session-level process dimensions. The results show that early exits are associated with broad process degradation, but the largest drops occur in state tracking, constraint updating, and decision pacing. This suggests that user abandonment is not mainly caused by the absence of clarification alone. Instead, users are more likely to leave when the agent fails to remember previously revealed information, update the active constraint set, or make timely progress toward a grounded recommendation.

Interaction phenomena. Figure 3 analyzes performance across different interaction phenomena. Revision-heavy phenomena, including constraint updates, preference overrides, recommendation rejections, and bundle re-planning, are consistently more challenging than simpler refinement or comparison cases. These phenomena require the agent to propagate new user feedback through the maintained session state, candidate set, and downstream recommendation plan. The observed performance drops therefore indicate that current shopping agents remain fragile when the dialogue requires explicit state revision rather than one-shot preference matching.

Scenario complexity. Figure 4 further shows that scenario complexity amplifies the mismatch between local response quality and session-level shopping success. Agents perform best in explicit single-item sessions, where the target is directly searchable and the amount of state revision is limited. In contrast, ambiguous, bundle, multi-intent, and long-horizon scenarios introduce hidden constraints, subgoal dependencies, and longer-range dialogue memory requirements. As a result, locally plausible responses become less predictive of successful shopping trajectories as the session requires more sustained decision control.

User interaction styles. Figure 5 evaluates how different user styles affect session robustness. Collaborative or patient users allow agents more opportunities to recover from incomplete clarification or imperfect recommendations. By contrast, efficiency-first, high-skepticism, and low-patience users expose whether the agent can quickly identify blocking constraints, avoid redundant questioning, verify candidates, and converge without repeated mistakes. The higher early-exit rates and larger turn-to-session gaps under these styles show that realistic shopping-agent evaluation should include convergence pressure rather than assuming uniformly cooperative users.

Diagnosis-oriented agent comparison. Table 4 summarizes how REALSHOP_AGENT addresses the diagnostic failure modes identified above. Compared with the prompt-only backbone and the adapted backbone, the full agent achieves consistent improvements on early-exit-prone episodes, revision-heavy interaction phenomena, complex scenarios, and demanding user profiles. These results show that backbone adaptation is helpful but insufficient on its own: robust session-level shopping behavior also requires explicit state management, action timing, catalog-grounded verification, and runtime guards against stale or unsupported recommendations.

B Comparison with Existing Benchmarks and Methods

Benchmarking conversational shopping agents. Existing benchmarks for conversational shopping agents can be broadly grouped into two lines. The first line is *outcome-oriented* evaluation, which is rooted in conversational recommendation. These benchmarks typically evaluate whether a system can infer user preferences and return relevant final items, often using metrics such as recommendation accuracy, preference satisfaction, explanation quality, or user-perceived utility. Recent work has improved this line of evaluation by scaling up dialogue data, grounding users in behavioral traces, introducing controllable user simulators, or expanding the set of acceptable target items (Liang et al., 2024; Chhetri et al., 2025; Huang et al., 2024; Kim et al., 2025a; Vlachou, 2025; Chen et al., 2025; Huang et al., 2026). For example, LLM-

Finding Source	Diagnostic Subset	Metric	Base	+SFT/RL	Full Agent	Δ vs. Base
Figure 2: Early-exit sessions reveal weak session-level process control						
Fig. 2	Overall sessions	Early Exit Rate ↓	0.25	0.20	0.09	0.16 ↓
Fig. 2	Early-exit-risk episodes	Sess. Avg. ↑	0.60	0.79	1.21	0.61 ↑
Fig. 2	Early-exit-risk episodes	State Tracking ↑	0.51	0.75	1.33	0.82 ↑
Fig. 2	Early-exit-risk episodes	Constraint Updating ↑	0.47	0.78	1.24	0.77 ↑
Figure 3: Interaction phenomena stress state revision and trajectory management						
Fig. 3	Constraint refinement	Success Rate ↑	0.63	0.70	0.88	0.25 ↑
Fig. 3	Product comparison	Success Rate ↑	0.62	0.69	0.85	0.23 ↑
Fig. 3	Constraint update	Success Rate ↑	0.55	0.64	0.83	0.28 ↑
Fig. 3	Preference override	Success Rate ↑	0.51	0.58	0.75	0.24 ↑
Fig. 3	Recommendation rejection	Success Rate ↑	0.50	0.55	0.78	0.28 ↑
Fig. 3	Bundle re-planning	Success Rate ↑	0.54	0.63	0.80	0.26 ↑
Figure 4: Complex scenarios amplify the turn-to-session quality gap						
Fig. 4	Explicit single-item	Success Rate ↑ / Gap ↓	0.65 / 0.19	0.75 / 0.16	0.91 / 0.04	0.26 ↑ / 0.15 ↓
Fig. 4	Explicit bundle	Success Rate ↑ / Gap ↓	0.63 / 0.20	0.73 / 0.18	0.86 / 0.07	0.23 ↑ / 0.13 ↓
Fig. 4	Ambiguous single-item	Success Rate ↑ / Gap ↓	0.60 / 0.32	0.66 / 0.29	0.85 / 0.10	0.25 ↑ / 0.22 ↓
Fig. 4	Ambiguous bundle	Success Rate ↑ / Gap ↓	0.50 / 0.34	0.60 / 0.31	0.74 / 0.11	0.24 ↑ / 0.23 ↓
Fig. 4	Multi-intent shopping	Success Rate ↑ / Gap ↓	0.49 / 0.35	0.57 / 0.27	0.76 / 0.07	0.27 ↑ / 0.28 ↓
Fig. 4	Long-horizon session	Success Rate ↑ / Gap ↓	0.48 / 0.36	0.63 / 0.24	0.72 / 0.09	0.24 ↑ / 0.27 ↓
Figure 5: Demanding interaction styles increase early exits						
Fig. 5	Collaborative	Early Exit Rate ↓	0.18	0.17	0.05	0.13 ↓
Fig. 5	High patience	Early Exit Rate ↓	0.12	0.11	0.04	0.08 ↓
Fig. 5	Low skepticism	Early Exit Rate ↓	0.19	0.14	0.08	0.11 ↓
Fig. 5	Efficiency-first	Early Exit Rate ↓	0.27	0.25	0.12	0.15 ↓
Fig. 5	High skepticism	Early Exit Rate ↓	0.30	0.26	0.15	0.15 ↓
Fig. 5	Low patience	Early Exit Rate ↓	0.34	0.28	0.10	0.24 ↓

Table 4: Finding-oriented diagnostic results of REALSHOP_AGENT. Rows are aligned with the failure modes analyzed in Figures 2–5. Base denotes the prompt-only Qwen3.5-27B backbone, +SFT/RL denotes the adapted backbone without the full execution harness, and Full Agent denotes REALSHOP_AGENT with the complete execution harness. For Figure 4, Gap is computed as Turn Avg. minus Sess. Avg. Δ reports the absolute improvement of Full Agent over Base; for paired metrics in Figure 4, the two gains are reported in the same order as the metric column.

ReDial constructs conversational recommendation data from user behaviors, CONCEPT emphasizes both system-centric and user-centric factors, target-free user simulation reduces dependence on pre-specified target items, and RecUserSim improves the realism and diversity of simulated users. These benchmarks are useful for studying preference elicitation and final-item relevance, but they often abstract away the evolving decision process that occurs during a full shopping session. In particular, they provide limited visibility into whether an agent can maintain hidden constraints, revise earlier assumptions, coordinate multiple subgoals, recover from user rejections, and decide when to clarify, compare, recommend, or confirm.

