Small Agents, Big Gains: Journey-Aware and Critic-Guided Simulation for Long-Horizon Shopping Dialogues

Qing Ping[*]
Amazon
pingqing@amazon.com

Changyou Chen
Amazon
cchangyou@gmail.com

Binxuan Huang[†]
Amazon
binxuanhuang@gmail.com

Abstract

Modern e-commerce assistants must go beyond simple product search to support inspiration, comparison, and tool-grounded fact-checking across non-linear shopping journeys. However, distilling these complex behaviors into efficient, deployable models is bottle-necked by a lack of post-training data: trajectories must cover diverse agentic workflows with high fidelity, yet the desired outputs are open-ended without a single ground truth. We propose a closed-loop Multi-Agent Simulation Framework to synthesize diverse, faithful, and policy-aligned shopping trajectories. The system orchestrates a journey-aware, stateful user simulator to drive exploration, a shopping agent that manages both tools and UI elements, and a critic agent that provides rubric-driven feedback to iteratively refine the data. On a domain-specific benchmark, this synthetic data enables a small model to significantly outperform same-size baselines and surpass a large-model baseline, making it viable to deploy the smaller backbone at 8× higher inference throughput than the large baseline.

1 Introduction

Traditional e-commerce search engines mainly serve the "tip of the iceberg": cases where customers already have clear purchase intent (e.g., searching for "*Nike Air Max size 6*)". Consumer research, however, has long framed shopping as a multi-stage cognitive problem-solving process (Engel et al., 1990). Users may seek inspiration (e.g., gift ideas) or iteratively compare and verify information (e.g., "*How is its battery life compared to others?*") before committing to a purchase. These under-supported stages motivate the industry shift toward agentic shopping assistants (Perplexity, 2024; Amazon, 2024; Walmart, 2025; OpenAI, 2025; Google, 2025). Although frontier models provide strong general capabilities, production deployment often favors specialized small models for lower cost and latency, and for stricter compliance with proprietary tools and policies.

Teaching these versatile behaviors to smaller specialized models is fundamentally a data-coverage and verification problem: (i) training trajectories must cover diverse user behaviors across domains and shopping stages, and (ii) target responses are often open-ended with no single reference answer for supervision. Effective data synthesis must therefore capture: (1) *diversity*: broad coverage over product domains and shopping journey stages from vague inspiration, to multi-option comparison to precise fact-checking; (2) *fidelity*: stateful, interactive, non-linear user simulation that reflect real shopping behavior rather than scripted turns; and (3) *quality alignment*: responses that jointly optimize multiple objectives, such as helpfulness, factuality, policy compliance, and UI presentation.

To address this, we propose a closed-loop Multi-Agent Simulation Framework for synthesizing high-fidelity, policy-aligned shopping trajectories. Our contributions are threefold: (1) **Journey-Aware User Simulation**: We orchestrate a stateful user simulator that explicitly models the non-linear exploration patterns of real-world shopping journeys. This user simulator advances the shopping journey via interacting with a shopping agent who operates in a hybrid tool-and-UI action space; (2) **Textual Feedback as Gradient**: We introduce a critic agent that converts static business rubrics into iterative, corrective feedback, serving as a "textual gradient" to refine trajectories during synthesis; and (3) **Efficient Distillation**: We demonstrate that this data synthesis loop closes the quality gap between a small student and a larger teacher on complex shopping tasks; because the deployed backbone is smaller, serving it yields 8× higher inference throughput than the larger baseline.

[1] *Corresponding author.
[2] †Work was done when the author was with Amazon.

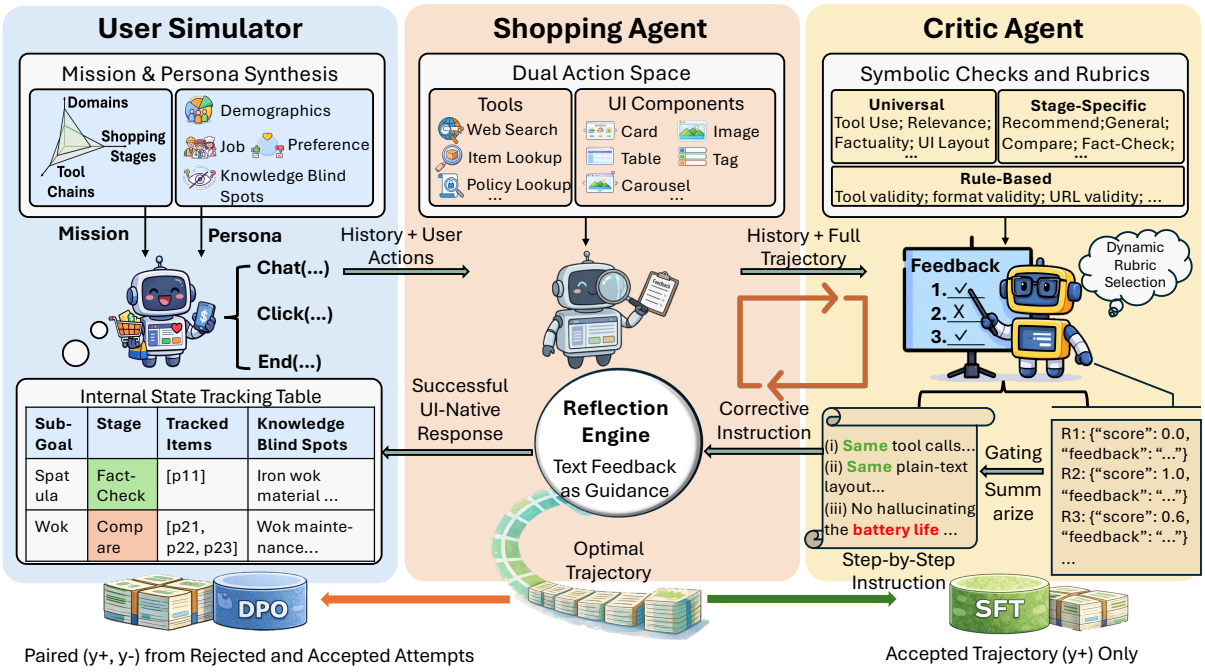


Figure 1: Multi-Agent Data Synthesis Framework.

2 Related Work

Data Synthesis for Shopping Agents. Most existing synthesis methods frame the shopping agent task as a closed-world problem over small product catalogs, enabling structured approaches such as decision-tree construction or graph traversal for product search (Li et al., 2025a; Leszczynski et al., 2023; Xiao et al., 2021). More recent work (Gao et al., 2026) employs multi-agent workflows to synthesize shopping agent trajectories, but still relies on small, closed-world catalogs to define verifiable rewards. While human-annotated datasets offer more realistic scenarios (Zou et al., 2025; Bernard and Balog, 2023), they are expensive to curate at scale. LLM-assisted approach (Luo et al., 2025) improves scalability but is designed around templated responses. In contrast, we target synthesizing trajectories for open-ended queries over millions of products, where verifiable reward ground-truth is infeasible and the responses must be adaptive to the multi-turn conversation.

