CONSUMERGYM: TRAINING LLM AGENTS FOR REAL-WORLD CONVERSATIONAL COMMERCE VIA MULTI-TURN REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Conversational commerce requires agents to track user needs that emerge and change during dialogue, making both environment design and credit assignment important for policy learning. We introduce CONSUMERGYM, a unified training and evaluation environment combining controllable user-state evolution, catalog-grounded tools, decomposed rewards, and replayable interaction snapshots. The environment separates latent user requirements, revealed oracle states, and agent-maintained states without exposing oracle inputs to the policy. We further propose RUS-GPO, which combines failure-aware trajectory resampling with user-state graph and boundary-critic estimation, context-matched counterfactual comparisons, and credit assignment to state updates, plans, tool actions, and responses. Evaluations on two internal protocols and three external benchmarks cover Qwen3.5-4B, 9B, and 27B. Relative to the strongest evaluated non-RUS baseline at each scale, the PPO-based variant improves fixed-context Avg. P@1 by 4.8–5.4 points and ConsumerGym success by 3.3–6.1 percentage points. Improvements with a GRPO-based implementation show that the gains are not confined to the primary PPO configuration.

1 INTRODUCTION

Large language models (LLMs) are increasingly used to help users search for products, compare options, and order services through dialogue and tool use (Yao et al., 2022b;a; Zhou et al., 2024). Yet conversational commerce does not always begin with a fully specified goal: users may reveal requirements, revise preferences, reject candidates, or change their intent as the interaction unfolds (Wang et al., 2025; Huang et al., 2026; Feng et al., 2025b; Fan et al., 2026). For example, a user seeking a portable laptop may later lower their budget, exclude refurbished devices, and request a comparison before authorizing a purchase. Responding appropriately requires the agent to track current needs, acceptable options, unresolved questions, and the scope of user authorization, and to decide when to clarify, compare, recommend, or act (Montazeralghaem et al., 2025; Li et al., 2025).

Existing environments and benchmarks provide important foundations for grounded agent interaction and evaluation. WebShop and WebArena study shopping and web interaction through executable environments (Yao et al., 2022a; Zhou et al., 2024), while AgentGym supports training and evaluation across heterogeneous environments (Xi et al., 2025). Recent shopping benchmarks further cover product search, comparison, personalization, and conversational reasoning (Lyu et al., 2025; Wang et al., 2026a; Kim et al., 2026; Fan et al., 2026). However, executable interaction and final outcomes alone provide limited guidance for learning under evolving user requirements. Learning therefore requires environments that track disclosed information and decision consequences, support controlled action comparisons, and provide evidence for decision-level credit assignment.

Learning effective policies in this setting presents three challenges. First, decision quality depends on the user's current state: a recommendation that was appropriate earlier may become unsuitable after a budget revision, even when the catalog remains unchanged. Second, outcomes reflect decisions made both across turns and within each assistant turn. An agent may correctly update its user state but still issue a tool call based on outdated constraints, making a shared trajectory- or turn-level advantage too coarse to distinguish their contributions. Third, stochastic user responses make it

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

difficult to determine whether differences in return reflect assistant decisions or variation in the simulated interaction. Although step-level and graph-based methods support finer-grained comparisons (Feng et al., 2025a; He et al., 2026a; Wang et al., 2026c), applying them to conversational commerce requires comparing decisions under comparable user states and interaction contexts, distinguishing within-turn responsibilities, and accounting for role-play variability.

To address these challenges, we introduce CONSUMERGYM, a training and evaluation environment combining versioned catalogs, executable tools, and simulated users guided by latent profiles and session plans. It separates latent requirements, revealed information, and agent-maintained estimates, using oracle states for supervision without exposing them to the policy. Decomposed feedback and replayable snapshots support diagnosis and controlled comparison. Building on these interfaces, RUS-GPO prioritizes weak training scenarios and combines user-state graph estimates, a boundary critic, and selective counterfactual replay to estimate component-wise turn advantages. Dynamic responsibility assignment and unit-balanced optimization then direct this credit to state updates, plans, tool actions, and responses.

Across Qwen3.5-4B, 9B, and 27B, RUS-GPO (PPO) improves fixed-context Avg. P@1 by 4.8–5.4 points and CONSUMERGYM success by 3.3–6.1 percentage points over the strongest evaluated non-RUS baseline at each scale. Gains also extend to external benchmarks and the GRPO-based implementation. Separate environment checks and fixed-learner comparisons assess CONSUMERGYM, while component ablations evaluate the contributions of RUS-GPO.

Our main contributions are as follows:

- We introduce CONSUMERGYM, a user-state-aware commerce environment supporting policy training, evaluation, controlled replay, and decision-level diagnosis.
- We propose RUS-GPO, which connects failure-aware experience collection with state-conditioned, component-wise, and dynamic within-turn credit assignment.
- We validate the environment and evaluate policy learning on two internal protocols and three external benchmarks across three model sizes and both PPO and GRPO implementations.

2 RELATED WORK

2.1 INTERACTIVE ENVIRONMENTS FOR CONVERSATIONAL COMMERCE

WebShop and WebArena study grounded shopping and web interaction (Yao et al., 2022a; Zhou et al., 2024), while agent frameworks support training and evaluation across heterogeneous environments (Xi et al., 2025; 2026). Shopping benchmarks extend coverage to search, comparison, personalization, and conversational reasoning (Lyu et al., 2025; Wang et al., 2026a; Kim et al., 2026; Fan et al., 2026). User-simulator research provides complementary approaches to scalable interaction and studies whether simulated users are reliable evaluation proxies (Zhan et al., 2023; Yoon et al., 2024; Dou et al., 2025). CONSUMERGYM brings together controlled information disclosure, executable commerce feedback, training-side state supervision, and replay. The emphasis is not that prior resources lack interaction, but that these interfaces jointly support learning and decision-level diagnosis under evolving user needs.

2.2 AGENTIC REINFORCEMENT LEARNING AND CREDIT ASSIGNMENT

Reinforcement learning enables LLM agents to improve their decisions through interaction with an environment. In long-horizon tasks, however, delayed outcomes make it difficult to determine which intermediate decisions deserve credit. Building on policy optimization methods such as PPO (Schulman et al., 2017) and GRPO (Shao et al., 2024), recent work explores more structured ways to compare interaction experience. GiGPO (Feng et al., 2025a) compares actions at repeated states, while HGPO (He et al., 2026a) accounts for interaction history when forming comparison groups. G2PO (Wang et al., 2026c) organizes experience into state-transition graphs, and ReBel (Tang et al., 2026) uses belief-consistency supervision to guide learning. These approaches highlight the importance of interpreting decisions in their state and interaction context. Our work follows this direction in conversational commerce, where user requirements evolve and each assistant turn contains distinct functional units. RUS-GPO combines failure-aware experience collection with user-state

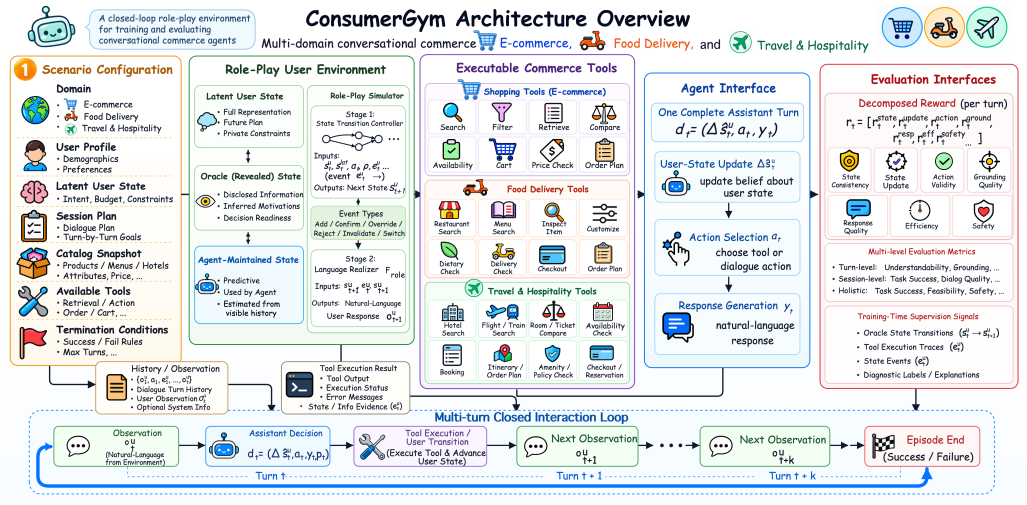


Figure 1: CONSUMERGYM architecture. The policy acts from dialogue and tool feedback; latent and revealed oracle states support simulation, supervision, and diagnosis. Shared tool, reward, and replay interfaces support training and evaluation.

graph–critic estimation and selective replay, then dynamically assigns component-wise credit to state updates, plans, tool actions, and responses.

3 CONSUMERGYM

3.1 ENVIRONMENT AND INTERACTION INTERFACE

CONSUMERGYM is a unified environment for supervised trajectory construction, interactive RL, evaluation, and failure diagnosis across e-commerce, food ordering, and travel and hospitality. Each episode specifies a user profile, latent requirements, a session and disclosure plan, a versioned catalog, executable tools, and simulator randomness. Dialogue, assistant decisions, tool observations, state events, reward components, and termination outcomes are retained in a common trajectory representation (Figure 1).

At assistant turn t, the policy observes the dialogue history and policy-visible tool feedback and produces a structured decision:

$$d_t = \left(q_t^{\text{state}}, q_t^{\text{plan}}, a_t, y_t\right), \tag{1}$$

where q_t^{state} encodes the agent-maintained estimate $\hat{s}_t^u$ of user needs and constraints, q_t^{plan} specifies the intended next step, a_t collects optional tool actions, and y_t is the user-visible response. State and plan are generated by the policy rather than supplied as oracle information. An assistant turn may include multiple tool calls and subsequent generations, with each execution result returned to the policy. After the final response, the role-play simulator advances the user state and generates the next user utterance.

3.2 USER-STATE-AWARE ROLE-PLAY AND REPLAY

The environment maintains three complementary state views. The latent state s_t^{lat} contains complete user requirements and the session plan; the revealed oracle state s_t^{rev} contains only information disclosed or inferable from policy-visible history by turn t; and $\hat{s}_t^u$ represents the agent's own estimate from that history. Comparing $\hat{s}_t^u$ with s_t^{rev} supports diagnosis of missed disclosures and stale constraints without treating hidden preferences as known facts. Neither the latent state nor the revealed oracle state is provided as an input to the actor during training or deployment.

User simulation separates structured state evolution from natural-language realization. Conditioned on the current user state, session plan, and assistant decision, a controller selects disclosure, revision, rejection, or intent-switch events. A role player then realizes the selected event as a natural-language utterance, allowing linguistic variation under structured control. Future disclosures remain outside the revealed state until supported by the visible interaction. The transition formulation and consistency checks are provided in Appendix B.

Turn-boundary snapshots preserve the dialogue prefix, user state, catalog version, tool state, session plan, and simulator randomness. Restoring a snapshot establishes a common pre-decision context while allowing subsequent interactions to evolve with the selected assistant decision. These interfaces support controlled diagnosis and the context-matched counterfactual comparisons used by RUS-GPO.

3.3 Executable Actions and Multi-Level Supervision

The commerce gateway supports catalog-grounded retrieval, filtering, comparison, price calculation, and order previews, validating calls against schemas, entity references, quantities, active constraints, and authorization rules. Invalid calls return structured errors rather than being silently corrected, and outcomes are determined by execution, not textual claims. Supervision comprises 12 reward components covering turn-level decisions, session-level processes, and holistic outcomes, with deterministic verifiers checking executable properties and frozen judge models assessing semantic properties. Raw component scores, provenance, and disjoint token masks for state, plan, action, and response are retained, allowing standard RL and RUS-GPO to share environment interfaces while differing in reward aggregation and credit assignment.