The second line is *execution-oriented* evaluation, which moves closer to realistic shopping environments by testing agents in simulated or web-based e-commerce settings. These benchmarks evaluate whether agents can perform shopping-related actions such as searching, browsing, filtering, comparing items, using tools, or completing task stages under explicit instructions (Liu et al., 2026; Lyu et al., 2025; Zhao et al., 2026; Wang et al., 2026b). For instance, recent benchmarks study personalized shopping assistants, simulated recommender worlds, deep research shopping agents, stage-wise e-commerce tasks, and human online shopping behavior simulation. This line of work is important because it brings evaluation beyond static recommendation and into action-oriented shopping environments. However, the main focus is often whether an external action is completed or whether a task stage is executed correctly, rather

Benchmark	Eval. Focus	Real Product Inventory	User Simulator	Full Trajectory	State / Constraint Eval.	Complex Scenarios	Session-Level Process Eval.
LLM-REDIAL (Liang et al., 2024)	Outcome-oriented	✗	△	△	✗	✗	✗
CONCEPT (Huang et al., 2024)	Outcome-oriented	✗	△	△	△	△	△
PEPPER (Kim et al., 2025a)	Outcome-oriented	✗	✓	△	△	△	△
ConvRecStudio (Chhetri et al., 2025)	Outcome-oriented	✗	△	△	✗	△	✗
Fashion-AlterEval (Vlachou, 2025)	Outcome-oriented	△	△	△	△	△	✗
RecUserSim (Chen et al., 2025)	Outcome-oriented	✗	✓	△	△	△	△
RecoWorld (Liu et al., 2026)	Execution-oriented	△	△	✓	△	△	△
DeepShop (Lyu et al., 2025)	Execution-oriented	✓	✗	△	△	△	✗
OPeRA (Wang et al., 2026b)	Execution-oriented	△	✓	△	✗	△	✗
RecBench+ (Huang et al., 2026)	Outcome-oriented	△	✗	✗	△	△	✗
ShoppingBench (Wang et al., 2026a)	Execution-oriented	✓	✗	△	△	△	✗
EComStage (Zhao et al., 2026)	Execution-oriented	△	✗	△	△	△	△
REALWORLDSHOP (Ours)	**Session Process**	✓ **(3.28M SKUs)**	✓	✓	✓	✓	✓

Table 5: Comparison with representative benchmarks for conversational shopping agents. **Eval. Focus** is grouped into three categories: *Outcome-oriented*, *Execution-oriented*, and *Session Process*. **Real Product Inventory** indicates whether the benchmark is grounded in a real product/SKU inventory; REALWORLDSHOP contains 3.28M deduplicated products. ✓indicates explicit support, △indicates partial or indirect support, and ✗indicates limited support. Compared with prior benchmarks, REALWORLDSHOP jointly supports grounded inventory, profile-grounded and action-controlled user simulation, full multi-turn trajectories, state and constraint evaluation, complex shopping scenarios, and session-level process evaluation.

than whether the agent manages the conversational decision process behind those actions. In real-world shopping, the central challenge is not only to retrieve or recommend an item, but also to elicit hidden needs, track and revise evolving constraints, coordinate dependent subgoals, ground product claims in catalog evidence, and guide the user toward a grounded and actionable purchase decision. REALWORLDSHOP addresses this gap by evaluating full multi-turn shopping trajectories with explicit assessment of clarification, state tracking, constraint updating, decision pacing, personalization, multi-intent coordination, and grounded convergence. A detailed comparison with representative benchmarks is provided in Table 5.

Improving conversational shopping agents. Recent methods improve conversational shopping agents from several complementary directions, including user modeling, strategic action selection, feedback-loop modeling, tool-augmented reasoning, reinforcement learning, and reasoning-enhanced recommendation. PersonaX retrieves multi-persona profiles from long behavior sequences to provide richer user representations for downstream recommendation agents (Shi et al., 2025). CSI performs contextual user profiling for conversational sales agents and selects actions across preference elicitation, recommendation, explanation, and persuasion (Kim et al., 2025b). AFL models an agentic feedback loop between recommendation agents and user agents, using iterative feedback and memory to improve both recommendation and user simulation (Cai et al., 2025). RecThinker adopts an Analyze-Plan-Act framework with specialized tools for evidence acquisition and recommendation reasoning, while ChatShopBuddy studies reinforcement-learning-based optimization under multi-dimensional shopping rewards (Zhang et al., 2026a; Cheng et al., 2026). Other frameworks, such as R2Rec and GRAM, incorporate explicit reasoning traces or structured item identifiers to improve the reasoning and generation capabilities of LLM-based recommenders (Zhao et al., 2025; Zhang et al., 2026b).

These methods provide useful building blocks for more capable shopping agents, but most of them mainly strengthen individual capabilities rather than the full session-control loop. User-profiling methods improve personalization, but they do not by themselves determine how the agent should update state after a new constraint or rejection. Strategic action methods improve high-level interaction planning, but they may still lack explicit mechanisms for stale-state detection, constraint invalidation, or catalog-faithful verification. Tool-augmented reasoning methods improve evidence acquisition, but retrieved evidence can still become obsolete when the user changes budget, compatibility requirements, delivery constraints, or bundle composition. Reinforcement-learning methods

improve policy behavior under task rewards, but without explicit runtime structure they may still produce premature recommendations, redundant tool calls, or unsupported SKU-level claims. Thus, existing methods are complementary to our work, but they do not fully address executable session-level control, where an agent must jointly manage clarification, evolving state, multi-intent coordination, grounded retrieval, verification, and timely commitment within a single shopping trajectory.

REALSHOP_AGENT focuses on this missing session-control perspective. It treats conversational shopping as a state-driven decision process rather than as isolated response generation or one-shot recommendation. The Session State Manager maintains revealed intents, active subgoals, constraints, rejected options, confirmed preferences, and stale dependencies; the Shopping-Flow Controller decides when to clarify, retrieve, verify, compare, recommend, or confirm; the Catalog-Grounded Retriever connects the current state to product evidence; and Runtime Execution Guards prevent over-retrieval, unsupported SKU-level claims, and recommendations based on stale constraints. In this sense, REALSHOP_AGENT is not a replacement for prior advances in profiling, reasoning, feedback modeling, or reinforcement learning, but an executable framework that integrates these capabilities into a controlled shopping trajectory.

C Additional Details of the REALWORLDSHOP Benchmark

This appendix provides additional details of the REALWORLDSHOP benchmark, including product inventory construction, episode generation, profile synthesis, simulator design, and the role-play evaluation protocol. Compared with the main text, the goal here is to make the benchmark construction procedure more explicit and reproducible.

C.1 Product Inventory

Category-driven inventory construction. We construct the product inventory from category-driven crawl queries over a large real-world e-commerce platform. Starting from more than one thousand category and subcategory seeds, we retrieve product results in batches, merge the returned items across seeds, and retain only valid deduplicated SKUs. This design avoids over-reliance on raw crawl pages and ensures that the final inventory is organized around a stable product identifier space rather than noisy page-level results.

Rule-based filtering. The platform's raw category space contains many entries that are not suitable for conversational shopping over shippable physical goods, including services, finance, digital-only products, local-life offerings, and travel-related entries. We therefore apply an additional rule-based filter to remove such categories before finalizing the pool. The purpose of this step is to align the benchmark with realistic shopping-agent behavior over purchasable physical goods rather than service booking or O2O workflows.

Inventory snapshot and deduplication. The inventory is treated as a fixed benchmark snapshot during evaluation. All systems are evaluated against the same product pool, catalog fields, and fulfillment metadata, so that differences in performance come from agent behavior rather than changing product availability. Deduplication is performed at the SKU level when stable product identifiers are available, and otherwise relies on normalized title, category path, brand, and key attribute fields. This snapshot-based design improves reproducibility, although dynamic platform factors such as price changes, stock updates, promotions, and temporary availability changes are not modeled during evaluation.

Inventory scale and coverage. The final inventory contains **3.28M deduplicated products**, spanning a broad range of major product groups. This scale is important for two reasons. First, it enables realistic diversity of shopping intents, including both single-product and multi-product sessions. Second, it makes the benchmark meaningfully different from toy inventories or narrowly scoped recommendation corpora, where candidate spaces are artificially small and decision difficulty is underestimated.

Catalog fields. Each product record retains rich catalog information used by our agent tools and benchmark construction pipeline. At a minimum, this includes price, attributes, category labels, product descriptions, and fulfillment signals such as free shipping, next-day delivery, and same-day delivery. In practice, the retained record also preserves additional platform-side metadata that supports grounded retrieval and post-hoc analysis. This field richness is crucial because it allows shopping episodes to be grounded in a catalog-faithful evi-

dence space rather than a synthetic schema with manually simplified attributes.

C.2 Episode Construction

An evaluation instance in REALWORLDSHOP is a **structured shopping episode**

$$\mathcal{E} = \langle \mathcal{P}, \mathcal{S}, \Pi \rangle,$$

consisting of a user profile $\mathcal{P}$, a shopping scenario $\mathcal{S}$, and a session plan Π. This design departs from conventional one-shot benchmark construction, where a user query is paired with a single answer. Instead, our episode structure explicitly models shopping as a multi-turn interaction between a partially observable user state and a decision-making shopping agent.