User Simulators. LLM-based user simulators enable scalable trajectory synthesis without relying on private user logs. However, studies show that simple prompt-driven simulators often suffer from over-cooperative goal revelation, role drift, and goal misalignment (Naous et al., 2026; Shim et al., 2026; Wang et al., 2025; Mehri et al., 2025). Prior methods mitigate these issues through generator-verifier loops or fine-tuning (Luo et al., 2024; Sekulic et al., 2024), but are mainly validated on rigid slot-filling settings rather than open-ended shopping conversations. A parallel line of work targets user simulation for recommendation systems (Afzali et al., 2023; Zhang et al., 2025; Bougie and Watanabe, 2025; Zhan et al., 2023), simulating structured preference signals instead of content-rich, open-ended, multi-stage shopping journeys.

Critique, Reflection, and Textual Feedback. Recent work improves agent behavior at inference time by iteratively revising responses based on self-reflection or receiving natural-language critiques (Shinn et al., 2023; Madaan et al., 2023; Gou et al., 2024; Li et al., 2025b). Compared to scalar rewards, textual feedback is more interpretable and prescriptive, enabling targeted revisions without model weight updates. The "textual gradient" formulation further cast it as directional updates in text space (Yuksekgonul et al., 2024; Lee et al., 2025). Inspired by this, our critic agent produces rubric-driven corrective feedback in a continuous refinement loop until acceptance, yielding high-quality preference pairs for post-training.

3 Trajectory Synthesis via Multi-Agent Simulation

Synthesizing training data for an open-ended, multi-turn shopping assistant is a sequential

decision-making problem with two key properties. First, the assistant only observes the dialogue history and tool outputs, not the user's internal states. Second, open-ended shopping responses rarely admit a unique reference answer, making scalar reward signals less effective than textual critiques that localize errors in responses against qualitative rubrics. We therefore formulate trajectory synthesis as a Partially Observable Markov Decision Process (POMDP), which we define below and instantiate as a closed-loop multi-agent simulation in 3.2.

3.1 Problem Formulation

We formulate the shopping task as a Partially Observable Markov Decision Process (POMDP) defined by the tuple $\langle S, A, \Omega, T, R \rangle$. The global state S at turn t is $s_t = (m, z_t, h_t, x_t)$, where m is the user mission, z_t is the user's latent journey state, h_t is the observable dialogue history, and x_t is the E-commerce environment (e.g., tools and databases). We explicitly model the User Simulator as a stochastic part of the environment: it observes (m, h_t) to drive state transitions (T) in z_t. Consequently, the Shopping Agent observes only the $o_t = (h_t, \text{evidence}(x_t)) \in \Omega$ but has no direct access to m or z_t. Its action space is hybrid ($a_t \in A = A_{\text{tool}} \times A_{\text{UI}}$), which allows for interleaved tool execution and UI-native response generation.

Crucially, the reward function R does not return a scalar for gradient-based policy updates. Instead, the Critic Agent evaluates each candidate action against business-derived rubrics and returns a pair (accept/reject, textual critique). On rejection, the simulation freezes at turn t, and the Shopping Agent iteratively refines a_t conditioned on accumulated critique until accepted, at which point the User Simulator reads refined a_t and advances to z_{t+1} via T.

3.2 Multi-Agent Simulation

We instantiate the POMDP as a closed-loop simulation with three specialized agents (Figure 1), each realizing a component of the formalism: the User Simulator enacts the latent state dynamics z_t via T, the Shopping Agent is the policy π_θ under study, and the Critic Agent realizes R. The remainder of this section details the mission sampling process (3.2.1), each agent (3.2.2 - 3.2.4), and the full simulation loop (3.2.5).

3.2.1 Mission Generator

Before simulation, we build a factorized mission space along three axes: **product domains**, **user persona**, and **tool-use complexity**. Instead of single-product task synthesis, we prompt a teacher LLM to iteratively construct a daily-life mission ontology and sample leaf mission skeletons that each cover at least k product domains, yielding more natural cross-domain shopping scenarios. We then instantiate each task skeleton by adding sub-goals, constraints, and preferences. Personas are mission-specific, including preference and knowledge gaps that directly influence user-simulator strategies. Finally, we vary task difficulty via sampling diverse multi-step tool chains as task skeletons. Each sampled configuration yields a mission context that initializes the user simulator (Appendix I.1).

3.2.2 User Simulator

Naive prompt-based user simulators suffer from: (i) *over-revelation* (dumping most requirements upfront), (ii) *role drift* (slipping into an assistant-like voice as context grows longer), and (iii) *premature termination* (losing track of each sub-goal by ending early without deeper exploration, evaluation and fact-check for each sub-goal).

Journey-Aware State Tracking. To mitigate these issues, we model the simulator as a stateful, goal-directed agent that follows a staged shopping funnel inspired by consumer behavior studies (Engel et al., 1990): `Broad-Search` → `Refined-Search` → `Focused-Search` → `Evaluation` → `Fact-Check` → `Done`. We enforce this behavior with a **structured thinking template**. Before generating each user turn, the simulator maintains an internal status table with one row per sub-objective. Each row records: (1) the sub-goal to be achieved, (2) a running shortlist of candidate items, (3) the current funnel stage for that sub-goal, and (4) any unresolved knowledge gaps or questions that must be answered to proceed. The simulator updates this table turn-by-turn to go back and forth across the shopping funnel stages. It is encouraged to progressively disclose constraints, maintain a consistent user role, and prevent early stopping; it cannot end the conversation until all sub-goals are explicitly marked DONE (Appendix I.3).

Action Space. Conditioned on the mission specification, persona, dialogue history, the internal status table, and a stage-transition policy in the system prompt, the simulator selects one of three ac-

tions: `click` (inspect one of the assistant-presented items), `chat` (ask a natural-language question to refine requirements), or `end` (only when all sub-goals are complete).

Validation. Expert inspection shows our simulator outperforms prompt-based baseline with less role drift, better sub-goal completion, and more diverse shopping stages with coherent transitions. (Appendix-A).

3.2.3 Shopping Agent

The Shopping Agent serves as the *data-generation teacher*. Unlike the User Simulator, whose persona varies by mission, it maintains a fixed persona: a helpful, factual, and policy-compliant assistant, covering the shopping lifecycle on any E-commerce website, from inspiration and exploration to recommendation, comparison, and post-purchase support.

Dual Action Space. At each turn, the agent (i) calls tools to gather evidence (e.g., product search, policy lookup, occasional web search), (ii) plans a UI layout (e.g., [`Text`, `Carousel`, `Tags`]), and (iii) generates a well-elaborated response with populated UI components, grounded in retrieved facts.

Behavior Guidelines. The Shopping Agent follows three system-level guidelines: (1) evidence-seeking (use tools when facts are uncertain rather than relying on its base knowledge), (2) cognitively-efficient layout (choose UI components that fit the user goal and the retrieved evidence, e.g., tables for comparisons vs. carousels for exploration), and (3) constraint satisfaction (remain helpful while complying with safety and business policies).