3.4 Training and Evaluation Protocols

Training and evaluation share the observation and transition protocol. Quality control covers scenario validity, user-simulation quality, and agent-trajectory quality, including response language and tool grounding. Construction, evaluation, and audit details appear in Appendices A–C.

4 Method: RUS-GPO

RUS-GPO uses ConsumerGym's state, reward, and replay interfaces to guide both experience collection and policy learning (Figure 2). Failure-aware resampling prioritizes weak training scenarios. A user-state graph, boundary critic, and selective counterfactual replay estimate component-wise turn advantages, which are then assigned to individual decision units for policy optimization.

For trajectory i and assistant turn t, the actor observes $o_{i,t} = (h_{i,t}, e^{\mathrm{obs}}_{i,t})$ and produces the structured decision in Eq. (1). An assistant turn includes all policy generations and tool interactions between consecutive user utterances. Let $k \in \mathcal{K}$ index the 12 reward components and T_i denote the number of assistant turns. After rewards are assigned to their corresponding turn boundaries, the component return is $G^k_{i,t} = \sum_{\ell=t}^{T_i-1} \gamma^{\ell-t} r^k_{i,\ell}$. We present the primary PPO formulation below; further algorithm details and the GRPO adaptation are provided in Appendix D.

4.1 Failure-Aware Trajectory Resampling

Following SFT, on-policy diagnosis identifies scenarios associated with low returns, state-tracking errors, invalid actions, or uncertainty. These signals increase the priority of weak cases while maintaining coverage of the original task pool. Selected scenarios produce fresh trajectories from the rollout policy, not copies of earlier failures. Diagnosis and priority updates remain separate from actor optimization; each trajectory records its rollout-policy and sampling configuration. The sampling formulation is given in Appendix D.2.

4.2 User-State Graph and Boundary-Critic Estimation

Comparable state transitions. A scenario-context key q_i records the user profile, session plan, catalog snapshot, and harness configuration. Within each context, $g_{i,t} = \phi(s^{\mathrm{rev}}_{i,t})$ canonicalizes the

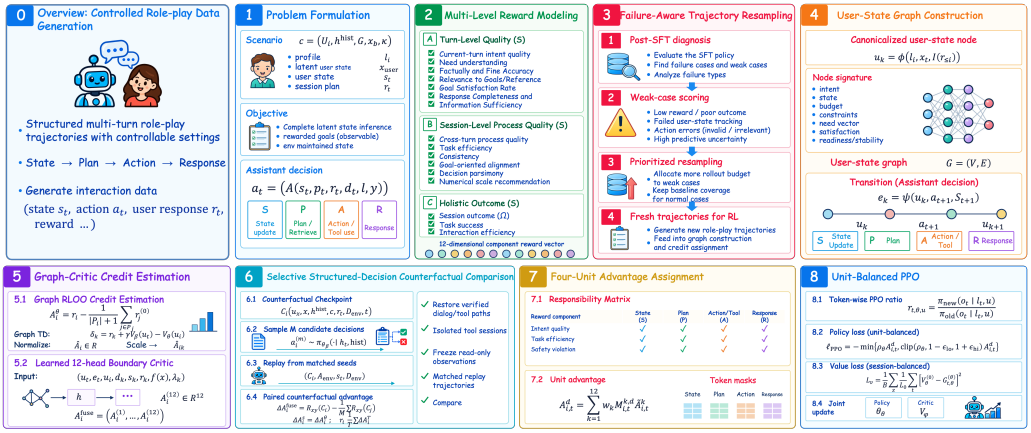


Figure 2: RUS-GPO workflow. Failure-aware sampling guides experience collection; graph–critic estimation and selective replay provide component-wise credit for dynamic four-unit assignment and unit-balanced optimization. The PPO implementation is shown.

revealed user state, retaining the interaction stage, active requirements, and rejected or confirmed candidates. The graph indexes observed transitions $(g_{i,t}, d_{i,t}, g_{i,t+1})$ for cross-trajectory comparison. Matching is exact over the canonical fields, preserves relevant candidate/reference context, and excludes current decisions, rewards, and future outcomes.

Leave-one-trajectory-out baseline. For each matched state, $V^k_{G,-i}(q_i, g)$ estimates the component value from other trajectories, first averaging repeated visits within each trajectory. Excluding trajectory i avoids self-inclusion, while within-trajectory averaging prevents repeated visits from receiving extra weight. The complete estimator is given in Appendix D.3. For supported current and successor nodes, the component-wise graph advantage is

$$A^{G,k}_{i,t} = r^k_{i,t} + \gamma(1 - b_{i,t})V^k_{G,-i}(q_i, g_{i,t+1}) - V^k_{G,-i}(q_i, g_{i,t}), \tag{2}$$

where $b_{i,t}$ indicates termination. When the current node is supported but its successor is not, we use the return residual $G^k_{i,t} - V^k_{G,-i}(q_i, g_{i,t})$ with reduced graph confidence. Unsupported current nodes instead rely on the boundary critic.

Confidence-weighted critic fallback. A 12-head boundary critic supplies component-wise advantages $A^{C,k}_{i,t}$ through generalized advantage estimation. During training, the critic supplements observable history with scenario-side context, while the actor receives only policy-visible information. Future realized rewards, future tool outcomes, and counterfactual returns are excluded from critic inputs. Graph confidence $c^k_{i,t} \in [0, 1]$, determined by cross-trajectory support and return dispersion, combines the two estimates:

$$A^{B,k}_{i,t} = c^k_{i,t} A^{G,k}_{i,t} + (1 - c^k_{i,t}) A^{C,k}_{i,t}. \tag{3}$$

Reliable graph evidence receives greater weight, whereas sparse or unsupported states rely more heavily on the critic. Values and advantages are computed using frozen rollout-time estimators and held fixed during the corresponding actor update. The graph-confidence formulation is provided in Appendix D.3.

4.3 Selective Context-Matched Counterfactual Comparison

At selected informative turn boundaries, we restore the pre-decision history, user state, latent plan, catalog, and tool state. The original decision $d^{\text{on}}_{i,t}$ is compared with N_{alt} replacements sampled from the frozen rollout policy. Branches share this starting context and the continuation-policy checkpoint, while subsequent interaction is stochastic. Let $\eta_{j,0}$ and $\eta_{j,m}$ denote the randomness of the original and replacement branches in repetition j. For bounded-horizon component return $Q^k_{i,t}$,

we compute

$$A_{i,t}^{\mathrm{CF},k} = \frac{1}{N_{\mathrm{rep}} N_{\mathrm{alt}}} \sum_{j=1}^{N_{\mathrm{rep}}} \sum_{m=1}^{N_{\mathrm{alt}}} \left[Q_{i,t}^{k}(d_{i,t}^{\mathrm{on}}; \eta_{j,0}) - Q_{i,t}^{k}(d_{i,t}^{(m)}; \eta_{j,m}) \right]. \tag{4}$$

A valid replay comparison augments the graph–critic estimate:

$$\widetilde{A}_{i,t}^{k} = A_{i,t}^{B,k} + \lambda_{\mathrm{CF}} m_{i,t}^{\mathrm{CF}} A_{i,t}^{\mathrm{CF},k}, \tag{5}$$

where $m_{i,t}^{\mathrm{CF}} \in \{0, 1\}$ marks replay validity and λ_{CF} controls the correction strength. Only the original on-policy decision receives this correction; replacements are comparison controls, not additional actor-training samples. Matching refers to the pre-decision context, not identical random draws. The finite-horizon comparison is not an exact causal attribution. Further details appear in Appendix D.4.

4.4 Dynamic Four-Unit Credit and Unit-Balanced Optimization

Dynamic responsibility assignment. Component-wise advantages distinguish evaluation dimensions, but each assistant turn also contains different functional units. We therefore distribute this credit across $\mathcal{D} = \{\mathrm{state}, \mathrm{plan}, \mathrm{action}, \mathrm{response}\}$. Let $\mathcal{D}_{i,t}^{+}$ denote the units containing valid actor tokens. For each turn, a responsibility allocator constructs $M_{i,t} = \mathcal{R}(E_{i,t}) \in \mathbb{R}_{+}^{12\times 4}$ from interaction-specific training evidence $E_{i,t}$. Unlike a fixed global routing table, its weights vary across turns.

Using the component weights w_k defined in Appendix C, the unit advantage is

$$A_{i,t}^{d} = \sum_{k\in\mathcal{K}} w_k M_{i,t}^{k,d} \widetilde{A}_{i,t}^{k}, \qquad \sum_{d\in\mathcal{D}_{i,t}^{+}} M_{i,t}^{k,d} = 1. \tag{6}$$

Responsibility weights are zero for inactive units. Row normalization distributes each component's credit among the active units, allowing the same reward component to emphasize different units in different interactions. The responsibility formulation is described in Appendix D.5.

Unit-balanced optimization. Policy-generated tokens inherit the advantage of their decision unit, while PPO importance ratios remain token-wise. User messages and tool observations have zero actor-loss mask. Let $\ell_{i,t,d,n}$ denote the negative clipped PPO surrogate for token n using advantage $A_{i,t}^{d}$, and let $n_{i,t,d}$ be the number of valid tokens in unit d. For a batch of B trajectories, we average losses within units and then across active units, turns, and trajectories:

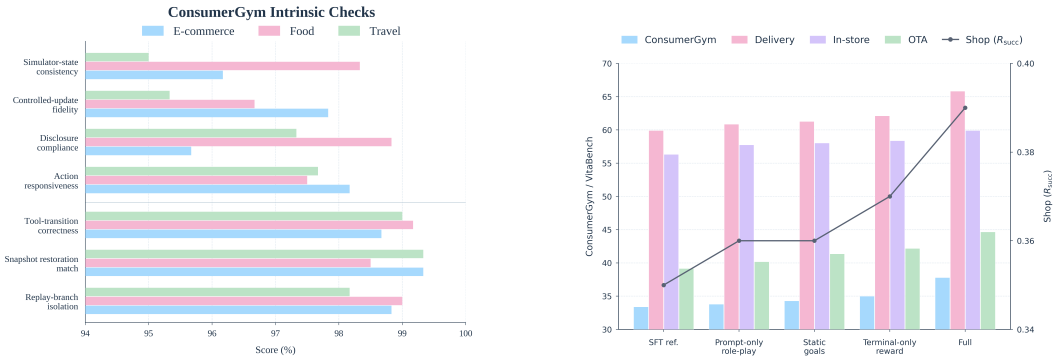
$$\mathcal{L}_{\mathrm{UB}} = \frac{1}{B} \sum_{i=1}^{B} \frac{1}{|\mathcal{T}_i^{+}|} \sum_{t\in\mathcal{T}_i^{+}} \frac{1}{Z_{i,t}} \sum_{d\in\mathcal{D}_{i,t}^{+}} \frac{\alpha_d}{n_{i,t,d}} \sum_{n=1}^{n_{i,t,d}} \ell_{i,t,d,n}, \tag{7}$$

where $\mathcal{T}_i^{+}$ contains active assistant turns and $Z_{i,t} = \sum_{d\in\mathcal{D}_{i,t}^{+}} \alpha_d$. Empty units are omitted, and $Z_{i,t}$ normalizes the remaining unit weights. With equal α_d, active units receive equal base aggregation weight regardless of their token lengths; advantage magnitude and clipping still affect their gradients. The PPO and GRPO implementations share this structured interface, with their estimation and normalization procedures described in Appendix D.7. At inference, only the actor and ordinary commerce interfaces are required.