Category-anchored two-stage synthesis. We construct episodes through a category-anchored two-stage synthesis pipeline rather than generating shopping interactions from scratch. The purpose of anchoring episode construction to category batches is to make generated shopping sessions both realistic and analyzable: the benchmark can later be sliced by scenario type, constraint type, interaction phenomenon, and product-category distribution.

Stage 1: shopping scenario synthesis. In the first stage, a batch of product category queries and their source paths is mapped to a natural-language shopping scenario. To improve coverage and controllability, we introduce a two-level scenario taxonomy:

- **Task archetypes**, which specify the high-level shopping structure, such as building a bundle from scratch, upgrading a core item, replacement or repair, scenario-based bundle purchase, purchasing for others, consumable restocking, vague exploration, mixed errands, and urgency-driven purchase.
- **Constraint modifiers**, which specify additional decision pressure or complexity, such as compatibility sensitivity, budget trade-off, high-risk decision, logistics or time sensitivity, and bundle-splitting decisions.

Conditioned on these two levels, the generator produces three outputs: a shopping scenario, a set of emphasized categories, and an initial user-utterance hook. The shopping scenario provides the natural-language situational context of the episode; the emphasized categories highlight the main category anchors that should remain salient during later profile construction; and the utterance hook provides the simulator with a plausible opening turn.

Session plan construction. The scenario is then paired with a session plan Π, which specifies how the shopping interaction is expected to unfold. This includes whether the session is explicit-intent or ambiguous-intent, whether it contains single or multiple shopping subgoals, and which constraints are likely to be revealed, tightened, or updated over the course of the conversation. In this way, episode construction determines not only what the user is shopping for, but also the expected interaction dynamics of the session.

Diagnostic slicing metadata. Each episode is annotated with metadata used for diagnostic analysis, including scenario type, interaction phenomenon, constraint type, and interaction-profile attributes. Scenario types distinguish explicit single-item, explicit bundle, ambiguous single-item, ambiguous bundle, multi-intent shopping, and long-horizon sessions. Interaction phenomena cover constraint refinement, product comparison, constraint update, preference override, recommendation rejection, and bundle re-planning. These labels are assigned during episode construction rather than inferred from model outcomes, which prevents diagnostic subsets from being biased by a particular agent's realized behavior.

C.3 User Profile Construction

Structured profile synthesis. In the second stage, each shopping scenario is converted into a structured user profile. Rather than directly generating free-form personas, we map each scenario into a profile schema that can be consumed by the simulator and by the benchmark slicing pipeline. The schema includes shopping-relevant background information, long-term preferences, historical shopping behavior, current purchase intent, hard, soft, and negotiable constraints, interaction style, and optional multi-intent planning fields.

Scenario-to-profile mapping. The scenario archetype determines the overall task structure of the profile, while the constraint modifiers shape the resulting constraints. For example, mixed errands and scenario-bundle settings encourage multi-intent profiles with purchase-intent subgoals and multi-intent plans; urgent timing and logistics-sensitive settings increase urgency level and de-

livery constraints; compatibility-sensitive settings strengthen compatibility-related hard constraints; and budget trade-off settings encourage a clearer separation between hard, soft, and negotiable budget preferences. This mapping is important because it makes profile construction systematic rather than stylistic.

Visible and latent profile fields. For each profile, we separate the information available to the agent from the latent information used only by the simulator. The visible prior $\widetilde{P}$ contains stable shopping-relevant background and broad preferences, while the shopping scenario S is provided separately through the episode context and opening interaction. Latent profile fields contain hidden intents, unrevealed constraints, preference updates, rejection triggers, and disclosure plans. This separation makes the benchmark partially observable: agents cannot access the complete profile or session plan, but must progressively elicit and update the relevant state through interaction.

Explicit-intent versus ambiguous-intent profiles. We further distinguish between *explicit-intent* and *ambiguous-intent* profiles. In explicit-intent profiles, the initial user goal is relatively searchable and exposed in the opening interaction. In ambiguous-intent profiles, the opening utterance reveals only a scenario, problem, or recipient, while the true goal is stored in latent fields and disclosed gradually through a planned disclosure schedule. This design introduces partial observability into the benchmark and enables evaluation of both direct product-search ability and clarification-driven conversational shopping.

Catalog grounding during profile synthesis. To reduce hallucinated or unverifiable constraints, profile synthesis can be grounded in the local product catalog. For each scenario, we retrieve a small evidence pack from the product inventory, including product titles, brands, attributes, prices, and fulfillment tags. The generator is instructed to use this evidence when instantiating concrete hard constraints, and to downgrade unsupported details into soft or negotiable preferences. This grounding mechanism improves realism while reducing brittle or impossible constraints that cannot be verified against the available catalog.

Programmatic validation. After synthesis, all profiles are normalized and validated programmatically. Validation includes enum validation, multi-intent consistency checks, ambiguous-intent field checks, category anchoring, and catalog-grounding checks. Profiles that violate the schema, contain inconsistent subgoal structures, or fail grounding checks are rejected or regenerated. This procedure ensures that the resulting profiles are valid, reproducible, and suitable for controlled slice-based analysis.

Interaction-profile factors. Each user profile contains interaction-profile factors that describe how the user communicates, reacts, and makes decisions during a shopping session. These factors are separated from purchase intent and product constraints: they determine the user's conversational behavior rather than the target item itself. Table 6 summarizes the factors and value ranges.

Together, these factors allow the simulator to generate diverse interaction behaviors, such as skeptical evidence-seeking users, low-patience users who exit early, tradeoff-seeking users who negotiate constraints, and comparison-oriented users who request multiple alternatives. This design enables controlled stress testing of clarification, evidence presentation, decision pacing, comparison handling, constraint updating, and early-exit prevention.

C.4 Simulator Design and Validation

Action-controlled simulation. The simulator follows an action-first generation process. At each user turn, it first selects a user action from the session plan and dialogue state, such as providing information, refining a constraint, rejecting a candidate, requesting comparison, confirming a decision, shifting intent, or ending the session. It then realizes the selected action as a natural-language utterance conditioned on the interaction-profile factors. This design improves controllability because behavioral changes are represented at the action level before surface wording is generated.

Profile faithfulness. We validate whether simulated user utterances remain faithful to the structured profile and session plan. The simulator should not reveal latent information before the planned disclosure point, contradict hard constraints, ignore specified interaction style, or introduce unsupported shopping goals. This check ensures that the benchmark difficulty comes from controlled partial observability and evolving constraints rather than uncontrolled simulator drift.

Factor	Values
Coping style	problem-focused, emotion-focused, avoidant, assertive, compliant
Interaction style	directive, collaborative, browsing, interrogative, efficiency-first, tradeoff-seeking
Verbosity	low, medium, high
Politeness	direct, neutral, warm
Skepticism	low, medium, high
Decisiveness	low, medium, high
Patience	low, medium, high
Comparison habit	rarely-compare, compare-key-options, compare-extensively
Purchase-intent divergence	non-divergent, related-divergence-allowed, broad-divergence-allowed

Table 6: Interaction-profile factors used to control simulated user behavior. These factors affect how users communicate, compare options, tolerate clarification, react to recommendations, and decide whether to continue or exit a shopping session.

Action controllability. We also check whether generated utterances realize the intended action labels. For example, a rejection action should express dissatisfaction or refusal of the current candidate; a refinement action should add or tighten a constraint; a comparison action should request trade-off information; and a confirmation action should indicate readiness to proceed. These checks are used to verify that action-controlled simulation produces interpretable and reproducible interaction dynamics.

C.5 Evaluation Protocol

We evaluate conversational shopping systems through a role-play-based, session-level protocol. Each evaluation episode is instantiated from a structured shopping configuration $\mathcal{E}$, and the system under evaluation interacts with the simulator over multiple turns rather than being assessed from a single static input.

Role-play setup. For each episode, Kimi-K2.6 plays the role of the customer, conditioned on the structured episode configuration and the dialogue history. The shopping agent—the system under evaluation—observes only the conversation history and the visible prior $\tilde{\mathcal{P}}$, and must progressively infer the hidden session intent and constraints through interaction. The agent slot is model-agnostic: in our experiments, we instantiate the same protocol with both general LLM baselines and specialized shopping-agent methods under the same episode configuration, visible prior, and judging pipeline.

Turn-by-turn interaction. Conversations unfold turn by turn. At each turn, the simulator first produces a customer response under its action-first policy, and the agent then replies. The action-first design is important because it decouples latent user behavior from surface realization: the simulator first decides what the user is doing, such as providing information, refining a constraint, rejecting a candidate, requesting comparison, confirming, or ending, and only then verbalizes that action in natural language.