Reflection Engine. To address inevitable failures (e.g., hallucination, sub-optimal recommendations, policy violations), we implement a critic-guided reflection engine. When a turn is rejected by the critic agent, the simulation "freezes" at time t. The Shopping Agent is conditioned on the accumulated critiques (treated as a *textual gradient*) to localize errors, re-plan tool usage, and regenerate the UI-native response. This enables iterative refinement from trial-and-error to policy-compliant, high-quality trajectories (see example in Appendix-I.4).

3.2.4 Critic Agent

The Critic Agent is the quality-control component in the simulation loop. It combines LLM-as-judge with symbolic checkers, acting as both (i) a gate that accepts/rejects each turn and (ii) a coach that provides textual actionable feedback to follow.

Algorithm 1 Multi-Agent Simulation Loop

Require: Mission Space $\mathcal{C}$, Max Turns T_{max}, Retry Limit K

Initialization:
Sample Mission $m, p \sim \mathcal{C}$
Initialize User State $z_0 \leftarrow \text{InitState}(m)$, History $h_0 \leftarrow \emptyset$
for $t = 1$ **to** T_{max} **do**
  User Step:
  $z_t \leftarrow \text{UpdateState}(z_{t-1}, h_{t-1})$
  $a_t^u \leftarrow \pi_{user}(m, z_t, h_{t-1})$
  Append a_t^u to h_t
  if $a_t^u ==$ `end_conversation` **then**
    break
  end if
  Assistant + Critic Step (Reflection Loop):
  $k \leftarrow 0, \mathcal{I}_{corr} \leftarrow \emptyset$
  repeat
    $a_t^{ag} \leftarrow \pi_{assistant}(h_t, \mathcal{I}_{corr})$ {Plan Tools + UI}
    Scores S, Critiques $\mathcal{E} \leftarrow \pi_{critic}(h_t, a_t^{ag})$
    if $\forall s \in S, s \geq \tau$ **then**
      break {Trajectory Accepted}
    end if
    $\mathcal{I}_{corr} \leftarrow \mathcal{I}_{corr} \cup \text{Summarize}(\mathcal{E})$
    $k \leftarrow k + 1$
  until $k \geq K$
  Append a_t^{ag} to h_t { Gold Trajectory}
end for
return Trajectory h_t

Evaluation Facets/Rubrics. To handle heterogeneous requests, the Critic evaluates a set of quality facets. We use **universal facets** on every turn (relevance, conciseness, UI/layout quality, tool selection, and context carryover), and **stage-specific facets** are only selected by the Critic Agent when relevant (e.g., recommendation quality, comparison quality, and fact-check accuracy). Each facet is operationalized as a set of weighted, binary rubrics (pass/fail) with natural-language descriptions.

LLM-as-Judge and Symbolic Verification. For each facet on each turn, the LLM judge receives conversation history, the tool trajectory and UI-native response, and the selected rubric definitions, and returns both rubric scores and feedback explaining root causes and required fixes. Further, we also run symbolic checkers such as tool-call validity, response-format correctness and so on, which are deterministic programs that return $s \in \{0, 1\}$ with explicit error messages upon violation.

Gating and Verbal Feedback Loop. A turn is accepted only if all selected facets pass. Upon failure, the Critic agent aggregates rubric violations and symbolic errors into a structured step-by-step corrective instruction. We treat this instruction as

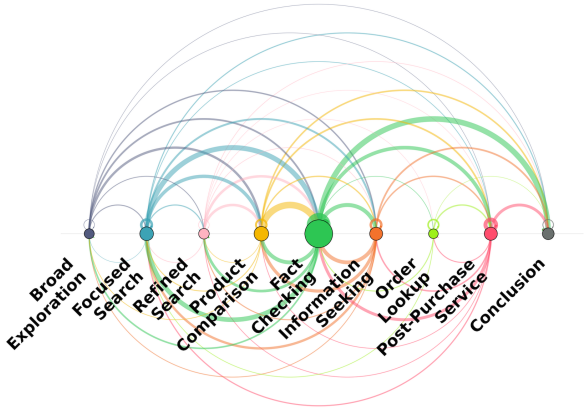


Figure 2: **Shopping Stage Transition Graph**. Nodes represent shopping stages. Edges indicate transitions from same-color source node. Arcs above the horizontal axis denote forward progression (left-to-right), while arcs below the axis denote backward revisit (right-to-left). Edge thickness reflects the transition probability.

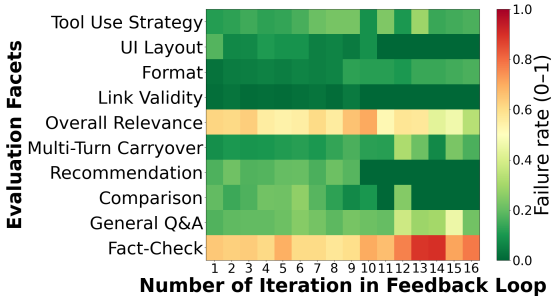


Figure 3: Failure Rate Heatmap Across Feedback Iteration Steps. x-axis: Iteration Steps. y-axis: Facets. Value: Failure Rate = Failed Rubrics / All Rubrics.

a textual gradient (Yuksekgonul et al., 2024): unlike scalar rewards, it provides directional semantic guidance for the agent to iteratively localize and fix errors until fully acceptance.

SOP-Driven Rubric Construction. Shopping assistance is inherently open-ended. There is rarely a unique "correct" solution. We therefore encode product-design principles and business requirements as explicit rubrics rather than optimizing for a fixed ground-truth. Rubrics are built with a human-in-the-loop pipeline: domain experts curate Standard Operating Procedures (SOPs) specifying desired behaviors; we decompose SOPs into facets and synthesize rubric candidates; experts then tune the rubrics by calibrating on a held-out set until Critic agent's judgments are consistently aligned with human judgments (see Appendix-B), while symbolic constraints are validated via unit tests.

3.2.5 Full Simulation Loop

Algorithm 1 summarizes the full loop. Each turn, the User Simulator updates z_t and emits a_t^u (lines 6–8); the Shopping Agent drafts a response scored by the Critic Agent against rubrics (lines 14–16), revising up to K times if rejected until all scores exceed τ (lines 17–19). The accepted response becomes the gold trajectory turn.

4 Experiments

4.1 Training Data

Dataset Profile We collect multi-agent simulation trajectories as in-domain data. The dataset covers a diverse set of roughly 70 tool-combinations and over 300 domains and personas, capturing complex conversations with 10 turns on average.

Figure 2 visualizes the state transitions within these conversations. The graph highlights the non-linear nature of the shopping process. The user simulator models both forward progression towards purchase (arcs above the horizontal axis) and backtracking to earlier stages (arcs below the axis). This structure captures realistic behavior where, for example, when encountering unsatisfactory information during fact-checking, users return to the search or comparison phases to re-explore options.