5 Experiments

5.1 Experimental Setup

Benchmarks. We evaluate on two internal protocols and three external benchmarks. ConsumerGym's fixed-context evaluation contains 3,000 examples and measures next-decision quality through four category-level rubric scores (Action, Composition, Reference, and State) and aggregate Avg. P@1/P@3. Interactive role-play evaluates 1,500 scenarios, each repeated three times, using Turn, Session, and Success metrics. Both test sets cover e-commerce, food ordering, and travel and hospitality. External evaluation follows the official settings of ShoppingReasoningBench (Fan et al., 2026), ShopSimulator (Wang et al., 2026b), and VitaBench (He et al., 2026b). Detailed protocols are provided in Appendix A.



(a) Intrinsic environment checks.

(b) Training utility under ordinary PPO.

Figure 3: CONSUMERGYM validation and training utility. Left: seven environment checks across three domains. Right: each metric averaged across Qwen3.5-4B/9B/27B, with RUS-GPO enhancements disabled. Numerical results appear in Tables 10 and 11. Axes are truncated.

Models and baselines. We use Qwen3.5-4B, 9B, and 27B and compare against Base, SFT, PPO (Schulman et al., 2017), GRPO (Shao et al., 2024), GiGPO (Feng et al., 2025a), HGPO (He et al., 2026a), G2PO (Wang et al., 2026c), and ReBel (Tang et al., 2026). Within each backbone, RL methods share the SFT initialization, training task pool, and evaluation configuration. We report both PPO- and GRPO-based implementations of RUS-GPO. The data resources include 30,000 training scenarios and 200,000 SFT trajectories (Appendix B.4). Main results are reported per backbone; environment comparisons and ablations average each metric equally across the three model sizes.

5.2 CONSUMERGYM VALIDATION AND TRAINING UTILITY

Environment reliability. The four user-simulation checks reach at least 95% across all domains, while tool-transition correctness, snapshot restoration, and replay isolation reach at least 98% (Figure 3; Table 10). Automatic scores also agree closely with human judgments: Spearman correlations are 0.88–0.91 for Turn and 0.83–0.97 for Session (Table 6). Agreement with human rankings complements these checks of executable environment behavior, supporting the tested simulation and evaluation interfaces.

Impact of environment design on policy learning. Under the same ordinary PPO learner, full CONSUMERGYM achieves 37.8% task success, compared with 33.8% for prompt-only role-play, 34.3% for static goals, and 35.0% for terminal-only reward (Table 11). The static-goal control retains structured simulation but removes requirement revisions and intent switches; terminal-only training retains the dynamic environment but removes process feedback. Their respective gaps of 3.5 and 2.8 percentage points support training on evolving requirements with supervision beyond the final outcome. The benefit also extends to external tasks: relative to terminal-only feedback, full CONSUMERGYM improves Delivery from 62.1 to 65.8 and OTA from 42.2 to 44.7. These gains use ordinary PPO, independently supporting CONSUMERGYM's value as a training environment.

5.3 MAIN RESULTS

Internal evaluations. Across the three model sizes, RUS-GPO (PPO) improves fixed-context Avg. P@1 by 4.8–5.4 points and role-play success by 3.3–6.1 percentage points over the strongest non-RUS baseline at each scale (Tables 1 and 2). Avg. P@3 also improves by 2.6–5.7 points, so the advantage is observed under both single- and three-attempt evaluation. Gains extend across all four diagnostic categories and both Turn and Session scores. At 27B, State improves from the strongest baseline's 0.683 to 0.740, while Session improves from 0.535 to 0.589. The State rubric tests retention and revision of active requirements, whereas Session evaluates the cross-turn process. Composition and Reference gains further extend the pattern to maintaining multi-item plans and resolving historical entities. Because fixed-context evaluation holds the available history fixed, its

Table 1: Fixed-context diagnostics and ShoppingReasoningBench. Higher is better; bold values indicate the best result within each backbone.

Backbone	Method	Internal Fixed-Context						ShoppingReasoningBench	
		Action	Composition	Reference	State	Avg. P@1	Avg. P@3	Full-MT	Hard-MT
Qwen3.5-4B	Base Model	0.339	0.303	0.378	0.360	29.7	58.8	68.6	59.3
	SFT Backbone	0.532	0.456	0.508	0.473	43.3	70.7	72.5	62.2
	PPO	0.581	0.506	0.588	0.544	52.4	75.0	73.3	64.7
	GRPO	0.530	0.431	0.533	0.474	44.8	71.6	72.8	62.9
	GiGPO	0.529	0.465	0.547	0.492	50.1	73.0	73.1	64.0
	HGPO	0.545	0.499	0.539	0.520	51.2	73.3	75.5	66.1
	G2PO	0.556	0.527	0.552	0.569	52.0	76.8	76.1	67.3
	ReBel	0.611	0.529	0.606	0.578	53.6	75.9	75.8	67.0
	RUS-GPO (GRPO)	0.683	0.536	0.630	0.593	55.3	78.0	78.8	70.6
	RUS-GPO (PPO)	**0.704**	**0.561**	**0.654**	**0.611**	**58.4**	**82.5**	**81.2**	**73.3**
Qwen3.5-9B	Base Model	0.428	0.372	0.532	0.392	37.3	65.1	75.7	61.9
	SFT Backbone	0.568	0.518	0.608	0.524	48.1	76.9	81.3	64.2
	PPO	0.701	0.592	0.667	0.579	57.2	80.3	84.0	67.1
	GRPO	0.609	0.540	0.652	0.542	51.6	77.3	82.6	65.9
	GiGPO	0.648	0.557	0.654	0.552	52.2	78.5	83.0	66.7
	HGPO	0.644	0.575	0.663	0.579	58.5	81.8	84.4	65.3
	G2PO	0.646	0.607	0.638	0.614	58.7	81.9	85.0	68.5
	ReBel	0.695	0.616	0.675	0.599	59.6	82.8	85.2	70.1
	RUS-GPO (GRPO)	0.763	0.640	0.694	0.635	62.6	84.4	87.8	73.8
	RUS-GPO (PPO)	**0.792**	**0.653**	**0.711**	**0.650**	**64.9**	**86.6**	**88.3**	**75.4**
Qwen3.5-27B	Base Model	0.548	0.469	0.614	0.497	45.9	74.1	79.7	66.0
	SFT Backbone	0.625	0.565	0.659	0.578	57.6	79.5	86.9	70.3
	PPO	0.760	0.642	0.708	0.681	66.2	85.9	88.2	73.2
	GRPO	0.641	0.623	0.659	0.616	63.0	81.4	86.3	70.5
	GiGPO	0.671	0.651	0.684	0.623	65.5	82.6	87.0	70.9
	HGPO	0.696	0.674	0.662	0.628	65.0	83.1	88.0	72.4
	G2PO	0.687	0.665	0.667	0.674	64.4	82.3	87.7	73.1
	ReBel	0.721	0.707	0.691	0.683	66.7	86.3	89.3	75.6
	RUS-GPO (GRPO)	0.779	0.739	0.778	0.714	70.2	88.3	90.5	78.8
	RUS-GPO (PPO)	**0.790**	**0.762**	**0.783**	**0.740**	**72.1**	**88.9**	**92.4**	**81.4**

Notes. The four diagnostic columns report mean rubric scores; Avg. P@1/P@3 report aggregate pass rates, not averages of those columns. SRB uses its native importance-weighted rubric scores. Protocols appear in Appendix A.

gains reflect the next decision rather than differences in earlier generated dialogue. The role-play results separately show that the improvements persist when the agent's decisions shape subsequent interaction.

External benchmarks. On ShoppingReasoningBench Hard-MT, RUS-GPO (PPO) exceeds the strongest non-RUS baseline by 5.3–6.0 points across model sizes. It also achieves the highest reported scores on all three ShopSimulator metrics and the Delivery, In-store, and OTA domains of VitaBench at each scale. At 27B, ShopSimulator R_{succ} rises from the strongest baseline's 0.60 to 0.71, while OTA rises from 65.50 to 73.25. Consistent gains across these separately scored reasoning and interactive benchmarks support benefits beyond CONSUMERGYM's internal scoring rubric.

PPO and GRPO implementations. Both RUS-GPO variants improve fixed-context Avg. P@1 and role-play success over their corresponding optimizer baselines at every model size. At 9B, success increases by 7.3 percentage points over PPO and 6.1 over GRPO. These paired comparisons use the same backbone and SFT initialization, showing that the improvements are not simply a consequence of choosing PPO over GRPO. Although the PPO-based variant achieves stronger overall performance, the gains with both implementations support the usefulness of failure-aware experience collection and structured credit assignment beyond the primary PPO configuration.

5.4 ABLATION STUDY

PPO-based component ablations (Appendix E.3) fix the actor rollout budget and average scores across the three backbones. Removing any component lowers every reported metric. Without failure-aware resampling, Avg. P@1 falls by 3.55 points, the largest decrease; replacing graph–

Table 2: Interactive evaluation on CONSUMERGYM, ShopSimulator, and VitaBench. Higher is better; bold values indicate the best result within each backbone.

Backbone	Method	ConsumerGym Role-play			ShopSimulator			VitaBench		
		Turn	Session	Success (%)	R_{loose}	R_{strict}	R_{succ}	Delivery	In-store	OTA
Qwen3.5-4B	Base	0.343	0.257	21.5	0.44	0.16	0.14	36.00	32.75	18.75
	SFT Backbone	0.496	0.359	29.9	0.56	0.33	0.20	52.25	45.50	30.75
	PPO	0.510	0.489	34.4	0.60	0.35	0.24	57.75	47.00	35.25
	GRPO	0.492	0.442	32.5	0.57	0.30	0.21	53.00	47.25	33.50
	GiGPO	0.522	0.459	32.8	0.58	0.35	0.28	53.75	50.00	35.00
	HGPO	0.533	0.453	33.9	0.60	0.33	0.31	52.50	48.00	36.75
	G2PO	0.516	0.457	33.4	0.61	0.32	0.30	55.75	47.75	36.00
	ReBel	0.547	0.460	35.7	0.58	0.33	0.30	55.00	47.25	35.50
	RUS-GPO (GRPO)	0.566	0.483	36.1	0.64	0.41	0.37	60.00	50.25	37.00
	RUS-GPO (PPO)	**0.581**	**0.504**	**39.0**	**0.67**	**0.45**	**0.39**	**63.75**	**53.00**	**40.25**
Qwen3.5-9B	Base	0.407	0.310	26.9	0.46	0.25	0.22	45.75	47.50	31.00
	SFT Backbone	0.484	0.417	33.4	0.57	0.40	0.38	59.75	60.50	37.75
	PPO	0.556	0.517	38.2	0.63	0.43	0.40	64.00	65.50	43.75
	GRPO	0.524	0.484	35.2	0.59	0.41	0.38	60.25	61.00	40.00
	GiGPO	0.523	0.480	36.6	0.63	0.44	0.38	62.00	62.25	40.00
	HGPO	0.527	0.494	35.9	0.66	0.45	0.36	63.00	62.00	41.25
	G2PO	0.540	0.491	38.0	0.66	0.44	0.39	65.00	62.50	43.00
	ReBel	0.572	0.499	39.4	0.64	0.47	0.43	66.25	63.50	45.50
	RUS-GPO (GRPO)	0.594	0.520	41.3	0.71	0.54	0.45	67.75	64.25	47.00
	RUS-GPO (PPO)	**0.633**	**0.548**	**45.5**	**0.74**	**0.57**	**0.49**	**71.75**	**68.00**	**51.00**
Qwen3.5-27B	Base	0.480	0.398	32.7	0.60	0.43	0.39	61.00	58.25	43.25
	SFT Backbone	0.569	0.461	36.8	0.73	0.60	0.46	67.75	63.00	49.00
	PPO	0.630	0.526	40.9	0.78	0.66	0.52	75.75	67.25	55.00
	GRPO	0.586	0.488	38.0	0.74	0.60	0.47	71.00	65.50	50.00
	GiGPO	0.589	0.486	39.6	0.75	0.62	0.50	71.75	67.00	52.25
	HGPO	0.597	0.481	40.2	0.77	0.61	0.54	72.00	68.00	54.50
	G2PO	0.622	0.514	40.4	0.75	0.60	0.55	76.75	70.00	58.25
	ReBel	0.638	0.535	42.4	0.77	0.68	0.60	78.00	72.50	65.50
	RUS-GPO (GRPO)	0.669	0.568	43.9	0.80	0.71	0.67	80.75	75.50	68.25
	RUS-GPO (PPO)	**0.694**	**0.589**	**47.8**	**0.83**	**0.77**	**0.71**	**83.00**	**78.50**	**73.25**

Notes. Scores retain their native scales. Methods share the inference and judge configuration within each evaluation; benchmark protocols are given in Appendix A.

critic estimation with PPO/GAE reduces it by 2.43 points. Removing counterfactual replay lowers ShopSimulator R_{succ} from 0.530 to 0.514; saved replay interactions are not reallocated, so total interaction cost is not matched. Removing structured unit credit jointly disables dynamic responsibility assignment and unit-balanced optimization, causing the largest decreases in Session (0.547 to 0.502) and task success (44.10% to 40.82%). These different degradation patterns support complementary roles for experience selection and within-turn credit assignment. Full plots, numerical results, and controls appear in Appendix E.