Termination taxonomy. A session terminates in exactly one of three states:

- SUCCESS: the interaction reaches a grounded and actionable purchase decision;
- EARLY-EXIT-WITHOUT-CONVERGENCE: the user ends the session before committing;
- MAX-TURN FAILURE: the maximum of 30 assistant turns is reached.

This taxonomy separates voluntary abandonment from capacity-limited timeouts, making the benchmark more diagnostic than a single binary success/-failure label.

Three-level evaluation. Our evaluation framework is organized into three complementary levels: *turn-level output quality*, *session-level process quality*, and *final outcome quality*. This separation is necessary because systems can perform well on one level while failing on another. For example, an agent may generate locally plausible recommendations at individual turns, yet fail to maintain a coherent shopping trajectory or converge to an actionable decision.

Three-stage LLM-as-judge. After each episode, we apply a three-stage GPT-5.5 judge to the dialogue, tool trace, and episode configuration. The three stages correspond directly to the three evaluation levels above (Tian et al., 2026b, 2025a).

Stage 1: turn judge. For each assistant turn, the judge scores local response quality on a $\{0, 1, 2\}$ scale. It evaluates:

- **Need Understanding**: whether the assistant correctly understands the user's current request or newly introduced changes;
- **Recommendation Accuracy**: if products are recommended, whether they are appropriate given the currently known requirements, factually accurate, and grounded in available evidence;
- **Recommendation Rationale Quality**: if products are recommended, whether the recommendation reasons are specific, helpful, and aligned with the user's needs.

Recommendation-related dimensions are marked as N/A for non-recommendation turns and are excluded from averaging for that dimension.

Stage 2: session judge. The judge then scores the full dialogue trajectory on five session-level dimensions, again on a $\{0, 1, 2\}$ scale:

- **Clarification**: whether the agent identifies and asks about key missing information before reliable recommendation;
- **State Tracking**: whether the agent preserves previously revealed user needs, constraints, and rejected candidates across the session;
- **Constraint Updating**: whether the agent revises its behavior after the user updates or overrides earlier constraints;
- **Decision Pacing**: whether the agent chooses the right next action at the right time, including when to clarify, retrieve, verify, compare, recommend, or confirm, without premature commitment, excessive delay, or redundant tool use;
- **Personalized Recommendation**: whether the final or near-final recommendations reflect the user's profile, session-specific needs, and progressively revealed information.

Tool-use quality is not scored as a standalone top-level dimension. Instead, tool timing and redundancy are absorbed into **Decision Pacing**, while recommendation grounding is absorbed into **Recommendation Accuracy**. Similarly, preference override is not treated as a separate top-level metric; it is operationalized through **Constraint Updating** and **Personalized Recommendation**.

Stage 3: holistic judge. Finally, the judge emits the overall session outcome: overall_success $\in \{\text{success}, \text{partial}, \text{failed}\}$, together with an optional diagnostic label convergence_quality $\in \{\text{strong}, \text{moderate}, \text{weak}, \text{none}\}$. We ordinalize convergence_quality as none = 0, weak = 1, moderate = 2, and strong = 3. These labels capture whether the interaction ultimately converges to a usable purchase decision and how cleanly that convergence is achieved.

Metric aggregation. For each episode, Turn Avg. is computed by averaging applicable turn-level dimensions across assistant turns, excluding N/A recommendation dimensions from their corresponding denominators. Sess. Avg. is the average of the five session-level process scores. Gap is defined as Turn Avg. minus Sess. Avg., measuring the mismatch between local response quality and trajectory-level shopping competence. Succ. reports the fraction of episodes judged as successful by the holistic judge. Conv. averages the ordinalized convergence-quality score across episodes. Early Exit reports the fraction of sessions terminated as EARLY-EXIT-WITHOUT-CONVERGENCE by the simulator.

Judge prompt design. Judge prompts are designed to improve scoring consistency in several ways. First, they explicitly separate content quality from timing quality: recommendation accuracy evaluates whether a recommendation is correct, while decision pacing evaluates whether it is made at the right time. Second, turn-level prompts support N/A outputs for non-applicable recommendation dimensions. Third, all stages return structured JSON outputs with scores, evidence turns, and short justifications. Finally, the scoring process is guided by small internal checklists rather than relying solely on a free-form global impression.

Judge-side compaction. To keep judge prompts within token limits without biasing scores, we apply deterministic compaction to the tool trace before passing it to the judges, truncating verbose fields such as comment statistics and capping long attribute or tag lists. The full trace remains available in the logged artifact for downstream analysis.

Robustness and episode status. Each LLM call, including simulator, agent, and judge calls, is wrapped in a 5-retry exponential-backoff loop to absorb transient endpoint errors. Episodes are categorized by a three-valued episode_status:

- VALID-SCORED-EPISODE: simulation and all judge stages succeed;
- JUDGE-PARSE-FAILURE: simulation succeeds

but at least one judge stage returns an unparseable output;

- GENERATION-FAILURE: a simulator or agent call fails after retries.

All aggregate metrics are reported over valid-scored episodes, with failure counts listed separately for reliability checking.

Final reporting. The benchmark reports turn-level averages, session-level capability scores, and holistic outcome labels as complementary signals. This makes it explicit that a system may produce locally strong turns while still failing to manage the full shopping trajectory toward grounded convergence.

Why role-play evaluation. We adopt a role-play, session-level protocol because the target abilities are trajectory-level by definition. Need understanding and recommendation quality can be assessed locally, but clarification, state tracking, constraint updating, decision pacing, personalization, and final convergence only become observable through a full multi-turn interaction. Taken together, the benchmark is *realistic* because it is grounded in a real catalog and validated user profiles, *controllable* because each episode is defined by a structured configuration and driven by an action-controlled simulator, and *diagnostic* because it separates local response quality, trajectory-level process quality, and final decision outcomes.

C.6 Automatic Judge Reliability

Reliability of automatic evaluation. We assess judge reliability through human validation and repeated-judging analysis. We stratify 150 sessions across model families, scenario types, and outcomes, and have three human annotators independently score them using the same evaluation rubrics as the automatic judge. For ordinal dimensions, we report quadratic weighted kappa (QWK), mean absolute error (MAE), and exact agreement between aggregated human annotations and automatic judgments; outcome labels are evaluated using the corresponding agreement-based statistics.

We further repeat automatic judging for three independent passes and measure stability using ICC(2,1), mean score standard deviation, and label flip rate. Together, these analyses evaluate both human alignment and robustness to judge sampling variance.

Judgment Target	QWK / κ	MAE	Agreement
Need Understanding	0.82	0.07	0.95
Recommendation Accuracy	0.81	0.09	0.91
Rationale Quality	0.81	0.13	0.90
Clarification	0.82	0.11	0.93
State Tracking	0.86	0.07	0.94
Constraint Updating	0.89	0.06	0.96
Decision Pacing	0.76	0.12	0.89
Personalization	0.83	0.08	0.92
Overall Success	0.86	0.03	0.97
Convergence Quality	0.81	0.19	0.88

Table 7: Human validation of the automatic judging protocol. For ordinal score dimensions, the first column reports quadratic weighted kappa (QWK). For outcome labels, it reports agreement-based κ when applicable. MAE denotes mean absolute error, and Agreement denotes exact agreement with aggregated human annotations.

Metric Group	ICC(2,1)	Mean Std.	Flip Rate
Turn-level Quality	0.86	0.09	0.03
Session-level Process Quality	0.87	0.08	0.02
Overall Success	0.92	0.04	0.02
Convergence Quality	0.88	0.11	0.04

Table 8: Stability of the automatic judging protocol across repeated judging passes. We report ICC(2,1), mean standard deviation, and label flip rate to assess whether the main trends are robust to judge sampling variance. Flip Rate denotes the proportion of sessions whose final discrete label changes across repeated judging passes; for score-based dimensions, scores are first mapped to their nearest discrete rubric value.

D Simulator Validation

Since REALWORLDSHOP relies on role-play evaluation, the reliability of the user simulator is central to the validity of the benchmark. We therefore validate Kimi-K2.6 along two complementary axes: *profile faithfulness* and *action controllability*. Specifically, we randomly sample 200 user profiles from REALWORLDSHOP and evaluate whether the simulator can faithfully realize the corresponding user intents, constraints, disclosure plans, and interaction-profile factors during multi-turn role-play. To complement automatic validation, three human annotators manually audit the sampled simulated sessions along naturalness, profile faithfulness, action faithfulness, goal consistency, and hidden-intent control. The simulator is not used as an unconstrained role-player. Instead, at each user turn, it is conditioned on a structured user profile, hidden session intent, disclosure plan, interaction-profile factors, dialogue history, and an explicit target user action. It first follows the intended behavioral action and then realizes that

action as a natural-language user utterance.