Feedback Loop Dynamics Figure 3 illustrates the failure rates across feedback iterations temporally. We observe that symbolic-check constraints (e.g., link validity, UI formatting) exhibit consistently low failure rates and stabilize during refinements rapidly. In contrast, semantic failures such as fact-checking, relevance, and context carryover, are significantly more persistent, requiring a higher number of iterations to resolve.

4.2 Evaluation

Evaluation Benchmark. We evaluate on a held-out Helpfulness benchmark of expert-curated diverse multi-turn conversations with turn-level, question-specific rubrics independent of our Critic rubrics (Appendix-C). We report LLM-judged overall helpfulness and sub-metrics (intro, recommendation, comparison), plus rule-based quality check.

Baselines. We compare against baselines that vary in data recipe and training stages:

- **SFT/General + Manual In-Domain Data**. The SFT stage uses a mixture of general instruction-following data and a small set of manually curated in-domain single-turn trajec-

Stage / Domain Data	Throughput (Model) ↑	Δ Helpfulness ↑	Δ Intro ↑	Δ Recommendation ↑	Δ Verifiable Rec ↑	Δ Comparison ↑
SFT / Manual (Δ Base)	8x (Small)	0.0%	0.0%	0.0%	0.0%	0.0%
DPO / N/A	8x (Small)	-3.4%	-0.7%	-8.9%	+2.0%	-4.9%
RL / Syntehsized	8x (Small)	+1.2%	-2.5%	-1.4%	**+5.7%**	+7.5%
SFT / Manual	1x (Large)	-0.2%	-3.7%	+49.6%	-5.6%	-4.6%
DPO / N/A	1x (Large)	+14.9%	-0.4%	+57.3%	-10.8%	+6.2%
RL / Syntehsized	1x (Large)	+18.7%	-0.4%	**+77.2%**	-1.0%	**+11.1%**
SFT / Multi-Agent (Ours)	8x (Small)	**+24.4%**	-1.4%	+73.2%	**+5.9%**	+7.9%
DPO / Multi-Agent (Ours)	8x (Small)	**+28.3%**	**+1.1%**	**+74.9%**	+2.7%	**+13.2%**

Table 1: Main evaluation results. All Δ values are relative to Row 1 ($100 \times (x_i - x_0)/x_0$). **Bold** denotes best and underline denotes second-best per metric. Rows 1-3: Small Baselines. Rows 4-6: Large baselines. Rows 7-8: Ours.

tories. This baseline lacks user-simulator driven multi-turn interactions, and trajectory diversity and quality depend on manual curation efforts.

- **DPO/General Data**. The DPO stage uses general instruction-following preference pairs without in-domain data, where the model are initialized from the baseline SFT checkpoints.
- **RL/General Data + Synthesized In-Domain Data**. The RL stage uses internally curated multi-turn conversations generated by single-prompt synthesis without user simulation. It initializes the policy model from baseline DPO checkpoints and optimizes it using LLM-as-judge scores and symbolic checks, serving as our strongest internal baseline.

For our proposed method, we mix the same general instruction-following data with our multi-agent simulation data: accepted trajectories for SFT and accepted–rejected pairs for DPO on a small model. Baseline data recipes are used to fine-tune both small and large models (Appendix-F).

4.3 Experiment Results

4.3.1 Main Results

Table 1 shows that our simulation data substantially strengthens the small models: our small models gain 24.4% (SFT) and 28.3% (DPO) improvements over same-size baselines on the overall `Helpfulness` metric, and surpasses the strongest large RL model by 9.6%. The gains are not only in the overall score: our DPO model reaches large-model level on `Recommendation` and even exceeds it on `Intro`, `Verifiable-Rec` and `Comparison`.

Across training stages, large models improve from SFT→DPO→RL, whereas the small baseline drops from SFT to DPO (-3.4%). One plausible explanation is that DPO baseline data recipe uses only general-domain preference pairs, and without in-domain supervision smaller models are more susceptible to distribution shift and forgetting. When we add simulation-derived in-domain preference pairs, small-model DPO recovers and becomes the best variant.

Stage / Domain Data	Model	Δ Error Reduction ↑
SFT / Manual (Δ Base)	Small	0.0%
DPO / N/A	Small	**+52.7%**
RL / Synthesized	Small	+42.1%
SFT /Manual	Large	+78.9%
DPO / N/A	Large	**+94.9%**
RL / Synthesized	Large	+62.8%
SFT / Multi-Agent (Ours)	Small	+89.9%
DPO / Multi-Agent (Ours)	Small	**+100.0%**

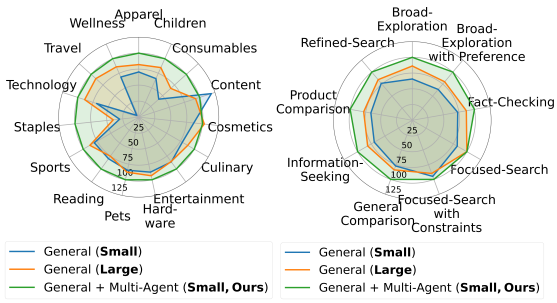
Table 2: Tool Calling Error Reduction Rates. Row **1-3**: small model (baseline). Row **4-6**: large model (baseline). Row **7-8**: small model (ours).

4.3.2 Tool Calling Performance

Table 2 reports tool-calling errors (e.g., invalid tool names or incompatible parameters). Among the baselines, DPO training reduces more errors for both small and large models, likely by strengthening the adherence to tool schemas through general instruction following preference pairs. Adding our domain-specific simulation data achieves 100.0% error reduction after DPO. This suggests that exposing the student model to diverse in-domain tool traces during training helps translate general instruction-following ability into reliable tool calls.

4.4 Model Capabilities by Product Domains and Shopping Stages

Figure-4 breaks down DPO model performances by product domains and journey stages. The Small baselines show uneven coverage, with sharp drops in several domains, while the large baselines are more balanced. Adding our simulation data largely closes this gap: our DPO model is the most bal-



(a) DPO by Domains (b) DPO by Stages

Figure 4: Model Capabilities Across (a): Product Domains; (b) Shopping Stages. Value: helpfulness relative to our DPO model.

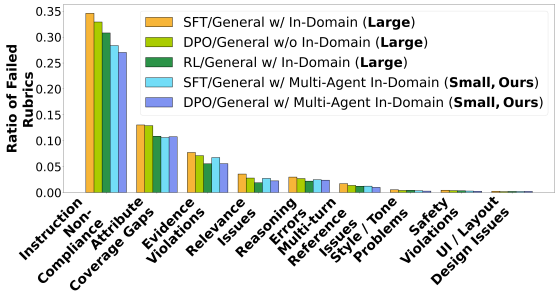


Figure 5: Model Failure Patterns. x-axis: Evaluation Rubric Categories. y-axis: Failed Rubric Percentage.

anced across domains and are competitive or best across stages, especially on later-stage behaviors (e.g., comparison/fact-checking). This aligns with the idea of our mission generator and user simulator modeling, which explicitly aims at domain and journey-stage coverage (see more in Appendix-G).