6 CONCLUSIONS

We introduce CONSUMERGYM and RUS-GPO for learning conversational commerce policies under evolving user requirements. CONSUMERGYM provides controlled user-state evolution, executable tools, multi-level feedback, and replay; RUS-GPO combines failure-aware sampling with state-aware estimation and dynamic decision-unit credit assignment. Environment controls support the training value of changing requirements and process feedback, while ablations highlight the complementary roles of experience selection and structured credit. Evaluations on two internal protocols and three external benchmarks show improved decision quality and task completion across three model sizes with both PPO and GRPO.

AI USE STATEMENT

During manuscript preparation, we used AI tools to assist with language polishing and the creation and refinement of workflow diagrams and figures. The authors reviewed and edited the AI-assisted text and visual materials and take full responsibility for the accuracy and integrity of the final manuscript. The use of language models for data construction, user simulation, and evaluation is documented in Appendices A–C.

ETHICS STATEMENT

All research procedures in this study, including data construction, model training, and human and automated evaluation, were conducted in accordance with applicable institutional ethical guidelines and Institutional Review Board (IRB) requirements. Agent interactions in CONSUMERGYM took place in a controlled simulation environment. Ordering and booking were restricted to sandboxed preview operations, without initiating real purchases, payments, or reservations.

REPRODUCIBILITY STATEMENT

The appendices describe the benchmark protocols, environment interfaces, reward design, algorithmic procedures, and experimental comparisons. Upon acceptance, we will publicly release the complete dataset and codebase, including the data construction, training, evaluation, and analysis pipelines. The release will also include split manifests, evaluation prompts, and the experiment-specific configuration files with resolved implementation defaults, including the sampling, responsibility-allocation, and replay settings needed to reproduce the reported experiments.

REFERENCES

Yao Dou, Michel Galley, Baolin Peng, Chris Kedzie, Weixin Cai, Alan Ritter, Chris Quirk, Wei Xu, and Jianfeng Gao. Simulatorarena: Are user simulators reliable proxies for multi-turn evaluation of ai assistants? In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 35212–35290, 2025.

Shuxian Fan, Seonwoo Min, Youna Hu, Botao Xia, Jayakrishnan Unnikrishnan, Rowan Musselmann, Yifan Gao, Qingyu Yin, Priyanka Nigam, and Bing Yin. Shopping reasoning bench: An expert-authored benchmark for multi-turn conversational shopping assistants. *arXiv preprint arXiv:2606.12608*, 2026.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for llm agent training. *Advances in Neural Information Processing Systems*, 38:46375–46408, 2025a.

Xueyang Feng, Jingsen Zhang, Jiakai Tang, Wei Li, Guohao Cai, Xu Chen, Quanyu Dai, Yue Zhu, and Zhenhua Dong. Expectation confirmation preference optimization for multi-turn conversational recommendation agent. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 5896–5914, 2025b.

Shuo He, Lang Feng, Qi Wei, Xin Cheng, Lei Feng, and Bo An. Hierarchy-of-groups policy optimization for long-horizon agentic tasks. In *The Fourteenth International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=T8Dev99qnz.

Wei He, Yueqing Sun, Hongyan Hao, Xueyuan Hao, Zhikang Xia, Qi Gu, Hui Su, and Xunliang Cai. Vitabench: Benchmarking llm agents with versatile interactive tasks in real-world applications. In *International Conference on Learning Representations*, volume 2026, pp. 91654–91681, 2026b.

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

Sunghwan Kim, Ryang Heo, Yongsik Seo, Jinyoung Yeo, and Dongha Lee. Agenticshop: Benchmarking agentic product curation for personalized web shopping. In *Proceedings of the ACM Web Conference 2026*, pp. 2489–2500, 2026.

Chuang Li, Yang Deng, Hengchang Hu, Min-Yen Kan, and Haizhou Li. Chatcrs: Incorporating external knowledge and goal guidance for llm-based conversational recommender systems. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 295–312, 2025.

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

Ali Montazeralghaem, Guy Tennenholtz, Craig Boutilier, and Ofer Meshi. Asking clarifying questions for preference elicitation with large language models. *arXiv preprint arXiv:2510.12015*, 2025.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Wenjie Tang, Minne Li, Sijie Huang, Liquan Xiao, and Yuan Zhou. Rewarding beliefs, not actions: Consistency-guided credit assignment for long-horizon agents. *arXiv preprint arXiv:2605.20061*, 2026.

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

Pei Wang, Yanan Wu, Xiaoshuai Song, Weixun Wang, Gengru Chen, Zhongwen Li, Kezhong Yan, Qi Liu, Ken Deng, Shuaibing Zhao, et al. Shopsimulator: Evaluating and exploring rl-driven llm agent for shopping assistants. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 35600–35619, 2026b.

Xiaolei Wang, Chunxuan Xia, Junyi Li, Fanzhe Meng, Lei Huang, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Search-based interaction for conversation recommendation via generative reward model based simulated user. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 75–84, 2025.

Yunan Wang, Minghui Song, Zihan Zhang, Shaohan Huang, Haizhen Huang, Furu Wei, Weiwei Deng, Feng Sun, and Qi Zhang. Group-graph policy optimization for long-horizon agentic reinforcement learning. *arXiv preprint arXiv:2606.22995*, 2026c.

Zhiheng Xi, Yiwen Ding, Wenxiang Chen, Boyang Hong, Honglin Guo, Junzhe Wang, Xin Guo, Dingwen Yang, Chenyang Liao, Wei He, Songyang Gao, Lu Chen, Rui Zheng, Yicheng Zou, Tao Gui, Qi Zhang, Xipeng Qiu, Xuanjing Huang, Zuxuan Wu, and Yu-Gang Jiang. AgentGym: Evaluating and training large language model-based agents across diverse environments. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 27914–27961, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1355. URL https://aclanthology.org/2025.acl-long.1355/.

Zhiheng Xi, Dingwen Yang, Jiaqi Liu, Jixuan Huang, Honglin Guo, Baodai Huang, Tinggang Chen, Qi Zhang, Zhonghang Lu, Chenyu Liu, et al. Agentgym2: Benchmarking large language model agents in de-idealized real-world environments. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 44451–44479, 2026.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022a.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *NeurIPS 2022 Foundation Models for Decision Making Workshop*, 2022b.

Se-eun Yoon, Zhankui He, Jessica Maria Echterhoff, and Julian McAuley. Evaluating large language models as generative user simulators for conversational recommendation. *arXiv preprint arXiv:2403.09738*, 2024.

Qiusi Zhan, Xiaojie Guo, Heng Ji, and Lingfei Wu. User simulator assisted open-ended conversational recommendation system. In *Proceedings of the 5th Workshop on NLP for Conversational AI (NLP4ConvAI 2023)*, pp. 89–101, 2023.

Shuyan Zhou, Frank F Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, et al. Webarena: A realistic web environment for building autonomous agents. In *International Conference on Learning Representations*, volume 2024, pp. 15585–15606, 2024.

A Benchmark and Evaluation Details

A.1 Internal Evaluation Protocols

A.1.1 Fixed-Context Evaluation

Test set. The fixed-context test set contains 3,000 decision examples from 3,000 distinct scenarios or sessions, with one evaluated decision per example. The test set is evenly distributed across the three domains considered in ConsumerGym: 1,000 examples for e-commerce, 1,000 for food ordering, and 1,000 for travel and hospitality. Each example provides the complete policy-visible interaction prefix before the evaluated assistant decision, including dialogue history and available tool observations. Although only the next assistant decision is evaluated, the supplied context may contain multiple preceding dialogue turns. The assistant may invoke at most 10 tools within the evaluated turn.

Each test example belongs to exactly one of four non-overlapping diagnostic categories: Action, Composition, Reference, or State. These categories describe the capability being evaluated and are distinct from the four State/Plan/Action/Response decision units used by RUS-GPO.

Action. Action evaluates whether the assistant correctly controls the boundary between search, refresh, calculation, selection, and ordering according to the current intent and interaction history. The assistant must execute only user-authorized operations while preserving the correct target item, quantity, and price conditions. Expressions such as interest in a product, requests for comparison, or requests for assistance in choosing an option must not be silently upgraded into authorization to place an order. When an order operation succeeds, the assistant must present the corresponding execution-grounded order preview; failed operations must be reported faithfully rather than fabricated or replaced with previous execution results.

Composition. Composition evaluates the ability to maintain multi-item solutions under local revisions and global constraints. The assistant must correctly add, remove, replace, or restore components while preserving unaffected items and avoiding the reintroduction of explicitly removed components. The final configuration must remain consistent with the latest requirements in item identity, quantity, compatibility, and other active constraints. Total-price and budget calculations must be correct, and components from different candidate solutions must not be inadvertently mixed.

Reference. Reference evaluates whether expressions such as "the second one," "the former," "it," or "the last item from the initial recommendation" are grounded to the correct historical entity. The assistant must identify the corresponding dialogue turn and candidate-list version while preserving the referenced item's identity and specifications. Newly retrieved candidates must not silently replace an earlier referent. When multiple referring expressions resolve to the same item, this relationship should be handled consistently, and accessories must be matched to the appropriate model, size, or generation.

State. STATE evaluates whether the assistant maintains and updates user constraints, preferences, and task states across turns. Previously stated conditions remain active unless they are explicitly revised or removed. New revisions supersede outdated conditions, while only explicitly cancelled constraints should be discarded. When the user switches between tasks or later resumes an earlier task, the corresponding state must be retained without mixing constraints across tasks or users. Hard constraints must not be relaxed without explicit user authorization.

Scoring. GPT-5.5 serves as the rubric evaluator for the fixed-context benchmark. The ACTION, COMPOSITION, REFERENCE, and STATE columns report the mean rubric score over examples in the corresponding diagnostic category.

We additionally report aggregate Pass@1 and Pass@3 over the fixed-context test set. Pass@1 measures the proportion of examples for which a single generated attempt satisfies its category-specific rubric, whereas Pass@3 measures the proportion for which at least one of three independently generated attempts satisfies the rubric. Formally,

$$\text{Pass@}\,k = \frac{1}{|\mathcal{V}|}\sum_{i\in\mathcal{V}}\mathbb{I}\left[\max_{1\le j\le k} b_{i,j} = 1\right], \qquad k \in \{1,3\}, \tag{8}$$

where $\mathcal{V}$ denotes the scored test examples and $b_{i,j} \in \{0,1\}$ indicates whether attempt j for example i passes its rubric. Hence, the category-level rubric scores and Avg. P@1/P@3 measure different quantities and the latter are not computed by averaging the four diagnostic columns.