Profile-level validation. We evaluate whether simulated users remain consistent with their structured profiles throughout the session. Specifically, we randomly sample 200 user profiles and check six aspects: (1) *long-term preference consistency*, whether generated utterances preserve stable user preferences; (2) *hard-constraint consistency*, whether hard constraints such as budget, compatibility, logistics, and risk tolerance are respected; (3) *soft-constraint alignment*, whether soft or negotiable preferences are reflected without being over-enforced as strict requirements; (4) *disclosure-plan compliance*, whether session-specific intents and constraints are revealed at the intended stages; (5) *hidden-intent control*, whether ambiguous-intent profiles avoid prematurely leaking latent goals or hidden constraints; and (6) *interaction-style fidelity*, whether behavioral factors such as skepticism, patience, decisiveness, and comparison orientation are reflected in the dialogue. As shown in Table 9, Kimi-K2.6 achieves an average profile-level faithfulness score of 0.93, suggesting that the simulator can generally preserve structured profile information while following the intended disclosure and interaction-style controls.

Action-level validation. We further evaluate whether Kimi-K2.6 follows the action-controlled user policy. Each user turn is assigned an intended action, such as providing missing information, refining constraints, updating prior requirements, overriding long-term preferences, rejecting a recommendation, comparing candidates, confirming a decision, or ending the session. We measure *action compliance* by checking whether the generated utterance semantically matches the intended action, and *transition validity* by checking whether the action is appropriate under the current dialogue state. As shown in Table 10, Kimi-K2.6 achieves strong action-level controllability, with an average action-level score of 0.95 over simulated user turns from the 200 sampled validation profiles. The simulator obtains high scores on intended-action matching (0.94) and state-valid transitions (0.97), indicating that it generally follows the assigned user actions while maintaining coherent dialogue-state progression. It also performs well on information provision (0.95), constraint refinement and update (0.97), and recommendation feedback actions (0.98). The relatively lower score for confirm/end control (0.92) suggests that decision and termination behavior is slightly harder to control, but the overall results indicate that Kimi-K2.6 can reliably realize target user actions and support controlled multi-turn shopping evaluation.

E Implementation Details of REALSHOP_AGENT

Session State Manager. REALSHOP_AGENT maintains an explicit session state M_t throughout the dialogue. The state register stores revealed user intents, active subgoals, hard constraints, soft preferences, negotiable requirements, rejected products, confirmed preferences, and stale dependencies. This representation separates long-term preferences from session-specific constraints. When a user reveals a new requirement during the session, the session-specific constraint overrides the visible long-term prior. For example, a user may generally prefer premium products but later disclose a strict budget for the current purchase; in this case, the budget becomes an active and binding constraint for subsequent retrieval, verification, and recommendation.

At each turn, the Session State Manager updates M_t from the previous state M_{t-1}, the latest user utterance, and the visible prior $\tilde{\mathcal{P}}$. The update identifies newly revealed intents, revised constraints, rejected candidates, and preference overrides. In bundle and multi-intent sessions, the manager also tracks dependencies among subgoals. When a core item, scenario assumption, budget, compatibility requirement, or bundle anchor changes, dependent subgoals and candidate sets are marked as stale. Stale entries cannot be directly used for recommendation; they must first be re-clarified, re-retrieved, or re-verified. This mechanism prevents the agent from carrying obsolete assumptions across turns after user updates, rejections, or bundle re-planning.

Shopping-Flow Controller. The Shopping-Flow Controller maps the current state M_t to a high-level shopping action CLARIFY, RETRIEVE, VERIFY, COMPARE, RECOMMEND, and CONFIRM. It selects CLARIFY when required fields are missing or when a hard constraint remains ambiguous; RETRIEVE when the active subgoal has enough information for product search but lacks fresh catalog evidence; VERIFY when retrieved candidates exist but have not yet been checked against the active constraint stack; and RECOMMEND only when at least one candidate is supported by current evidence and satisfies the active constraints, or when

Validation Aspect	Metric	Score / Rate	Description
Long-term preference consistency	Preference consistency rate	0.96	Generated turns preserve stable user preferences in the profile.
Hard-constraint consistency	Violation-free rate	0.93	The simulator avoids violating hard constraints such as budget, compatibility, and logistics.
Soft / negotiable constraint consistency	Soft-constraint alignment rate	0.91	The simulator reflects soft preferences without treating them as strict requirements.
Disclosure-plan compliance	Planned-disclosure compliance rate	0.94	Session-specific intents and constraints are revealed according to the predefined disclosure plan.
Hidden-intent control	Non-leakage rate	0.88	Ambiguous-intent profiles avoid prematurely leaking latent goals or hidden constraints.
Interaction-style fidelity	Style-consistency score	0.93	Behavioral factors such as skepticism, patience, decisiveness, and comparison orientation are reflected in user behavior.
Overall profile faithfulness	Average profile-level score	0.93	Aggregate validation score over profile consistency, disclosure control, and interaction-style fidelity.

Table 9: Profile-level validation of the Kimi-K2.6 user simulator. All metrics are reported such that higher values indicate better simulator reliability. We evaluate whether simulated user turns remain faithful to structured profiles, follow the disclosure plan, avoid premature leakage of hidden intent, and reflect interaction-profile factors.

remaining uncertainty is explicitly disclosed to the user. COMPARE is selected when multiple verified candidates remain or explicit trade-off analysis is requested, while CONFIRM is selected when the user signals readiness to commit and the current recommendation or purchase plan satisfies the active constraints. This action interface makes the decision process inspectable and reduces uncontrolled switching between questioning, searching, verifying, and recommending.

Catalog-Grounded Retriever. The Catalog-Grounded Retriever treats retrieval as an evidence-producing step rather than a generic search call. Given the active subgoal and constraint stack, it returns structured evidence records $E_t = \{e_i\}$, where each record contains product identifiers, category labels, attributes, price information, fulfillment signals, relevance scores, and retrieval-stage metadata. When available, category, compatibility, budget, and logistics constraints are enforced during retrieval. The retriever also supports state-consistent grounding: when the active intent, hard constraint, or bundle anchor changes, previously retrieved evidence tied to the outdated state is invalidated. This ensures that downstream recommendations are grounded in evidence that matches the current user state rather than earlier assumptions.

Before a product can be recommended, the retrieved candidates are verified against the active constraints in M_t. Hard constraints are treated as blocking conditions, while soft and negotiable preferences are used for ranking, comparison, and explanation. If no candidate satisfies all hard constraints, the agent either retrieves again under adjusted conditions or explicitly communicates the limitation to the user. For bundle construction, verification is applied both at the item level and at the bundle level, ensuring that individual products are suitable and that the overall bundle remains compatible with the user's scenario, budget, and updated preferences.

Runtime Execution Guards. Runtime Execution Guards are applied after action selection, retrieval, and draft generation. TOOLBUDGET limits redundant retrieval and encourages the agent to reuse existing evidence when it remains valid. Once the retrieval budget is exhausted, the agent must either answer from available evidence or acknowledge uncertainty instead of issuing additional unsupported searches. FABRICATIONGUARD checks whether product-specific claims, such as price, attributes, availability, or compatibility, are supported by the latest evidence records. Unsupported SKU-level claims are rewritten into uncertainty-aware statements or removed from the response. STATECONSISTENCYGUARD blocks recommendations that rely on stale subgoals, rejected options, or superseded constraints. Together, these guards form the final consistency layer before the assistant response is returned.

Validation Aspect	Metric	Score / Rate	Description
Overall action compliance	Intended-action match rate	0.94	Generated user utterances semantically match the assigned target actions.
Transition validity	State-valid transition rate	0.97	The intended user action is appropriate given the current dialogue state and session progress.
Information provision	Provide-action compliance rate	0.95	When assigned to provide missing information, the simulator reveals the expected intent, preference, or constraint.
Constraint refinement and update	Refine / update compliance rate	0.97	The simulator correctly refines, updates, or overrides prior requirements according to the target action.
Recommendation feedback	Reject / compare compliance rate	0.98	The simulator follows target actions involving rejection, comparison requests, or feedback on recommended candidates.
Decision and termination control	Confirm / end compliance rate	0.92	The simulator confirms a decision or ends the session only when the target action specifies confirmation or termination.
Style-conditioned action behavior	Action-distribution alignment	0.93	Action patterns reflect interaction-profile factors, such as more comparisons for comparison-oriented users and more rejection or ending actions for low-patience users.
Overall action controllability	Average action-level score	0.95	Aggregate validation score over action compliance, transition validity, and style-conditioned action behavior.