4.4.1 Failure Pattern Distribution

Figure 5 compares failure-pattern distributions for our small models against large-model baselines. Across models, the most prevalent categories are *Instruction Non-Compliance* (instruction violations or missing required content), *Coverage Gaps* (omission of key attributes), and *Evidence Violations* (unsupported claims). Our small models trained on simulation data consistently reduce failures across these categories, suggesting that the multi-agent iterative refinement process helps small models better optimize trade-offs among helpfulness, factuality, policy compliance, and other quality dimensions, resulting in more well-rounded responses.

4.4.2 Open-Weights Replication

To validate that our framework generalizes beyond proprietary model backbones, we replicate the core SFT/DPO pipeline on Qwen3-30B-A3B (Yang et al., 2025), an open-weight mixture-of-experts model with 3B active parameters, and evaluate with Claude Sonnet 4.5 as the judge (Table-3).

The direction of gains is consistent with our proprietary results (Table-1): multi-agent simulation data improves both helpfulness and tool-call reliability across training stages. The smaller margin compared to Table-1 (+10.0% vs. +24.4%) is expected, as Qwen-30B-A3B is a strong reasoning model whose higher base capability leaves less room for improvement. We include Claude Sonnet 4.5 zero-shot as an upper reference.

Stage / Domain Data	Helpfulness ↑	Tool-Call Err ↓
Baseline (No SFT)	0.714	0.0%
SFT / Manual	0.735	0.9%
SFT / Multi-Agent (Ours)	0.785	0.0%
DPO / Multi-Agent (Ours)	**0.809**	0.0%
Claude Sonnet 4.5	0.852	0.0%

Table 3: Open-weight replication on Qwen3-30B-A3B (3B active params). Our SFT/DPO data yields +6.8%/+10.0% helpfulness over Manual SFT and eliminates tool-calling errors. Claude Sonnet 4.5 zero-shot included as upper reference.

Conclusion

We presented a closed-loop multi-agent simulation framework for training shopping assistants. The system couples a journey-aware user simulator that models non-linear shopping progress, a shopping agent operating in a dual tool/UI action space, and a rubric-driven critic agent that iteratively refines trajectories through text gradients. The resulting data enables a small model to outperform same-size baselines and match or surpass a larger baseline in quality; deploying the small model then delivers 8× higher inference throughput than the larger baseline. Together, our work establishes a blueprint for aligning agents in open-ended domains where curating rigid ground truth is infeasible. We demonstrate that translating high-level design principles into qualitative textual feedback effectively balances autonomous reasoning with strict business compliance, without the rigidity of templates and the sparsity of scalar rewards.

Limitations

Our framework involves several limitations. First, our Critic agent uses LLM-based rubric scoring grounded in expert-curated Standard Operating Procedures (SOPs); this improves the verifiability in

business protocol-driven, high-stakes domains but may be less suitable for purely open-ended creative tasks and can introduce judge and model bias. Second, while the journey-aware simulator captures non-linear goal-oriented behaviors, it still assumes a degree of rationality and does not fully reflect the noise and spontaneity of real users. We scope the work of modeling the full chaos of human input, such as adversarial, irrational, or non-cooperative behavior as an important and independent direction for future work. Lastly, the iterative refinement loop adds additional offline compute (approximately +1.09 iterations of refinement per turn on average), which we treat as an investment to transfer complex reasoning into smaller models that achieve the higher inference throughput needed for cost-effective production deployment.

Ethical Considerations

Our framework relies on a fully synthetic data generation pipeline, which mitigates privacy risks by eliminating the need for real customer interaction logs or personally identifiable information. However, synthetic trajectories may still inherit statistical biases or stereotypes from the teacher models used for simulation. To reduce this risk, our Critic Agent incorporates safety and policy rubrics into the gating loop, together with verifiable checks, to filter harmful or biased content before it enters the training set; nevertheless, residual bias may remain and should be audited prior to deployment.

Next, although our method achieves near-zero tool-calling errors in our setting, real-world deployment should include deterministic guardrails, such as tool call pattern allow-lists/deny-lists, argument validation, rate limits, and so on, to reduce the risk of adversarially induced or otherwise unsafe tool invocations.

From an environmental perspective, the iterative refinement loop increases offline computational cost (averaging 1.09 retries per turn) relative to standard generation. We view this as a strategic investment as it enables substantially higher inference throughput and lower energy use during large-scale serving of smaller models.

References

Jafar Afzali, Aleksander Mark Drzewiecki, Krisztian Balog, and Shuo Zhang. 2023. Usersimcrs: A user simulation toolkit for evaluating conversational recommender systems. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 1160—1163.

Amazon. 2024. Amazon announces Rufus, a new generative AI-powered conversational shopping experience. https://www.aboutamazon.com/news/retail/amazon-rufus. Published: Feb. 1, 2024. Accessed: 2025-12-29.

Nolwenn Bernard and Krisztian Balog. 2023. Mgshopdial: A multi-goal conversational dataset for e-commerce. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2775—-2785.

Nicolas Bougie and Narimawa Watanabe. 2025. SimUSER: Simulating user behavior with large language models for recommender system evaluation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 6: Industry Track)*, pages 43–60.

James F. Engel, Roger D. Blackwell, and Paul W. Miniard. 1990. *Consumer Behavior*. Dryden Press series in marketing. Dryden Press.

Jiaxuan Gao, Jiaao Chen, Chuyi He, Shusheng Xu, Di Jin, and Yi Wu. 2026. From self-evolving synthetic data to verifiable-reward rl: Post-training multi-turn interactive tool-using agents. *Preprint*, arXiv:2601.22607.

Google. 2025. Let AI do the hard parts of your holiday shopping. https://blog.google/products/shopping/agentic-checkout-holiday-ai-shopping/. Published: Dec. 5, 2025. Accessed: 2025-12-29.

Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. CRITIC: Large language models can self-correct with tool-interactive critiquing. In *The Twelfth International Conference on Learning Representations*.

Yoonho Lee, Joseph Boen, and Chelsea Finn. 2025. Feedback descent: Open-ended text optimization via pairwise comparison. *Preprint*, arXiv:2511.07919.

Megan Leszczynski, Shu Zhang, Ravi Ganti, Krisztian Balog, Filip Radlinski, Fernando Pereira, and Arun Tejasvi Chaganty. 2023. Talk the walk: Synthetic data generation for conversational music recommendation. *Preprint*, arXiv:2301.11489.

Xiangci Li, Zhiyu Chen, Jason Ingyu Choi, Nikhita Vedula, Besnik Fetahu, Oleg Rokhlenko, and Shervin Malmasi. 2025a. Wizard of shopping: Target-oriented E-commerce dialogue generation with decision tree branching. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13095–13120.

Yafu Li, Xuyang Hu, Xiaoye Qu, Linjie Li, and Yu Cheng. 2025b. Test-time preference optimization: On-the-fly alignment via iterative textual feedback. In *Proceedings of the 42nd International Conference*

on Machine Learning, volume 267, pages 34630–34673.