A.1.2 INTERACTIVE ROLE-PLAY EVALUATION

Test scenarios. The interactive role-play test set contains 1,500 held-out scenarios, balanced across the three domains: 500 e-commerce scenarios, 500 food-ordering scenarios, and 500 travel-and-hospitality scenarios. Each scenario is evaluated through three independent role-play runs, resulting in 4,500 scheduled sessions per evaluated policy. Kimi K2.6 is used as the role-play user simulator.

Each session contains at most 20 interaction turns, and the assistant may issue at most 10 tool calls within each turn. The user's remaining requirements and disclosure plan are hidden from the policy throughout the interaction. Runs affected by infrastructure-level execution failures or timeouts are excluded from score aggregation. Invalid actions produced by the policy remain part of the evaluated trajectory and are scored as policy errors, rather than triggering exclusion of the session. An unsuccessful task outcome is not itself an exclusion criterion.

Turn-level quality. Turn-level evaluation measures the quality of individual assistant decisions using five rubric dimensions: *need understanding*, *factuality and price accuracy*, *recommendation relevance and selection quality*, *response completeness and recommendation sufficiency*, and *rationale usefulness*. Together, these dimensions evaluate whether the assistant correctly understands the current intent and constraints, grounds claims and calculations in available evidence, selects suitable options, provides sufficient information, and explains relevant trade-offs.

Session-level quality. Session-level evaluation measures cross-turn interaction quality using five dimensions: *clarification*, *state tracking*, *constraint updating*, *decision pacing*, and *personalized recommendation*. These dimensions assess whether the assistant requests missing information when necessary, maintains active constraints and user feedback, responds correctly to requirement changes, sequences clarification, retrieval, comparison, and execution appropriately, and adapts recommendations to the evolving user state.

Holistic outcomes. Holistic evaluation contains two components. *Task success* measures whether the final result satisfies the user's active goals and constraints, is supported by valid execution evidence, and respects the user's authorization. *Interaction efficiency* evaluates information gain, economical tool usage, avoidance of redundant interaction, and timely convergence and termination.

The main role-play results report **Turn**, **Session**, and **Success**. Turn and Session use the arithmetic mean of their respective rubric dimensions, while Success reports task-level completion. The component training weights do not define these evaluation averages. The same rubric definitions are used

for training supervision and evaluation, ensuring that evaluation measures remain aligned with the interaction qualities optimized during training. The complete 12-component reward specification and its training weights are given in Appendix C.

A.2 External Benchmarks

For all external benchmarks, we use the official released test sets, default evaluation configurations, and native scoring protocols without modifying their task definitions or metric implementations.

ShoppingReasoningBench. ShoppingReasoningBench evaluates expert-authored conversational shopping missions requiring preference refinement, product comparison, trade-off reasoning, compatibility assessment, and other forms of shopping-domain reasoning (Fan et al., 2026). We follow the benchmark's official multi-turn evaluation protocol and report its native importance-weighted rubric scores on the **Full-MT** and **Hard-MT** settings.

ShopSimulator. ShopSimulator provides an interactive shopping environment designed to evaluate personalized search, multi-turn user interaction, and fine-grained product selection (Wang et al., 2026b). We use the official default evaluation setting and report the three native metrics R_{loose}, R_{strict}, and R_{succ}, without modifying the benchmark's episode construction or scoring rules.

VitaBench. VitaBench evaluates interactive agents on real-world service scenarios involving complex tool use and evolving user requirements (He et al., 2026b). We follow its official default evaluation protocol and report the three task domains used in our experiments: **Delivery**, **In-store**, and **OTA**. All scores retain the benchmark's native definitions, evaluation repetitions, and aggregation procedure.

A.3 Model-Scale Aggregation

For environment-level comparisons that summarize results across the three evaluated model sizes, metric j for method m is macro-averaged as

$$\overline{S}_{m,j} = \frac{1}{3}\left(S^{\text{4B}}_{m,j} + S^{\text{9B}}_{m,j} + S^{\text{27B}}_{m,j}\right). \tag{9}$$

Only scores corresponding to the same metric are averaged across model scales; scores from heterogeneous benchmarks are never combined into a single overall metric. Complete policy results are reported separately for Qwen3.5-4B, Qwen3.5-9B, and Qwen3.5-27B in the main results tables.

B ConsumerGym Implementation and Data Quality

B.1 Scenario, State, and Trajectory Interfaces

A ConsumerGym scenario contains a user profile and latent requirements, a session/disclosure plan, catalog conditions, tool permissions, a terminal contract, and simulator randomness. The latent user state stores the simulator's complete requirements, whereas the revealed oracle contains only information supported by the policy-visible interaction prefix. The policy maintains its own user-state estimate from dialogue and tool observations; this estimate is generated by the policy rather than copied from the revealed oracle.

The common trajectory representation retains dialogue, policy-generated decisions, tool executions and observations, state transitions, reward components, and termination outcomes. Training and replay additionally require explicit turn boundaries and snapshot provenance. Table 3 summarizes the logical interface; the listed fields describe their semantic roles rather than literal implementation key names.

Table 3: Logical trajectory fields retained by the CONSUMERGYM environment.

Record group	Contents
Scenario	User profile, latent requirements, disclosure plan, catalog, permissions
Policy observations	Dialogue prefix and policy-visible tool feedback
Assistant decision	State estimate, plan, executable actions, and final response
State event	Pre/post revealed state and disclosure, revision, rejection, or switch event
Supervision	Component scores, provenance, decision-unit spans, and loss masks
Replay	Pre-decision snapshot, tool state, session-plan position, and simulator state
Outcome	Success/termination label, error type, and associated evidence

B.2 CONTROLLED STATE EVOLUTION AND LANGUAGE REALIZATION

CONSUMERGYM separates structured user-state evolution from natural-language realization:

$$(s_{t+1}^{\mathrm{rev}}, e_t^u) = F_{\mathrm{state}}\left(s_t^{\mathrm{rev}}, s_t^{\mathrm{lat}}, d_t; \rho\right), \tag{10}$$

$$u_{t+1} \sim F_{\mathrm{role}}\left(p, s_{t+1}^{\mathrm{rev}}, e_t^u, h_t, d_t; \zeta\right), \tag{11}$$

where p denotes the user profile, h_t the current interaction history, and e_t^u the selected user-state event. We use ρ to denote the controller-side configuration and randomness, and ζ to denote the role-generation-side configuration and randomness.

F_{state} combines rule-based control with LLM-based reasoning. Conditioned on the latent session plan, the current revealed state, and the assistant decision, it determines the next structured user event. Supported events include information disclosure, requirement revision, candidate rejection, and intent switching. Requirement revisions modify the corresponding latent user state rather than merely changing its surface realization.

The session plan is advanced through an explicit pointer together with stage- and turn-dependent conditions. This allows the environment to control when information becomes eligible for disclosure, when revisions can occur, and when the interaction progresses to a new stage.

F_{role} is instantiated with Kimi K2.6 and converts the selected structured event into a natural-language user response. The generated utterance is checked against the intended event before the revealed state is advanced. If the realization does not express the required event, the role player regenerates the utterance, with at most five attempts. A realization that still fails after the retry budget is rejected and marked as a simulator failure rather than silently recorded as a successful state transition. Consequently, information scheduled in the latent plan is not treated as revealed unless it is actually communicated in the dialogue.

B.3 EXECUTABLE COMMERCE TOOLS AND FAILURE HANDLING

Tool execution produces environment observations and state changes rather than treating natural-language claims as completed actions. Table 4 summarizes the executable tool families used in the three CONSUMERGYM domains.

Table 4: Executable commerce tools used across the three CONSUMERGYM domains.

Domain	Tools
E-commerce	`search`, `filter`, `retrieve`, `compare`, `availability`, `cart`, `price_check`, `order_preview`
Food ordering	`restaurant_search`, `menu_search`, `inspect_item`, `dietary_check`, `delivery_check`
Travel and hospitality	`hotel_search`, `flight_search`, `room_compare`, `availability`, `booking_preview`

Ordering and booking operations are sandboxed as preview operations. The environment can therefore return execution-grounded order or booking previews without performing a real external purchase, payment, or reservation.

Tool calls are validated against their schemas, referenced entities, quantities, active constraints, availability, and user authorization. Failures such as malformed schemas, invalid entity references, constraint violations, or unauthorized actions remain explicit structured observations in the trajectory instead of being silently corrected. Accordingly, a natural-language statement that an operation has succeeded cannot substitute for a valid tool result.

B.4 Data Construction and Quality Control

Data scale and partitioning. The data resources include 30,000 training scenarios and 200,000 SFT trajectories. The train/validation/test split ratio is 8:1:1. The released data will include split manifests and scenario-to-trajectory mappings. The fixed-context and role-play evaluation sets, including their test-set sizes and domain composition, are specified separately in Appendix A.

Data construction follows three stages: scenario validation, user-simulation validation, and agent-trajectory validation. The stages target different sources of error and are therefore applied separately.

Scenario validity. Candidate scenarios are generated with GPT-5.5. Scenario validation combines deterministic checks with GPT-5.5-based semantic review. The checks cover task coherence, consistency of latent user requirements, catalog and tool support, constraint feasibility, and the terminal contract.

A partially specified initial request is not considered invalid by itself: such scenarios may intentionally require clarification, preference elicitation, or constraint negotiation. Similarly, an infeasible shopping goal may be retained when recognizing that no acceptable option exists is part of the intended task.

User-simulation quality. Role-play trajectories are checked by GPT-5.5 for consistency with the user profile and latent session plan, legal state revisions, disclosure timing, and responsiveness to assistant behavior. A permitted preference change, such as revising a budget during the conversation, is treated as a valid state transition rather than a persona contradiction. Intended state events are checked against the information actually expressed by the generated user utterance.

Agent-trajectory quality. SFT trajectories are generated by Gemini 3.1 Pro Preview and checked by GPT-5.5. Quality checks cover user-state tracking, consistency between the generated plan and subsequent actions, tool grounding, factual support, and response quality, including coherence, relevance, sufficiency, and unsupported claims.

A trajectory is retained as an SFT demonstration only when it passes the trajectory-quality checks and successfully satisfies its terminal task contract.

This SFT demonstration-selection procedure is distinct from RL experience collection. Genuine policy errors and unsuccessful outcomes generated by the current RL policy remain useful learning and diagnostic evidence and are not discarded merely because the policy fails. Corrupted environment records and infrastructure failures are handled separately from genuine policy failures.

B.5 Snapshots and Replay Integrity

A turn-boundary snapshot preserves the decision prefix, revealed user state, relevant latent-session state, catalog version, tool state, session-plan position, and simulator state. Restoring the snapshot therefore re-establishes the same pre-decision interaction context for subsequent branching.

Mutable commerce state, including carts, order previews, and other tool state, is deep-copied for each replay branch. Alternative branches therefore cannot modify one another's user state or commerce state.

The environment-level replay contract guarantees restoration of the pre-decision state and isolation between branches. It does not require alternative assistant decisions to receive identical future user utterances, nor does it assume that alternative branches globally share the same user-simulator seed. Different decisions may legitimately induce different subsequent state transitions and language realizations.

The selective context-matched comparisons used by RUS-GPO are specified in Appendix D.4. Snapshot restoration and branch isolation match the pre-decision environment; subsequent continuations remain stochastic. A shared pre-decision state does not imply a shared user-simulator random stream.