Table 10: Action-level validation of the Kimi-K2.6 user simulator. All metrics are reported such that higher values indicate better simulator controllability. Percentage-based metrics are computed over simulated user turns from the 200 sampled validation profiles. We evaluate whether generated user utterances follow the intended target actions, satisfy dialogue-state transition constraints, and reflect scenario- and style-conditioned action patterns.

Backbone adaptation. The executable harness can be used with different backbone models. In our main implementation, we adapt a Qwen3.5-27B backbone to better follow tool-using shopping policies. The policy is initialized from supervised fine-tuning and further optimized with reinforcement learning over shopping rollouts. During training, the model interacts with the same runtime used at inference time, including catalog tools and execution guards. Rollouts produce dialogue trajectories, tool traces, and outcome signals, which are scored by task-specific graders. Before optimization, we filter groups dominated by infrastructure failures or zero reward variance, and then apply group relative policy optimization. This adaptation improves the base policy's ability to follow shopping-specific actions, such as asking useful clarifying questions, retrieving relevant evidence, revising plans after user feedback, and formulating grounded recommendations.

Complementarity between adaptation and executable control. Backbone adaptation and executable control address different sources of failure. Adaptation improves the model's learned shopping behavior, including action selection and response formulation under evolving user constraints. In contrast, the session-control harness imposes explicit structure on the interaction process, preventing the model from ignoring revised constraints, recommending from stale evidence, or making unsupported product claims. This complementarity explains why the full REALSHOP_AGENT outperforms both the prompt-only backbone and the adapted backbone without the complete control loop.

F Training Details

F.1 Training Setup

We adapt the Qwen3.5-27B shopping policy using supervised fine-tuning (SFT) followed by online reinforcement learning. The SFT stage initializes the model to follow the shopping-agent interaction format, invoke tools correctly, and generate concise grounded responses. We then optimize the policy with group relative policy optimization (GRPO) using trajectories generated by the current policy in the executable shopping runtime.

During RL training, each prompt is executed through the shopping-agent harness, which exposes the corresponding tools and produces a complete

Item	Value
SFT examples	10K
RL prompt seeds	2K
RL training prompts	1.5K
Held-out development prompts	0.5K
GRPO group size	8 rollouts per prompt
Max assistant turns per rollout	30

Table 11: Training scale for backbone adaptation.

tool-using trajectory. Task-specific graders assign scalar rewards based on trajectory-level behavior, including recommendation quality, constraint satisfaction, catalog grounding, tool use, and task completion. The training prompts are disjoint from the held-out REALWORLDSHOP evaluation episodes.

F.2 Training Tasks and Rewards

The training mixture contains commerce-related agent tasks covering search and recommendation, routing, service, and order-preview scenarios. These prompts serve as seeds for online rollout generation rather than supervised input–output pairs. The training runtime uses catalog and tool interfaces aligned with those used at evaluation time.

Each task-specific grader returns a scalar reward composed of normalized signals. At a high level, the reward is written as

$$r = r_{\text{task}} + r_{\text{ground}} + r_{\text{constraint}} + r_{\text{tool}} - p_{\text{invalid}},$$

where the active terms depend on the task type. Positive rewards encourage task completion, product relevance, constraint satisfaction, catalog grounding, and correct tool use, while penalties discourage unsupported product claims, invalid tool paths, redundant retrieval, and unsafe transaction behavior. Rewards are normalized within each prompt group before GRPO advantage computation.

F.3 GRPO Optimization

For each prompt, we sample a group of trajectories and compute group-relative advantages as

$$A_i = \frac{r_i - \text{mean}(\{r_j\}_{j=1}^{G})}{\text{std}(\{r_j\}_{j=1}^{G}) + \epsilon}.$$

The policy is optimized with the clipped GRPO objective using the configuration summarized in Table 12. Rollout groups dominated by infrastructure failures or with zero reward variance are excluded before optimization.

Hyperparameter	Value
Learning rate	1e-6
Weight decay	0.1
GRPO clipping	0.2
High clipping threshold	0.28
Global batch size	64
Samples per prompt	8
Rollout temperature	1.0
Gradient clipping	1.0
KL coefficient	0.00

Table 12: Main hyperparameters used for GRPO optimization. Additional implementation details follow the SLIME (Zhu et al., 2025b) training framework.

G Prompt Design

Profile Synthesis Prompt

You are generating a structured user profile for a multi-turn e-commerce shopping benchmark.

Goal. Create a realistic shopping user whose needs can unfold across a full conversation. The profile should support evaluation of clarification, state tracking, constraint updating, multi-intent coordination, bundle/split-order planning, and grounded recommendation.

Input fields. You will receive:
- product category anchors and source category paths;
- a shopping scenario describing the user's real-world context;
- a session plan specifying whether the intent is explicit or ambiguous, whether the session contains single or multiple subgoals, and when hidden constraints, preference updates, rejections, comparisons, confirmations, or intent shifts may appear;
- a scenario archetype, such as `combo_from_scratch`, `combo_core_upgrade`, `replacement_repair`, `scenario_bundle`, `consumable_restock`, `vague_exploration`, `mixed_errands`, or `urgent_timing`;
- constraint modifiers, such as `compatibility_sensitive`, `budget_tradeoff`, `high_risk_decision`, `logistics_time_sensitive`, or `bundle_split_decision`;
- optional catalog evidence, including product titles, brands, categories, prices, attributes, and fulfillment tags.

Generation requirements. Generate a profile that satisfies the following requirements:
- Separate long-term user preferences from current session-specific needs.
- Specify whether the session is single-intent or multi-intent consistently with the provided session plan.
- If the session is multi-intent, list subgoals, priority order, must-finish items, deferrable items, and bundle/split-order preference consistently with the provided session plan.
- Distinguish hard constraints, soft preferences, and negotiable constraints.
- Include compatibility, size, delivery, budget, safety, or fulfillment constraints when they are implied by the scenario.
- Include an interaction style with coping style, interaction mode, verbosity, politeness, skepticism, decisiveness, patience, and comparison habit.
- Generate the latent intent, constraints, interaction style, disclosure plan, and multi-intent structure consistently with the provided session plan.
- Do not modify, rewrite, or contradict the provided session plan.
- Include an evaluation focus describing what the shopping agent must do correctly to succeed.

Ambiguous-intent profiles. If the profile type is `ambiguous_intent`, do not expose the full target item in the opening utterance. Instead:
- write an opening utterance that reveals only a scenario, problem, recipient, or desired outcome;
- store the true goal in `latent_intent`;
- store hidden constraints in `latent_intent.must_have_constraints` and `latent_intent.nice_to_have_constraints`;
- construct a `disclosure_plan` consistently with the provided session plan, specifying how the corresponding hidden intent, constraints, preference updates, rejections, or intent shifts are revealed over turns.

Catalog-grounding rules. If catalog evidence is provided:
- concrete brands, attributes, categories, fulfillment tags, and product-specific constraints should be grounded in the evidence;
- unsupported details must not be written as hard constraints;
- if a detail is plausible but not supported by evidence, write it as a soft or negotiable preference;
- avoid inventing platform services, delivery guarantees, certifications, or exact product attributes that are not present in the evidence.

Output format. Return valid JSON with the following fields:
- `profile_id`, `profile_type`, `persona_seed`;
- `purchase_intent_scope`, `purchase_intent_subgoals`, `multi_intent_plan`;
- `shopping_scenario`, `synthesis_anchor`;
- `basic_information`, `long_term_preference`, `historical_behavior`;
- `current_intent`, `constraints`, `interaction_style`;
- `preference_priority`, `risk_preference`, `urgency_level`;
- `initial_utterance`, `initial_utterance_style`, `latent_intent`, `disclosure_plan`, `evaluation_focus`;
- `category_specific_extension`.

Figure 6: Profile synthesis prompt used to construct structured user profiles for REALWORLDSHOP.

User Simulator Prompt

You are playing the user in a controlled multi-turn e-commerce shopping conversation.

Goal. Generate the next user utterance according to the structured user profile, hidden state, dialogue history, and target user action. The utterance should be natural, but it must remain faithful to the profile and the assigned action.