Chen Luo, Dimitri Papadimitriou, Hariharan Muralidharan, Dhineshkumar Ramasubbu, Aakash Kolekar, Wenju Xu, Cong Xu, Anirudh Srinivasan, Mukesh Jain, and Qi He. 2025. Language model alignment for conversational shopping at amazon. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4314—-4318.

Xiang Luo, Zhiwen Tang, Jin Wang, and Xuejie Zhang. 2024. DuetSim: Building user simulator with dual large language models for task-oriented dialogues. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5414–5424.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: iterative refinement with self-feedback. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, volume 36, pages 46534–46594.

Shuhaib Mehri, Xiaocheng Yang, Takyoung Kim, Gokhan Tur, Shikib Mehri, and Dilek Hakkani-Tür. 2025. Goal alignment in LLM-based user simulators for conversational AI. In *First Workshop on Multi-Turn Interactions in Large Language Models (MTI-LLM) at NeurIPS 2025*.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. SimPO: Simple preference optimization with a reference-free reward. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Tarek Naous, Philippe Laban, Wei Xu, and Jennifer Neville. 2026. Flipping the dialogue: Training and evaluating user language models. In *The Fourteenth International Conference on Learning Representations*.

OpenAI. 2025. Introducing shopping research in ChatGPT. https://openai.com/index/chatgpt-shopping-research/. Published: Nov. 24, 2025. Accessed: 2025-12-29.

Perplexity. 2024. Shop like a pro: Perplexity's new AI-powered shopping assistant. https://www.perplexity.ai/hub/blog/shop-like-a-pro. Published: Nov. 18, 2024. Accessed: 2025-12-29.

Ivan Sekulic, Silvia Terragni, Victor Guimarães, Nghia Khau, Bruna Guedes, Modestas Filipavicius, Andre Ferreira Manso, and Roland Mathis. 2024. Reliable LLM-based user simulator for task-oriented dialogue systems. In *Proceedings of the 1st Workshop on Simulating Conversational Intelligence in Chat (SCI-CHAT 2024)*, pages 19–35.

Jeonghoon Shim, Woojung Song, Cheyon Jin, Seungwon KooK, and Yohan Jo. 2026. Non-collaborative user simulators for tool agents. In *The Fourteenth International Conference on Learning Representations*.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2023. Reflexion: language agents with verbal reinforcement learning. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, volume 36, pages 8634–8652.

Walmart. 2025. Walmart: The future of shopping is agentic. meet sparky. https://corporate.walmart.com/news/2025/06/06/walmart-the-future-of-shopping-is-agentic-meet-sparky. Published: Jun. 6, 2025. Accessed: 2025-12-29.

Kuang Wang, Xianfei Li, Shenghao Yang, Li Zhou, Feng Jiang, and Haizhou Li. 2025. Know you first and be you better: Modeling human-like user simulators via implicit profiles. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 21082–21107.

Liqiang Xiao, Jun Ma, Xin Luna Dong, Pascual Martínez-Gómez, Nasser Zalmout, Chenwei Zhang, Tong Zhao, Hao He, and Yaohui Jin. 2021. End-to-end conversational search for online shopping with utterance transfer. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3477–3486.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. 2024. Textgrad: Automatic "differentiation" via text. *Preprint*, arXiv:2406.07496.

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

Zijian Zhang, Shuchang Liu, Ziru Liu, Rui Zhong, Qingpeng Cai, Xiangyu Zhao, Chunxu Zhang, Qidong Liu, and Peng Jiang. 2025. LLM-powered user simulator for recommender system. In *Proceedings of the Thirty-Ninth AAAI Conference on Artificial Intelligence*, volume 39, pages 13339–13347.

Jie Zou, Mohammad Aliannejadi, Evangelos Kanoulas, Shuxi Han, Heli Ma, Zheng Wang, Yang Yang, and Heng Tao Shen. 2025. PSCon: Product search through conversations. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3659—-3669.

A User Simulator Evaluation

To assess the intrinsic utility of the simulator, we sampled 100 trajectories from each of two user simulator settings: (1) a naive prompt-based simulator and (2) our simulator with journey definition and state tracking. The 200 trajectories were blended and anonymized before annotation. Two annotators rated each user question with binary (0/1) labels along 7 pre-defined behavioral metrics (including role drift, over-cooperative revelation, fact-checking behavior, premature termination, backtracking, etc.). Scores were averaged per metric across the two settings. Inter-annotator agreement was 74%.

As in Figure 6, the results show that state tracking yields significantly higher utility, effectively eliminating role drift while boosting coverage of long-tail behaviors like fact-checking, side-by-side comparison, and iterative backtracking. In contrast, the baseline frequently drifts and misses complex interactions, confirming that structured state tracking is essential for capturing realistic user exploration and producing high-quality synthetic trajectories.

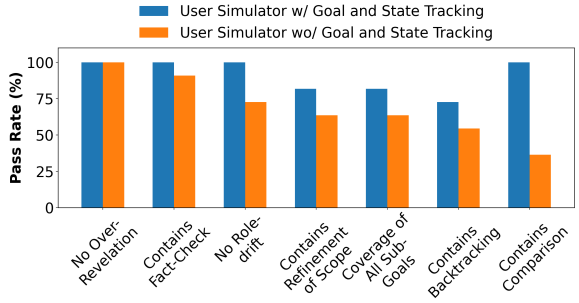


Figure 6: User Simulator Evaluation. x-axis: Evaluation Dimensions. y-axis: Human Judge Pass Rate.

B Evaluator Rubric Construction Process

The Critic Agent's rubric system comprises 10 groups organized into two tiers:

- **6 global groups**, applied at every turn: tool planning, overall relevance, logic/redundancy, multi-turn coherence, layout, and format.
- **4 feature-specific groups**, conditionally selected based on the interaction type: recommendation, general Q&A and so on.

Construction process. Rubrics are built through a four-stage human-in-the-loop pipeline:

1. **Establish SOPs.** Product managers curate Standard Operating Procedures grounded in user studies, encoding observed user behavior patterns and business requirements.
2. **Decompose SOPs into rubrics.** Each SOP is decomposed into binary (pass/fail) rubrics and assigned to the appropriate global or feature-specific group.
3. **Iterative calibration.** Domain experts run the multi-agent simulation on a 30-mission validation set (5-15 turns per mission), diagnose misalignments such as missing rubrics, misleading rubric descriptions, or phrasing that causes LLM misjudgments, revise the rubrics, and re-simulate until convergence.
4. **Cross-expert audit.** An independent expert (not involved in calibration) audits the final Critic judgments, yielding 94% human-LLM agreement.

C Training-Evaluation Separation

A natural concern with LLM-as-judge training loops is that the student model may overfit to the judge's preferences rather than genuinely improving. We mitigate this by design (Table-4): the training and evaluation pipelines differ on every axis that could enable such leakage.