B.6 Environment-Controlled Training Comparisons

Table 11 evaluates the contribution of the environment design under an ordinary PPO learner. All RUS-GPO-specific sampling, credit-estimation, responsibility-routing, and unit-balanced optimization mechanisms are disabled. The conditions are independent controls rather than successive training stages.

SFT reference. The SFT reference directly evaluates the task-specific SFT checkpoint without reinforcement learning and serves as a no-RL reference.

Prompt-only role-play + tools. This condition completely removes the structured state-transition controller. Kimi K2.6 performs free-form user role-play from the supplied user profile and session prompt, while the same executable commerce tools remain available. Thus, the condition preserves interactive language generation and tool grounding but removes explicit structured control over user-state disclosure, revision, rejection, and intent switching.

Static goals. This condition retains the structured simulator and executable tool interfaces but disables within-session requirement revision and intent switching. The user's initial goals and constraints therefore remain fixed throughout the interaction, while information disclosure and ordinary role-play continue under the structured simulator.

Terminal-only reward. This condition retains the complete interactive environment but trains PPO using only the holistic `task_success` reward. Turn-level rewards, session-level rewards, and the holistic `interaction_efficiency` component are removed.

Full ConsumerGym. The full condition uses dynamic user-state evolution, the structured role-play simulator, executable commerce tools, and all 12 reward components under ordinary PPO. Failure-aware resampling, graph–critic estimation, counterfactual replay, responsibility-based unit routing, and unit-balanced optimization are not enabled in this comparison.

All RL conditions use the same task-specific SFT initialization, task pool, optimization configuration, and interaction budget, and are evaluated on the same frozen evaluation suite. Results are obtained independently for Qwen3.5-4B, Qwen3.5-9B, and Qwen3.5-27B. The compact environment-utility results macro-average each metric equally across these three model sizes; heterogeneous benchmark metrics are never averaged together.

C Reward Design and Environment Validation

C.1 Multi-Level Reward Interface

ConsumerGym retains a 12-dimensional reward vector $r_t \in \mathbb{R}^{12}$ spanning turn-level, session-level, and holistic supervision. Turn-level components assess the quality of the current assistant decision, session-level components capture cross-turn interaction processes, and holistic components assess the final task outcome and interaction efficiency. The individual components are retained separately from policy-specific credit assignment and from the responsibility matrix used by RUS-GPO.

Table 5 summarizes the reward dimensions and their training weights. The weights sum to one and define the relative contribution of the components during training.

Scoring sources and boundaries. Factuality and price accuracy combines GPT-5.5 judgments with deterministic verifier checks at the turn boundary. State tracking is assessed by GPT-5.5 at the session boundary. Task success combines executable evidence with judge assessment at termination.

The remaining component-specific scoring configuration, prompts, and temporal return-assignment settings will be included in the code release.

Table 5: Multi-level reward components used in CONSUMERGYM. The training weights sum to 1.0.

Level	Component	Main criterion	Weight
Turn	Need understanding	Current intent, constraints, requirement changes, and cross-turn references	0.08
Turn	Factuality and price accuracy	Product facts, price calculations, tool evidence, and execution status	0.16
Turn	Recommendation relevance and selection quality	Constraint satisfaction, budget fit, candidate quality, and user feedback	0.14
Turn	Response completeness and recommendation sufficiency	Coverage of the current task, candidates, results, limitations, and next steps	0.16
Turn	Rationale usefulness	Evidence-grounded explanations of differences, trade-offs, and recommendations	0.04
Session	Clarification	Timely clarification of missing information without unnecessary repetition	0.04
Session	State tracking	Maintenance of constraints, preferences, selected/rejected items, references, and authorization	0.08
Session	Constraint updating	Correct handling of revised requirements and removal of outdated constraints	0.08
Session	Decision pacing	Appropriate sequencing of clarification, search, comparison, execution, and termination	0.03
Session	Personalized recommendation	Adaptation to historical preferences, priorities, and user feedback	0.03
Holistic	Task success	Satisfaction of final goals, active constraints, execution evidence, deliverables, and authorization	0.14
Holistic	Interaction efficiency	Information gain, economical tool use, avoidance of repetition, and timely convergence	0.02

The reward representation and the decision-unit representation serve different purposes. Reward components specify *what* aspect of the interaction is evaluated, whereas the state, plan, action, and response masks specify *where* policy-generated tokens occur. RUS-GPO subsequently uses its responsibility matrix to determine which decision units receive credit from each component.

C.2 ENVIRONMENT AND EVALUATOR VALIDATION

We validate CONSUMERGYM independently of trained-agent performance. Figure 3 summarizes seven checks of simulator behavior, execution, and replay integrity; Table 10 provides their complete numerical results. We additionally evaluate simulator plausibility, oracle-state quality, and agreement between automatic evaluators and independent human judgments.

Each domain-level audit uses 200 sampled cases. For the seven percentage checks in Table 10, each case is evaluated in three repeated runs. The reported percentage averages the binary outcomes across the 200 cases and three runs:

$$\mathrm{AuditPass}(\%) = \frac{100}{200 \times 3} \sum_{i=1}^{200} \sum_{r=1}^{3} b_{i,r}, \tag{12}$$

where $b_{i,r}$ indicates whether case i passes the check in run r. The resulting 600 case–run outcomes are repeated observations of 200 cases, not 600 distinct scenarios.

Human-evaluated quantities are separately assessed by three annotators, whose scores are averaged at the case level before domain-level aggregation. The three human judgments and the three environment runs are different replication dimensions; repeated environment runs are not additional independent human annotators.

User-simulation quality. Simulator-state consistency checks whether generated user behavior remains consistent with the currently valid profile and requirements. Controlled-update fidelity mea-

Table 6: Additional intrinsic validation of CONSUMERGYM across the three domains. Higher is better except for unrevealed-state leakage.

Metric	E-commerce	Food	Travel
Human plausibility (1–5) ↑	4.8	4.7	4.8
Oracle-state slot F1 (%) ↑	92.4	93.1	95.7
Unrevealed-state leakage (%) ↓	1.9	2.1	1.1
Turn-score agreement (ρ) ↑	0.88	0.90	0.91
Session-score agreement (ρ) ↑	0.85	0.83	0.97
Success-checker macro-F1 (%) ↑	99.18	99.33	99.38

sures whether an intended user-state change is correctly realized and retained. Disclosure compliance checks whether information is revealed only when intended, and action responsiveness measures whether the simulated user reacts appropriately to alternative assistant decisions. Human plausibility complements these structured checks with independent judgments of naturalness and scenario consistency.

Oracle-state quality. The revealed oracle state is evaluated only against information supported by the current dialogue prefix and policy-visible tool observations. Oracle-state slot F1 measures agreement between the environment-maintained revealed state and the corresponding reference state. Unrevealed-state leakage measures premature exposure of information that should remain hidden at the current interaction point.

Evaluator agreement. Turn- and session-level evaluator validity is measured using Spearman rank correlation ρ between automatic scores and the mean human judgments on the audit set. The resulting correlations are consistently high across the three domains, indicating strong agreement in the relative ranking of interaction quality. These correlations measure rank agreement rather than absolute score calibration.

The success checker is evaluated separately as a binary outcome evaluator. Success-checker macro-F1 averages the class-level F1 scores for successful and unsuccessful sessions and should therefore not be interpreted as the agent's task-success rate.

Execution and replay integrity. Tool-transition correctness verifies that tool outputs and resulting state changes match the executable environment specification, including correct rejection of invalid calls. Snapshot restoration checks whether the required semantic state is recovered after restoration, while replay-branch isolation checks that mutations in one restored branch do not affect another. Together, these checks validate the environment mechanisms required for controlled replay and counterfactual comparison.

D RUS-GPO ALGORITHM DETAILS

D.1 NOTATION AND DECISION BOUNDARIES

For session i and assistant turn t, let $o_{i,t} = (h_{i,t}, e^{\text{obs}}_{i,t})$ denote the policy-visible observation, consisting of the interaction history and policy-visible environment feedback. The actor produces the complete assistant decision

$$d_{i,t} = \left(q^{\text{state}}_{i,t}, q^{\text{plan}}_{i,t}, a_{i,t}, y_{i,t}\right), \tag{13}$$

corresponding to four decision units

$$\mathcal{D} = \{\text{State}, \text{Plan}, \text{Action}, \text{Response}\}. \tag{14}$$

An outer assistant turn may contain multiple model generations and tool calls. Policy-produced state and plan tokens belong to the State and Plan units, respectively; serialized tool-call tokens belong to the Action unit; and user-visible natural language belongs to the Response unit. User messages and tool observations are environment observations rather than actor-training tokens.

For reward component $k \in \mathcal{K}$, with $|\mathcal{K}| = 12$, let $r_{i,t}^k$ denote the component reward and w_k its training weight as defined in Appendix C. We use $G_{i,t}^k$ for the corresponding discounted rollout return:

$$G_{i,t}^k = \sum_{\ell=0}^{T_i-t-1} \gamma^\ell r_{i,t+\ell}^k. \tag{15}$$

The oracle user state, replay metadata, graph statistics, critic-side simulator information, and counterfactual outcomes are used only for training-side supervision and credit estimation. They are never appended to the actor observation at inference time.

D.2 Failure-Aware Experience Resampling

After supervised fine-tuning, RUS-GPO evaluates fresh on-policy interactions and constructs case-level diagnostic signals from observed failures. Diagnostics include unsuccessful outcomes, user-state errors, invalid or unsupported actions, and uncertainty in the policy trajectory. These signals modify *which training scenarios are sampled for new rollouts*; they do not change the correctness label or reward of an already observed action.

For case x, let

$$z_x = \left(z_x^{\text{fail}}, z_x^{\text{state}}, z_x^{\text{invalid}}, z_x^{\text{uncert}}\right) \tag{16}$$

denote its diagnostic vector, and let $P(z_x) \geq 0$ denote the resulting weakness priority. We use a mixture of the original training distribution and the priority distribution:

$$p_{\text{sample}}(x) = (1-\beta)\, p_{\text{base}}(x) + \beta \frac{P(z_x)}{\sum_{x'} P(z_{x'})}, \tag{17}$$

where β controls the strength of adaptive resampling.

The base component prevents the learner from collapsing onto a small set of currently difficult cases, whereas the priority component allocates additional rollout budget to recurring weaknesses. Resampling always produces fresh on-policy trajectories from the current rollout policy rather than duplicating previous failed trajectories. Each trajectory retains its policy checkpoint, scenario identifier, and sampling provenance so that subsequent graph and replay comparisons are performed between compatible records.

D.3 User-State Graph and Boundary Critic

Canonical revealed-state graph. At each decision boundary, the revealed oracle state is canonicalized as

$$g_{i,t} = \phi(s_{i,t}^{\text{rev}}), \tag{18}$$

where $\phi(\cdot)$ preserves disclosed requirements, active constraints, confirmed preferences, rejected candidates, and other interaction facts needed to determine decision comparability.

A graph edge records an observed transition

$$(g_{i,t}, d_{i,t}, g_{i,t+1}). \tag{19}$$

The graph is an empirical index over comparable interaction states rather than a graph neural network. Its purpose is to compare outcomes obtained from similar revealed user states across independently sampled trajectories.

Let

$$\mathcal{T}_j(g) = \{t : g_{j,t} = g\} \tag{20}$$

be the visits to state g in trajectory j. To prevent trajectories with repeated visits from dominating the baseline, component returns are first averaged within trajectory:

$$\overline{G}_{j,g}^k = \frac{1}{|\mathcal{T}_j(g)|} \sum_{t \in \mathcal{T}_j(g)} G_{j,t}^k. \tag{21}$$

For trajectory i, the leave-one-trajectory-out graph value is

$$V^k_{G,-i}(q_i, g) = \frac{1}{|\mathcal{J}_{-i}(q_i, g)|} \sum_{j \in \mathcal{J}_{-i}(q_i,g)} \overline{G}^k_{j,g}, \tag{22}$$

where $\mathcal{J}_{-i}(q_i, g)$ contains compatible trajectories other than i under the same scenario-context key q_i.