Input fields. You will receive:
- the visible and hidden user profile;
- the user's long-term preferences and current session intent;
- hard, soft, and negotiable constraints;
- latent intent and disclosure plan for ambiguous profiles;
- dialogue history;
- the target user action for this turn;
- interaction style, including patience, skepticism, verbosity, decisiveness, politeness, and comparison habit.

Action control. Follow the target action exactly:
- PROVIDE: answer the assistant's clarification question or provide a missing slot.
- REFINE: make an existing requirement more specific without changing it.
- UPDATE: change a previously stated constraint, budget, preference, priority, active subgoal, or shopping intent.
- OVERRIDE: let the current session need override a long-term preference.
- REJECT: reject a recommendation and give a profile-consistent reason.
- COMPARE: ask the assistant to compare candidates or trade-offs.
- CONFIRM: accept a recommendation or purchase plan if it satisfies the active constraints.
- END: terminate the session without convergence because the user is impatient, unconvinced, or no longer wishes to continue.

Hidden-state rules. Do not reveal the full hidden profile at once.
- Reveal hidden constraints only when the assistant asks a relevant question, when the target action requires it, or when a recommendation violates them.
- For ambiguous-intent sessions, begin from the opening style and avoid naming all target products immediately.
- If the assistant asks a good clarification question, reveal the corresponding slot from the disclosure plan.
- If the assistant recommends an item that violates a hard constraint, reject it and restate the violated constraint.
- If the assistant forgets a previously revealed constraint, respond according to the user's patience and skepticism.

Style rules. Realize the utterance using the user's interaction style:
- low-patience users should be concise and may end early after repeated failures;
- high-skepticism users should ask for evidence, compatibility, price, or fulfillment details;
- efficiency-first users should prefer direct answers and dislike redundant questions;
- collaborative users may provide extra context and tolerate clarification;
- comparison-oriented users should ask for trade-offs among candidates.

Output constraints. Return only one user utterance. Do not include analysis, hidden fields, action labels, JSON, or explanations.

Figure 7: User simulator prompt used to generate controlled multi-turn user behavior in REALWORLDSHOP.

Shopping-Agent Prompt

You are a conversational shopping assistant for a real-world e-commerce platform.

Goal. Help the user reach a grounded and actionable purchase decision over a multi-turn conversation. You should clarify hidden needs, track evolving constraints, retrieve product evidence, compare candidates, and recommend only when the recommendation is supported by current user state and catalog evidence.

Visible information. You can observe:
- the dialogue history;
- the visible user prior, if available;
- catalog search and product tools;
- previous tool results and candidate products.

You cannot observe the user's hidden constraints, latent intent, disclosure plan, or evaluation labels.

Session-state requirements. Maintain an internal session state containing:
- active shopping intent and active subgoals;
- hard constraints, soft preferences, and negotiable constraints;
- rejected products and rejection reasons;
- confirmed requirements;
- unresolved uncertainties;
- stale candidates whose assumptions no longer match the current state.

Decision policy. At each turn, choose the next behavior according to the current state:
- CLARIFY if a blocking field is missing, such as budget, size, compatibility, recipient, urgency, or usage scenario.
- RETRIEVE if the user's intent is clear but no catalog evidence has been collected.
- VERIFY if candidates exist but have not been checked against hard constraints.
- COMPARE if the user asks about trade-offs or if multiple viable candidates remain.
- RECOMMEND only when candidates are grounded, compatible with hard constraints, and aligned with the active subgoal.
- CONFIRM when the user is ready to commit and the recommendation is actionable.

Constraint-handling rules.
- Hard constraints must not be violated.
- When the user updates a hard constraint, invalidate candidates that no longer satisfy it.
- Session-specific constraints override long-term preferences.
- In multi-intent sessions, keep must-finish and deferrable subgoals separate.
- In bundle or split-order sessions, explain which items should be bought together and which should be ordered separately.
- If the user rejects an item, do not recommend the same item again unless the user explicitly reconsiders it.

Grounding rules.
- Use catalog tools before making SKU-level claims.
- Do not invent product attributes, compatibility, price, inventory, delivery time, certification, or fulfillment services.
- If catalog evidence is incomplete, state uncertainty and ask for missing information.
- Product explanations should cite concrete evidence from retrieved records, such as category, attributes, price, fulfillment tag, or compatibility field.

Response style. Be concise, helpful, and action-oriented. Avoid over-questioning when enough information is available. Avoid premature recommendation when blocking constraints are missing. For low-patience users, ask at most one targeted clarification question before moving forward.

Figure 8: Shopping-agent prompt used to control session-level decision making in REALWORLDSHOP.

Evaluation Judge Prompt

You are evaluating a completed multi-turn shopping-agent dialogue.

Goal. Score whether the assistant handled the shopping session correctly. Do not reward fluent but unsupported responses. Focus on whether the assistant understood the user, tracked state, updated constraints, used product evidence, and guided the session toward an actionable purchase decision.

Input fields. You will receive:
- structured episode profile;
- visible and hidden user constraints;
- dialogue trajectory;
- tool trace;
- recommended products or final purchase plan;
- termination reason.

Turn-level scoring. For each assistant turn, score the following dimensions on a $\{0, 1, 2\}$ scale:
- **Need Understanding**: whether the assistant correctly understands the current user request, newly revealed constraints, and changes from previous turns.
- **Recommendation Accuracy**: whether recommended products are appropriate, grounded, and consistent with known constraints.
- **Rationale Quality**: whether recommendation reasons are specific, evidence-based, and useful for the user's decision.

Use N/A for recommendation-related dimensions when no recommendation is made.

Session-level scoring. Score the full trajectory on a $\{0, 1, 2\}$ scale:
- **Clarification**: asks about key missing fields before reliable recommendation.
- **State Tracking**: remembers revealed needs, constraints, priorities, and rejected items.
- **Constraint Updating**: revises behavior after the user changes or overrides a requirement.
- **Decision Pacing**: chooses when to clarify, retrieve, verify, compare, recommend, or confirm without premature commitment or excessive delay.
- **Personalized Recommendation**: aligns final or near-final recommendations with the user's profile and session-specific constraints.

Holistic outcome. Assign:
- `overall_success`: `success`, `partial`, or `failed`;
- `convergence_quality`: `strong`, `moderate`, `weak`, or `none`.

Failure criteria. Mark the session down if the assistant:
- recommends before clarifying blocking constraints;
- forgets or contradicts previously revealed information;
- continues using stale candidates after a constraint update;
- violates hard constraints;
- makes unsupported SKU-level, compatibility, price, or delivery claims;
- fails to separate urgent and deferrable subgoals;
- over-questions a low-patience user or fails to converge when sufficient information is available.

Output format. Return structured JSON with turn-level scores, session-level scores, holistic outcome, convergence quality, evidence turns, and concise justifications.

Figure 9: Evaluation judge prompt used to assess completed shopping-agent trajectories in REALWORLDSHOP.

H Case Study

We provide representative case studies to illustrate how REALWORLDSHOP stresses session-level decision control. The selected profiles cover compatibility-sensitive shopping, ambiguous intent, multi-intent coordination, low-patience users, and split-order decisions. The failure cases highlight common weaknesses observed in baseline agents, including missing clarification, stale-state recommendations, priority loss, and unsupported compatibility claims.

H.1 Representative Profiles

To illustrate the diversity and controllability of REALWORLDSHOP profiles, we present three representative cases. Our benchmark contains both *explicit-intent* profiles, where the initial request already exposes directly searchable goals, and *ambiguous-intent* profiles, where key goals or constraints are revealed only through interaction. Each profile follows a normalized schema that separates *hard*, *soft*, and *negotiable* requirements, covering budget, compatibility, risk tolerance, logistics, brand preference, and interaction style. To reduce unverifiable assumptions, profile synthesis can be conditioned on local catalog evidence; constraints that cannot be grounded in catalog attributes are downgraded to soft or negotiable preferences. All profiles are programmatically normalized and validated for schema consistency, multi-intent structure, ambiguous-intent fields, category anchoring, and catalog grounding. Further details are provided in Appendix C.3.