Crucially, training rubrics are general quality checks (relevance, conciseness, tool selection) applied uniformly across all turns, while evaluation rubrics test question-specific factual properties grounded in product catalogs (e.g., verifying that a recommendation covers specific product attributes). A model that overfits to general training rubrics cannot pass evaluation rubrics that test entirely different factual properties. This separation is further reinforced by using judge models from different model families (DeepSeek for training, Claude for evaluation), preventing the student from learning stylistic biases of a single judge.

	Training	**Evaluation**
Dialogs	Synthesized by simulation	Curated by domain experts
Rubrics	General, all-turn	Question-specific, fact-grounded
Example	*"Is it relevant? Factual?"*	*"Must include 2+ fit types: bodycon, A-line, fitted, loose"*
Judge Model	DeepSeek	Claude
Purpose	Gating + feedback for refinement	Scoring accuracy

Table 4: Training vs. evaluation design comparison.

D Training Methods

Supervised Fine-Tuning (SFT). We first fine-tune the base model on the successful trajectories that are accepted by the critic agent (after iterative refinement). For each turn, the conditioning context x_i includes the system prompt, the tool and UI component schema, dialogue history, and tool results. The target y_i includes all assistant steps in the trajectory, including intermediate tool calls, layout plans, and the final accepted response. We minimize the standard negative log-likelihood over the policy π_θ:

$$\mathcal{L}_{\text{sft}}(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{t=1}^{|y_i|}\log \pi_\theta(y_{i,t} \mid x_i, y_{i,<t}). \tag{1}$$

To prevent catastrophic forgetting of general knowledge and capabilities, we mix this domain-specific data with general instruction-following examples.

Direct Preference Optimization (DPO). We further align the supervised-fine-tuned model using DPO. For a fixed context x, the multi-agent loop yields an accepted trajectory y^+ and multiple rejected trajectories y^-. We optimize the policy π_θ relative to the reference SFT model π_{ref} by pairing the y^+ and y^-:

$$\mathcal{L}_{\text{dpo}}(\theta) = -\,\mathbb{E}_{(x,y^+,y^-)\sim\mathcal{D}}[\log \sigma(\beta\,\Delta_\theta)]\,. \tag{2}$$

where Δ_θ represents the difference in implicit rewards: $\Delta_\theta = \log \frac{\pi_\theta(y^+|x)}{\pi_{\text{ref}}(y^+|x)} - \log \frac{\pi_\theta(y^-|x)}{\pi_{\text{ref}}(y^-|x)}$.

E Training Data Details

E.1 Baseline SFT Data

The manual in-domain SFT baseline contains 185 single-turn trajectories curated by annotators. Manual curation became infeasible as the product scope expanded, motivating our scalable simulation approach.

E.2 Multi-Agent Simulated SFT Data

We use frontier models (DeepSeek) for synthesizing the data. Table-5 presents descriptive statistics for the in-domain simulation training data. The dataset spans 329 product domains and 332 user personas. It incorporates 71 unique tool chains, with conversation lengths averaging 10.63 turns and ranging from 5 to 19 turns. The feedback loop averaged 1.09 iterations per turn, with a maximum of 31 iterations.

Statistic	Value
Domains	329
Personas	332
Unique tool chains	71
Turns (avg / min / max)	10.63 / 5 / 19
Refine iter. (avg / min / max)	1.09 / 0 / 31

Table 5: In-Domain Simulation Training Data Statistics.

F Experiment Setup

F.1 Proprietary Setting

- **Model backbones**. We use standard decoder-only Transformer backbones and report two capacity tiers: Tier-Small (lower capacity) and Tier-Large (higher capacity). Model identities, sizes, and weights are not disclosed for confidentiality.
- **Training protocol**. All experimental conditions follow a shared training protocol; the data recipe is the only controlled variable.
- **Reporting protocol**. We report relative changes with respect to matched baselines under the same protocol. Absolute metrics are omitted due to internal reporting constraints.
- **Evaluation**. We use offline, rubric-guided evaluation with identical prompts, judges, and scoring procedures across all conditions.

F.2 Open-Weight Setting

- **Model backbone.** We use Qwen3-30B-A3B[1], an open-weight mixture-of-experts model with 3B active parameters.
- **SFT.** We perform full-parameter fine-tuning on accepted simulation trajectories using DeepSpeed ZeRO-3. We train for 2 epochs with a learning rate of 1×10^{-5}, a cosine schedule with 10% warmup, an effective batch size of 256, and a maximum sequence length of 32768 tokens.
- **DPO.** We initialize from the SFT checkpoint and apply DPO with the SimPO loss variant (Meng et al., 2024) ($\beta = 1.0$, $\gamma = 0.5$) using DeepSpeed ZeRO-3. We train for 2 epochs with a learning rate of 3×10^{-6}, and a cosine schedule with 10% warmup. Preference pairs are constructed from missions where the Critic Agent rejected at least one iteration before final acceptance: the accepted trajectory serves as

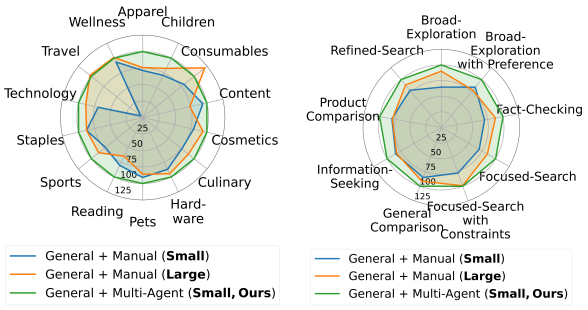
[1]Specifically `Qwen3-30B-A3B-Thinking-2507`.

y^+ and the rejected iteration as y^-.

- **Evaluation.** We evaluate on the same rubric-guided benchmark used in the proprietary setting (Table 1), with Claude Sonnet 4.5 as the judge. We report absolute helpfulness scores and tool-calling error rates rather than relative deltas.

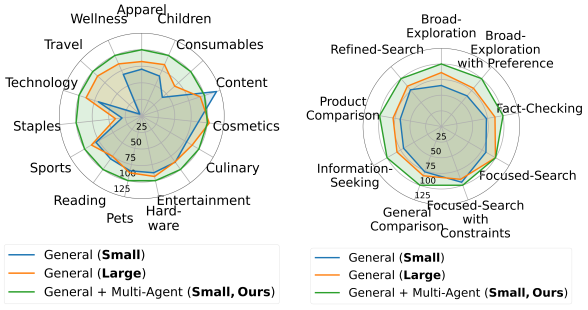
G Model Capabilities by Product Domains and Shopping Stages

Figure-7 visualizes the performance capability of the Small and Large models across diverse product domains and shopping journey stages, comparing both SFT and DPO training settings. The radar charts reveal that the small model baselines (blue) exhibit uneven coverage, with significant performance deficits in specific domains (e.g. Travel) and complex journey stages such as `Broad Exploration` and `Fact-Checking`.



(a) SFT by Domains

(b) SFT by Stages



(c) DPO by Domains

(d) DPO by Stages

Figure 7: **Model Capabilities Across Product Domains and Journey Stages.** (a) SFT by Domains; (b) SFT by Stages; (c) DPO by Domains; (d) DPO by Stages. Values: helpfulness relative to our DPO model.