When both the current state and its observed successor are supported by the graph, the component-wise graph advantage is

$$A^{G,k}_{i,t} = r^k_{i,t} + \gamma(1 - b_{i,t})V^k_{G,-i}(q_i, g_{i,t+1}) - V^k_{G,-i}(q_i, g_{i,t}), \tag{23}$$

where $b_{i,t}$ indicates a terminal transition.

Graph confidence. The reliability of graph evidence depends on both cross-trajectory support and return dispersion. Let

$$n_{-i}(q, g) = |\mathcal{J}_{-i}(q, g)|. \tag{24}$$

For supported states, we define

$$c^k_{G,-i}(q, g) = \frac{n_{-i}(q, g)}{n_{-i}(q, g) + \tau_n} \frac{1}{1 + \widehat{\mathrm{Var}}_{j \in \mathcal{J}_{-i}(q,g)}(\overline{G}^k_{j,g})/\sigma^2_{\mathrm{ref}}}, \tag{25}$$

where τ_n controls the effect of support size and σ_{ref} determines the dispersion scale.

If the current state is supported but its successor lies at the graph frontier, we instead use

$$A^{G,k}_{i,t} = G^k_{i,t} - V^k_{G,-i}(q_i, g_{i,t}), \tag{26}$$

with reduced confidence

$$c^k_{i,t} = \lambda_{\mathrm{fr}} c^k_{G,-i}(q_i, g_{i,t}), \qquad 0 \le \lambda_{\mathrm{fr}} \le 1. \tag{27}$$

Unsupported states receive zero graph confidence and therefore fall back to the boundary critic.

Asymmetric boundary critic. The critic contains one value head for each of the 12 reward components. It receives the policy-visible interaction context together with bounded training-side scenario information, interaction progress, and commerce feasibility features. This additional information is available only to the critic and is not provided to the actor.

Future realized rewards, future tool outcomes, and counterfactual returns are excluded from critic inputs. Component-wise generalized advantage estimation produces

$$A^{C,k}_{i,t}, \tag{28}$$

which serves as a dense fallback when comparable graph evidence is sparse.

Graph and critic estimates are combined according to graph confidence:

$$A^{B,k}_{i,t} = c^k_{i,t} A^{G,k}_{i,t} + (1 - c^k_{i,t}) A^{C,k}_{i,t}. \tag{29}$$

All values used for one actor update are computed from frozen rollout-time models and remain fixed during that actor update. This prevents the target credit from drifting as the policy is optimized.

D.4 Selective Context-Matched Counterfactual Replay

Graph and critic estimates identify decision boundaries for which additional controlled comparison is potentially useful. Rather than replaying every turn, RUS-GPO selectively restores high-value decision boundaries.

A checkpoint may be prioritized according to the magnitude and confidence of its component-wise graph evidence:

$$s^{\mathrm{CF}}_{i,t} = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} |A^{G,k}_{i,t}|\, c^k_{i,t}. \tag{30}$$

For an eligible checkpoint, the environment restores the same pre-decision interaction prefix, revealed user state, latent session-plan state, catalog/tool snapshot, and other replay state. The original decision $d_{i,t}^{\mathrm{on}}$ is compared with replacement decisions $d_{i,t}^{(1)}, \ldots, d_{i,t}^{(N_{\mathrm{alt}})}$ generated from the frozen policy configuration.

Branch continuations are sampled stochastically from the restored context. For repetition j, write $\eta_{j,0}$ for the original branch's randomness and $\eta_{j,m}$ for replacement branch m. These branch-specific symbols do not impose identical random draws or independence between branches. For a branch realization η, the bounded component return of decision d is

$$Q_{i,t}^{k}(d;\eta) = \sum_{\ell=0}^{H_{d,j}-1} \gamma^{\ell} r_{i,t+\ell}^{\mathrm{CF},k}(d;\eta), \qquad H_{d,j} \le H. \tag{31}$$

The context-matched component-wise counterfactual advantage is

$$A_{i,t}^{\mathrm{CF},k} = \frac{1}{N_{\mathrm{rep}} N_{\mathrm{alt}}} \sum_{j=1}^{N_{\mathrm{rep}}} \sum_{m=1}^{N_{\mathrm{alt}}} \left[Q_{i,t}^{k}(d_{i,t}^{\mathrm{on}};\eta_{j,0}) - Q_{i,t}^{k}(d_{i,t}^{(m)};\eta_{j,m}) \right]. \tag{32}$$

Matching controls the pre-decision interaction context, catalog/tool state, and continuation-policy checkpoint. It does not require a globally shared user-simulator seed, identical random draws, or identical future observations. Different decisions can induce different user responses, tool outcomes, and state transitions. Repeated sampled continuations provide the empirical branch comparison; no common-random-number variance-reduction guarantee is asserted. Branch sampling and random-state handling will be documented in the released replay implementation.

Counterfactual evidence is fused with the graph–critic estimate only when a valid replay comparison is available:

$$\widetilde{A}_{i,t}^{k} = A_{i,t}^{B,k} + \lambda_{\mathrm{CF}} m_{i,t}^{\mathrm{CF}} A_{i,t}^{\mathrm{CF},k}, \tag{33}$$

where $m_{i,t}^{\mathrm{CF}} \in \{0,1\}$ indicates whether a valid counterfactual comparison exists.

Replacement branches are used as controls for credit estimation rather than as additional actor-training trajectories.

D.5 Dynamic Responsibility Assignment

Component-level credit identifies *which evaluation dimensions* were responsible for an outcome, but an assistant turn contains multiple policy-generated decision units. Assigning the same turn advantage to all tokens would therefore conflate errors in state tracking, planning, tool use, and user-facing response generation.

We instead use a turn-dependent responsibility matrix

$$M_{i,t} \in \mathbb{R}_{+}^{|\mathcal{K}| \times |\mathcal{D}|} = \mathbb{R}_{+}^{12 \times 4}. \tag{34}$$

Unlike a fixed global routing table, $M_{i,t}$ is recomputed from the training-side evidence available for the current interaction. Its entries therefore reflect which decision units are responsible for each reward component in the particular turn being optimized. We write

$$M_{i,t} = \mathcal{R}(E_{i,t}), \tag{35}$$

where $E_{i,t}$ denotes the interaction-specific responsibility evidence and $\mathcal{R}$ the responsibility allocator.

For interpretability and stable scaling, the active responsibility mass of each component is normalized across decision units:

$$M_{i,t}^{k,d} \ge 0, \qquad \sum_{d \in \mathcal{D}_{i,t}^{+}} M_{i,t}^{k,d} = 1, \tag{36}$$

where $\mathcal{D}_{i,t}^{+}$ contains decision units with valid actor tokens.

The resulting unit-level advantage is

$$A_{i,t}^{d} = \sum_{k \in \mathcal{K}} w_k M_{i,t}^{k,d} \widetilde{A}_{i,t}^{k}, \qquad d \in \mathcal{D}_{i,t}^{+}. \tag{37}$$

This dynamic allocation allows the same reward component to assign different responsibility across interactions. For example, a task-level failure may be attributed primarily to an incorrect state update in one trajectory, an inappropriate tool action in another, or an incomplete final response in a third.

D.6 UNIT-BALANCED POLICY OPTIMIZATION

Let $\mathcal{T}_i^+$ denote active assistant turns in session i, $\mathcal{D}_{i,t}^+$ the decision units containing actor tokens, and $n_{i,t,d}$ the number of valid tokens in unit d.

For actor token x with its actual generation prefix h, the rollout-policy importance ratio is

$$\rho_\theta(x, h) = \exp\left[\log \pi_\theta(x \mid h) - \log \pi_{\text{old}}(x \mid h)\right]. \tag{38}$$

For PPO, a token-level clipped surrogate is

$$\ell_{\text{PPO}}(\theta; x, h, A) = -\min\left\{\rho_\theta A,\ \text{clip}\left(\rho_\theta, 1-\epsilon_{\text{lo}}, 1+\epsilon_{\text{hi}}\right) A\right\}. \tag{39}$$

Rather than averaging all actor tokens globally, RUS-GPO first normalizes within each decision unit:

$$L_{i,t,d} = \frac{1}{n_{i,t,d}} \sum_{x \in \mathcal{X}_{i,t,d}} \ell_{\text{PPO}}\left(\theta; x, h_x, A_{i,t}^d\right). \tag{40}$$

The active units are then combined at the turn level:

$$L_{i,t} = \frac{\sum_{d \in \mathcal{D}_{i,t}^+} \alpha_d L_{i,t,d}}{\sum_{d \in \mathcal{D}_{i,t}^+} \alpha_d}, \tag{41}$$

where α_d controls the relative contribution of each decision-unit type.

Finally, turns and sessions are averaged hierarchically:

$$L_{\text{actor}} = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \frac{1}{|\mathcal{T}_i^+|} \sum_{t \in \mathcal{T}_i^+} L_{i,t}. \tag{42}$$

This objective prevents a long response or action serialization from receiving greater optimization weight solely because it contains more tokens. Empty decision units are excluded from $\mathcal{D}_{i,t}^+$ and the remaining active-unit weights are renormalized through Eq. (41).

D.7 PPO AND GRPO INSTANTIATIONS

RUS-GPO separates structured experience collection and credit assignment from the final policy optimizer. We evaluate both PPO- and GRPO-based instantiations.

Both variants use the same CONSUMERGYM trajectory interface, failure-aware resampling, revealed-user-state graph, component-wise credit representation, selective replay interface, and dynamic four-unit responsibility assignment. They differ in how the resulting policy-update advantages are normalized and used by the underlying RL optimizer.

For the PPO instantiation, the unit-level advantages in Eq. (37) are optimized with the clipped objective in Eq. (39).

The GRPO instantiation uses an 8×8 grouping configuration. Group construction and normalization follow the defaults of the training implementation. Trajectories in an optimization group are compared using that implementation's group-relative normalization, with the component and unit decomposition applied before the token-level actor update. The release will provide the configuration keys defining both grouping dimensions, resolved defaults, and the PPO/GRPO-specific critic and replay settings.

Thus, PPO and GRPO provide alternative optimization backbones while the state-aware experience collection and fine-grained responsibility assignment remain shared components of RUS-GPO.

Algorithm 1 RUS-GPO training procedure

Require: SFT-initialized policy π_θ, training scenarios, CONSUMERGYM environment, and RUS-GPO configuration

Evaluate the SFT policy and construct case-level failure diagnostics
Initialize the failure-aware scenario sampler
while the training budget remains **do**
    Freeze the current rollout policy π_{old}
    Sample scenarios using the failure-aware mixture
    Collect fresh on-policy interaction trajectories
    Record decisions, rewards, state events, unit masks, and snapshots
    Canonicalize revealed user states and construct graph statistics
    Estimate component-wise graph and boundary-critic advantages
    Fuse graph and critic evidence according to graph confidence
    Select high-value decision boundaries for context-matched replay
    Compute valid component-wise counterfactual comparisons
    Fuse counterfactual evidence with graph–critic credit
    Compute the dynamic responsibility matrix for each active turn
    Route component credit to State, Plan, Action, and Response units
    Optimize the actor with unit-balanced PPO or GRPO updates
    Update auxiliary estimators and refresh diagnostics as scheduled
end while
return trained actor policy π_θ

D.8 OVERALL TRAINING PROCEDURE

Algorithm 1 summarizes the complete training pipeline.