H.2 Representative Failure Modes

Case	Scenario	Stress Factor	Diagnostic Target
Profile A	Compatibility-sensitive multi-intent shopping	Old-wall-box fit, narrow balcony space, urgent vs. deferrable items	State tracking, compatibility verification, split-order planning
Profile B	Ambiguous core-item lighting upgrade	Vague study-room lighting need; 75 mm opening and budget priority revealed later	Clarification, hidden-constraint elicitation, hard-constraint propagation, candidate verification
Profile C	Low-patience family purchase	Speaker compatibility, fast convergence, later add-on item	Verification before recommendation, low-patience decision pacing
Failure 1	Missing clarification	Underspecified small-kitchen request	Avoid premature recommendation under hidden constraints
Failure 2	Priority loss in multi-intent shopping	Urgent setup items vs. optional decorative items	Preserve subgoal priority and avoid stale planning
Failure 3	Constraint update not propagated	85 mm recommendation after user corrects to 75 mm	Invalidate stale candidates after hard-constraint update
Failure 4	Unsupported compatibility claim	6.35 mm speaker connection for a digital piano	Avoid ungrounded product claims and verify catalog evidence

Table 13: Summary of representative case studies and diagnostic targets.

Profile A: Compatibility-Sensitive Multi-Intent Shopping

Profile type	Partially explicit multi-intent profile with latent compatibility and logistics constraints.
Persona	A specification-conscious, price-sensitive user buying home-renovation items for elderly parents.
Scenario	The user wants to build a balcony wellness corner involving a leakage-protection switch, narrow planters, mild tea, and a compact bathtub.
Category anchors	Electrical safety device, balcony planter, mild tea product, and compact bathtub.
Session plan	The user first asks for the switch and planters, then reveals old-wall-box and narrow-door constraints. After seeing candidate products, the user asks whether urgent and deferrable items should be split into separate orders.
Requirement schema	*Hard*: the switch must fit an old wall box; the bathtub must pass through a narrow bathroom door; urgent electrical safety items should be ordered first. *Soft*: price sensitivity, concise comparison, and mild tea preference. *Negotiable*: planters, tea, and bathtub can be postponed or ordered separately.
Interaction style	Efficiency-first, medium skepticism, concise comparison expected.
Grounding and validation	Compatibility and logistics constraints should be verified against product specifications or catalog attributes when available; unsupported compatibility assumptions are treated as soft preferences rather than hard constraints.
Evaluation focus	Preserve compatibility constraints across turns, prioritize the urgent safety item, and produce a reasonable split-order strategy.

Figure 10: Representative profile for compatibility-sensitive multi-intent shopping.

Profile B: Ambiguous Core-Item Upgrade

Profile type	Ambiguous-intent profile centered on a high-stakes home-improvement need with gradually revealed installation and budget constraints.
Persona	A budget-conscious father improving his child's study-room environment.
Scenario	The user wants more comfortable study-room lighting for reading and drawing, while also considering low-priority drawing tools and household supplies.
Initial utterance	"I want to improve the lighting in my child's study room. It should be more comfortable for reading and drawing. What would you recommend?"
Category anchors	Study-room lighting, drawing tools, and household supplies.
Session plan	The user starts with a vague lighting need. If the agent probes the setup, he reveals a preference for a no-main-light design. If asked about installation, he reveals that the existing ceiling opening is 75 mm and cannot be re-cut. If asked about quality or budget, he prioritizes glare control, high color rendering, and allocating most of the budget to the core lighting item.
Requirement schema	*Hard*: the final lighting solution must fit the revealed 75 mm opening and support comfortable study use. *Soft*: anti-glare design, high color rendering, and cost-effectiveness. *Negotiable*: drawing tools and household supplies are peripheral and lower priority.
Interaction style	Directive, high skepticism, low patience, key-option comparison preferred.
Grounding	Installation-sensitive claims must be grounded in catalog specifications such as cutout size, glare-control design, beam angle, and color-rendering index. Products lacking these attributes should not be treated as verified matches.
Evaluation focus	Elicit the true product target, clarify installation compatibility, propagate newly revealed hard constraints, and recover after incompatible recommendations.

Figure 11: Representative profile for an ambiguous core-item upgrade scenario.

Profile C: Low-Patience Compatibility-Sensitive Family Purchase

Profile type	Compatibility-sensitive family purchase with low patience and a direct-answer preference.
Persona	A pragmatic parent buying a better digital piano for a child after an exam milestone.
Scenario	The user wants a professional digital piano, needs to verify compatibility with an old external speaker, and may add an adapter cable and small learning accessories.
Initial utterance	"I want to replace my child's entry-level keyboard with a professional digital piano. Can it connect to our old speaker with a normal 6.35mm audio cable?"
Category anchors	Professional digital piano, external speaker connection, audio cable or adapter, and learning accessories.
Session plan	The user first asks about compatibility. If the agent clarifies the speaker interface, the user confirms the 6.35mm requirement. If the agent recommends products before checking compatibility, the user becomes skeptical and asks for a direct answer.
Requirement schema	*Hard*: the piano must support the required audio-output pathway or a verified compatible adapter; unsupported compatibility claims are unacceptable. *Soft*: professional quality, child-learning suitability, and reasonable trade-offs between function and price. *Negotiable*: adapter cables and small learning accessories can be postponed if the core piano choice is not yet confirmed.
Interaction style	Low patience, medium skepticism, trade-off seeking, concise answers preferred.
Grounding and validation	Audio-interface compatibility should be checked against product specifications, port descriptions, or verified adapter requirements; if catalog evidence is missing, the agent should explicitly ask for the speaker model or avoid making a definitive claim.
Evaluation focus	Verify compatibility before recommendation, avoid unsupported claims, and converge quickly under low-patience pressure.

Figure 12: Representative profile for low-patience compatibility-sensitive purchase.

Failure Case 1: Missing Clarification

Scenario	The user gives an underspecified small-kitchen request.
Dialogue	**User:** I want something useful for my small kitchen. It should make cooking easier. **Assistant:** I recommend a large multifunctional air fryer with high capacity and many cooking modes. **User:** My counter is tiny, I live alone, and I care about noise. I did not ask for a large appliance.
Failure	The assistant recommends before eliciting hidden constraints such as space, household size, noise tolerance, and preferred appliance type.
Expected	Ask targeted clarification questions before retrieving or recommending products.

Figure 13: Representative failure caused by missing clarification under ambiguous intent.

Failure Case 2: Priority Loss in Multi-Intent Shopping

Scenario	The user has urgent opening-day items and deferrable decorative items.
Dialogue	**User:** I need a commercial display fridge, bottled drinks for opening day, a flatbed trolley, and some decorative items. **Assistant:** I suggest buying all of them together to maximize discounts. **User:** The fridge, drinks, and trolley are opening-day essentials. Decorations can wait. **Assistant:** Then I recommend starting with decorative shelves and matching wall art to improve the store atmosphere.
Failure	The assistant loses the revealed priority structure and shifts from urgent operational subgoals to optional decorative items.
Expected	Track urgent and deferrable subgoals separately, prioritize the fridge, drinks, and trolley, and recommend a split-order plan when optional items can be postponed.

Figure 14: Representative failure caused by priority loss in a multi-intent shopping session.

Failure Case 3: Constraint Update Not Propagated

Scenario	The user reveals a hard installation constraint after an unsuitable recommendation.
Dialogue	**User:** I want to improve the lighting in my child's study room. It should be comfortable for reading and drawing. **Assistant:** I found a recessed spotlight with an 85 mm cutout. It has anti-glare design and high color rendering. **User:** 85 mm will not work. The existing ceiling opening is 75 mm, and I cannot re-cut the ceiling. **Assistant:** Understood. This 85 mm spotlight is still a good option because it has better brightness.
Failure	The assistant acknowledges the newly revealed hard constraint but continues recommending from a stale candidate set.
Expected	Invalidate the stale candidate and retrieve or verify products compatible with the revealed 75 mm opening.

Figure 15: Representative failure caused by stale candidate reuse after a newly revealed hard constraint.

Failure Case 4: Unsupported Compatibility Claim

Scenario	The user asks whether a candidate product is compatible with existing equipment.
Dialogue	**User:** I want to buy a professional digital piano. Can it connect to our old speaker with a normal 6.35mm audio cable? **Assistant:** This piano has weighted keys and realistic sound, so I recommend it. **User:** But can this model connect with a 6.35mm cable? **Assistant:** Yes, this model should work with a 6.35mm cable. Most digital pianos support external speakers.
Failure	The assistant asserts SKU-level compatibility without verifying the product interface or asking for the speaker model.
Expected	Check catalog evidence for the piano's output ports, ask for the speaker's input interface when needed, and state uncertainty if compatibility cannot be verified.

Figure 16: Representative failure caused by an unsupported product-compatibility claim.