At deployment time, only the actor and the ordinary CONSUMERGYM-compatible observation interface are required. Oracle user states, graph statistics, boundary-critic side information, replay snapshots, counterfactual branches, and responsibility supervision are training-time mechanisms and are not exposed to the deployed policy.

D.9 IMPLEMENTATION CONFIGURATION AND RELEASE

The public code release will include the experiment-specific configuration files and resolved defaults used for both optimizer variants. These cover the failure-priority function and refresh schedule, sampling mixture, graph support and confidence parameters, counterfactual branch counts and horizon, fusion coefficient, dynamic responsibility allocator and its inputs, unit weights, PPO clipping, GAE settings, and GRPO grouping. The release will also document software versions, checkpoint identifiers, and random-state handling. The symbolic definitions above specify the algorithmic interfaces; the accompanying code and configurations will specify their concrete instantiation.

E COMPLETE RESULTS AND ADDITIONAL ANALYSES

E.1 CROSS-SCALE RESULT SUMMARY

Tables 1 and 2 in the main paper report the complete per-backbone results for all compared methods on Qwen3.5-4B, Qwen3.5-9B, and Qwen3.5-27B. To provide a compact cross-scale view, Table 7 reports the equal-weight macro-average across the three backbone sizes for each metric independently. Metrics from different benchmarks are never averaged together.

E.2 FOUR-MODULE ABLATION PROTOCOL

We conduct PPO-based ablations over the four major components of RUS-GPO. Each ablation removes one method component from the full RUS-GPO (PPO) configuration while retaining the remaining components whenever possible. All variants are independently trained on Qwen3.5-

Table 7: Cross-scale summary obtained by equal-weight averaging across Qwen3.5-4B, Qwen3.5-9B, and Qwen3.5-27B for each metric separately. Complete per-backbone results appear in Tables 1 and 2.

Method	FC P@1	CG Success	SRB Hard	Shop.	Delivery	In-store	OTA
Base Model	37.63	27.0	62.40	0.250	47.58	46.17	31.00
SFT Backbone	49.67	33.4	65.57	0.347	59.92	56.33	39.17
PPO	58.60	37.8	68.33	0.387	65.83	59.92	44.67
GRPO	53.13	35.2	66.43	0.353	61.42	57.92	41.17
GiGPO	55.93	36.3	67.20	0.387	62.50	59.75	42.42
HGPO	58.23	36.7	67.93	0.403	62.50	59.33	44.17
G2PO	58.37	37.3	69.63	0.413	65.83	60.08	45.75
ReBel	59.97	39.2	70.90	0.443	66.42	61.08	48.83
RUS-GPO (GRPO)	62.70	40.4	74.40	0.497	69.50	63.33	50.75
RUS-GPO (PPO)	**65.13**	**44.1**	**76.70**	**0.530**	**72.83**	**66.50**	**54.83**

Notes. FC P@1 denotes fixed-context Avg. P@1; CG Success denotes CONSUMERGYM role-play task success (%); SRB Hard denotes the Hard-MT score of ShoppingReasoningBench; Shop. denotes ShopSimulator R_{succ}. Macro-averages are computed from the reported per-backbone results and should not be interpreted as estimates of training-seed variability.

4B, Qwen3.5-9B, and Qwen3.5-27B. The tables below report the equal-weight per-metric macro-average across the three backbone sizes.

Without failure-aware resampling. We restore the original scenario-sampling distribution and disable weakness-based case prioritization. The training task pool and actor rollout budget remain unchanged. Fresh trajectories are still collected on-policy, so the realized rollouts need not match those of the full method.

Without graph–critic estimation. We remove both user-state graph credit estimation and boundary-critic fusion. Standard component-wise PPO/GAE is used as the replacement advantage estimator. Counterfactual replay remains enabled so that the ablation isolates the contribution of graph–critic estimation.

Because the full counterfactual selector uses graph evidence, replay checkpoints in this variant are instead prioritized according to the magnitude of the component-wise GAE estimate:

$$s_{i,t}^{\text{CF-GAE}} = \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \left| A_{i,t}^{\text{GAE},k} \right|. \tag{43}$$

The counterfactual replay budget is otherwise kept unchanged.

Without counterfactual replay. We disable context-matched replacement branches and remove the corresponding counterfactual credit correction. Failure-aware resampling, graph–critic estimation, and structured unit credit remain enabled. The actor rollout budget is kept identical to the full method. Interactions saved by removing counterfactual branches are not reallocated to additional on-policy rollouts; consequently, this ablation uses fewer total environment interactions than the full configuration.

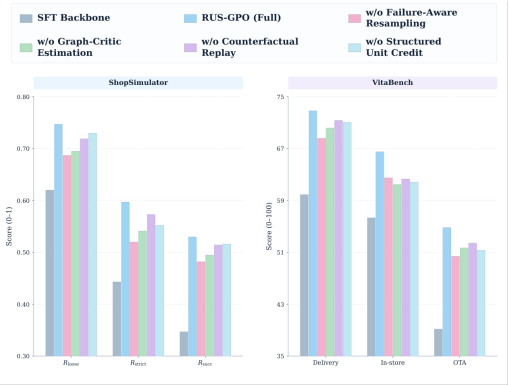
Without structured unit credit. We remove both dynamic responsibility assignment and unit-balanced optimization. A single turn-level advantage is broadcast to all actor tokens in the turn and optimized using the standard PPO aggregation. This is therefore a joint ablation of responsibility-based routing and unit-wise loss balancing rather than an attempt to separate the two mechanisms.

E.3 PPO-BASED COMPONENT ABLATIONS

Figure 4 visualizes the four component ablations. Table 8 reports ablations on the internal fixed-context evaluation, CONSUMERGYM role-play, and ShoppingReasoningBench. Table 9 reports the



(a) Internal evaluations and SRB.



(b) ShopSimulator and VitaBench.

Figure 4: Component ablations of RUS-GPO (PPO). Scores are equally averaged across Qwen3.5-4B, 9B, and 27B for each metric. Full denotes the complete method, and each ablated variant removes the indicated component; the SFT backbone is included as a reference. Internal evaluations comprise fixed-context diagnostics and ConsumerGym role-play; SRB denotes ShoppingReasoningBench. Higher scores are better. Y-axes are truncated.

Table 8: PPO-based component ablations on internal fixed-context evaluation, CONSUMERGYM role-play, and ShoppingReasoningBench. Results are equal-weight macro-averages across Qwen3.5-4B/9B/27B. Higher scores are better.

Variant / Reference	**Internal Fixed-Context**		**CONSUMERGYM Role-play**			**ShoppingReasoningBench**	
	Avg. P@1	Avg. P@3	Turn	Session	Success (%)	Full-MT	Hard-MT
SFT Backbone	49.67	75.70	0.516	0.412	33.37	80.2	65.6
ReBel	59.97	81.67	0.586	0.498	39.17	83.4	70.9
RUS-GPO (Full)	**65.13**	**86.00**	**0.636**	**0.547**	**44.10**	**87.3**	**76.7**
w/o Failure-Aware Resampling	61.58	82.51	0.597	0.514	41.28	84.6	72.8
w/o Graph–Critic Estimation	62.70	83.24	0.610	0.529	42.49	85.9	73.4
w/o Counterfactual Replay	63.25	84.59	0.622	0.530	43.57	85.3	75.8
w/o Structured Unit Credit	62.36	83.62	0.619	0.502	40.82	84.8	74.3

Notes. Full denotes the complete RUS-GPO (PPO) method. Each ablated variant removes the indicated module according to Section E.2. Turn and Session scores use a 0–1 scale; fixed-context, CONSUMERGYM Success, and ShoppingReasoningBench scores use a 0–100 scale. ShoppingReasoningBench values are shown to one decimal place.

corresponding results on ShopSimulator and VitaBench. SFT and ReBel are included as reference methods; the four ablated variants are derived from the full RUS-GPO (PPO) configuration described above.

Ablation summary. Removing any of the four components reduces performance relative to the full RUS-GPO (PPO) configuration across the reported evaluations. Failure-aware resampling contributes to performance across both internal and external tasks, while removing graph–critic estimation reduces the quality of state-aware credit assignment. Counterfactual replay provides additional gains on top of graph–critic estimation, and removing structured unit credit causes particularly visible degradation in session-level interaction quality and task success. Together, the results support the complementary roles of adaptive experience collection, state-aware credit estimation, controlled replay, and fine-grained decision-unit attribution.

Table 9: PPO-based component ablations on ShopSimulator and VitaBench. Results are equal-weight macro-averages across Qwen3.5-4B/9B/27B. Higher scores are better.

Variant / Reference	ShopSimulator			VitaBench		
	R_{loose}	R_{strict}	R_{succ}	Delivery	In-store	OTA
SFT Backbone	0.620	0.443	0.347	59.9	56.3	39.2
ReBel	0.663	0.493	0.443	66.4	61.1	48.8
RUS-GPO (Full)	**0.747**	**0.597**	**0.530**	**72.8**	**66.5**	**54.8**
w/o Failure-Aware Resampling	0.687	0.520	0.482	68.6	62.5	50.4
w/o Graph–Critic Estimation	0.695	0.541	0.495	70.2	61.5	51.7
w/o Counterfactual Replay	0.719	0.573	0.514	71.4	62.3	52.4
w/o Structured Unit Credit	0.730	0.552	0.516	71.0	61.8	51.3

Notes. ShopSimulator metrics use their native 0–1 scales. VitaBench scores are reported on their original 0–100 scales. Full denotes the complete RUS-GPO (PPO) configuration.

Table 10: Intrinsic checks of CONSUMERGYM, averaged over 200 cases per domain and three runs per case. Values are percentages; all rows assess the environment, not the agent. Audit definitions and additional human/oracle assessments appear in Appendix C.2.

Metric	E-commerce	Food	Travel
Simulator-state consistency	96.17	98.33	95.00
Controlled-update fidelity	97.83	96.67	95.33
Disclosure compliance	95.67	98.83	97.33
Action responsiveness	98.17	97.50	97.67
Tool-transition correctness	98.67	99.17	99.00
Snapshot restoration match	99.33	98.50	99.33
Replay-branch isolation	98.83	99.00	98.17

E.4 ENVIRONMENT VALIDATION AND TRAINING UTILITY

The intrinsic environment checks reported in Table 10 evaluate simulator behavior, tool execution, snapshot restoration, and replay isolation independently of agent performance. Additional human, oracle-state, and evaluator-agreement audits are described in Appendix C.2.

The environment-controlled training comparison in Table 11 uses ordinary PPO rather than RUS-GPO. As detailed in Appendix B.6, the SFT reference and full CONSUMERGYM rows correspond to the macro-averaged SFT and ordinary-PPO results reported in the main experiments, whereas the intermediate rows isolate specific environment design choices.

No averaging is performed across heterogeneous benchmark metrics in either the main result summary, the component ablations, or the environment-utility comparison.

Table 11: Training utility under ordinary PPO, macro-averaged across Qwen3.5-4B/9B/27B. Only the indicated environment configuration changes; RUS-GPO-specific enhancements are disabled. Higher scores are better.

Training environment	**ConsumerGym**	**Shop.**	**Delivery**	**In-store**	**OTA**
SFT reference (no RL)	33.4	0.35	59.9	56.3	39.2
Prompt-only role-play + tools	33.8	0.36	60.9	57.8	40.2
ConsumerGym: static goals	34.3	0.36	61.3	58.0	41.4
ConsumerGym: terminal-only reward	35.0	0.37	62.1	58.4	42.2
ConsumerGym: full	**37.8**	**0.39**	**65.8**	**59.9**	**44.7**

Notes. ConsumerGym: success (%); Shop.: R_{succ}; remaining columns: VitaBench scores on their original scale, rounded to one decimal place. The SFT reference has no RL training. Controls and aggregation are specified in Appendix B.6.