In contrast, the small models trained with the proposed simulation data (green) demonstrate a substantially more balanced profile, effectively filling these coverage gaps to match or exceed the performance consistency of the Large-model baseline (orange). Notably, the simulation-based small model rectifies the degradation observed in the small model baselines, achieving balanced competency across the full spectrum of product domains and shopping intents.

H Failure Analysis

We present two remaining gaps of the simulation-based small model. The first is "instruction/rubric non-compliance," which slightly differs from general instruction-following. It is highly nuanced and detail-focused, reflecting cases where the model misses specific details according to the high bar set by question-specific evaluation rubrics.

Failure 1: Lack Categorization

```
[Context]
The user asked the shopping agent to
 recommend deodorant options and
compare their features.

[Rubric]
Response must mention the different
available forms of the spray product
 (e.g., stick, gel, spray)

[Judge Finding]
FAIL. The table provides detailed
information about Brand-A Spice
options including scent and price
but omits different product forms.
```

Failure 2: Unsatisfied Constraints

```
[Context]
The user asked for gift recommen-
dations.

[Rubric]
Every recommended product must have
either 'best seller' status or '
Buyer's Choice' designation

[Judge Finding]
FAIL. After examining product
information for all six recommended
products, there is no indication
that any have 'best seller' or '
Buyer's Choice' designation, despite
 the recommended products have high
rating (4.5 on average).
```

A second major failure mode is context window degradation: as conversations grow longer (our eval set averages 10.6 turns), the model increasingly struggles to attend to every requirement in the system prompt, leading to more omissions in later turns.

I Case Study of Simulated Trajectory

Disclaimer: All personas, missions, conversations, and evaluation examples in this work are generated synthetically.

I.1 Persona Example

Disclaimer: This persona is synthetic and not based on any real user request, customer data, or production logs.

Persona Example

```
<persona>
    <role>You're a customer of an
    E-commerce website. You will need the
    shopping assistant to help you to
    complete your shopping missions.</role>
    <age>34</age>
    <profession>Historic Preservation
    Specialist</profession>
    <expertise>Average</expertise>
    <preferences>Non-adhesive
    installations, removable grips,
    clear/neutral aesthetic materials,
    vintage-compatible solutions,
    eco-friendly materials,
    wobble-resistant adjustable products
    for uneven floors</preferences>
    <knowledge_blind_spots>Compatibility of
    modern safety products with delicate
    1950s finishes, chemical reactions
    between adhesives and aged wood,
    specialized non-damaging gate mounting
    systems, eco-certifications for
    baby-proofing materials, temporary
    installation techniques for rental
    restrictions</knowledge_blind_spots>
</persona>
```

I.2 Mission Example

Disclaimer: This mission is synthetic and not based on any real user request, customer data, or production logs.

Mission Example

I need to childproof my 1950s-era hardwood-floored home for my 10-month-old, who's starting to crawl. I'm looking for a baby-proofing kit that includes non-slip rug pads (to prevent tripping and floor scratches), soft corner guards for sharp furniture edges, and safety gates that won't damage vintage baseboards. The products must be non-adhesive or use removable grips to preserve the original wood finish. I also need advice on materials that blend with the home's mid-century aesthetic—clear or neutral tones preferred. Since the floors are uneven in some areas, the assistant should recommend adjustable, wobble-resistant solutions. Budget is flexible, but eco-friendly materials are a priority. Finally, installation tips for delicate surfaces would help—I'm nervous about drilling or chemical adhesives harming the floors.

I.3 User Simulator's Journey State Tracking Table

Disclaimer: This user state tracking table is synthetic and not based on any real user request, customer data, or production logs.

Table 6 shows a snippet of the internal state tracking mechanism of the User Simulator based on the shopping mission. The simulator first decomposes the high-level mission into distinct sub-goals (e.g., non-slip rug pads, soft corner guards, safety gates). For each sub-goal, the table captures the iterative decision-making process through the following columns:

- **Sub-Goal:** Track the sub-goals one-by-one by decomposing the given mission.
- **Candidates:** Tracks the "shortlist" of promising products selected from the assistant's recommendations. The sub-goal is considered complete when this set converges to a single finalized choice.
- **Phase:** Indicates the current stage of the shopping journey, progressing or backtracking through `Exploration`, `Comparison`, `Fact-Checking` or `Knowledge`.
- **Knowledge Blind Spots & Constraints Satisfaction:** Drive the agent's behavior by tracking what the user still needs to learn (blind spots) and which specific requirements (constraints) have been met by the current candidates.
- **User Action:** Specifies the next-step action (e.g., `chat`, `click`, `end`) to resolve remaining blind spots, verify unsatisfied constraints, or initiate the stage of next sub-goal.

Note that the orange-highlighted cells illustrate the change-of-mind moment, where the simulator finds certain constraints are not satisfied by current candidates during Fact-Check stage. The green-highlighted cells further capture the simulator's backtracking behavior, e.g. revisiting exploration stage after identifying the unmet constraint.

Table 6: User Simulator's Journey State Tracking Example. Note: Shopping agent responses following each user action have been omitted due to space constraints. For example responses with critic agent's feedback loop, see I.4.

User Simulator's Journey State Tracking Table (Example)					
Sub-Goal	Candidates	Phase	Knowledge Blind Spots	Constraints Satisfaction	User Action
Non-slip rug pads	[]	Know-ledge	- Material compatibility with vintage hardwood	- On 1950s hardwood floors - Childproofing	chat(text="What materials are safe to use on 1950s hardwood floors for childproofing?")
	[]	Explore	- Chemical reactions between modern materials and aged wood	- On 1950s hardwood floors - Childproofing	chat(text="Can you recommend non-slip rug pads safe for vintage hardwood?")
	[P-A1, P-A2, P-A3]	Explore	- Chemical reactions between materials and aged wood	- On 1950s hardwood floors ✓ - Childproofing ✓ - Natural material ✓ - Eco-friendly Material ✓	click(pid="P-A2")
	[P-A1, P-A2, P-A3]	Fact-Check	None	- On 1950s hardwood floors ✓ - Childproofing ✓ - Natural material ✓ - Eco-friendly Material ✓	chat(text="What color options are there for this 8x10 rug pad?")
	[P-A2]	Done	None	- On 1950s hardwood floors ✓ - Childproofing ✓ - Natural material ✓ - Eco-friendly Material ✓ - Color-Neutral ✓	Move onto next sub-goal
Soft corner guards	[]	Explore	Compatibility with vintage furniture finishes	- Non-adhesive options	chat(text="What corner guards can I use without damaging vintage furniture?")
	[P-B1, P-B2, P-B3]	Compare	Chemical reactions between metal and aged wood finishes	- Non-adhesive options	chat(text="Can you compare these three corner protectors?")

Soft corner guards	[P-B1, P-B2, P-B3]	Compare	None	- Non-adhesive options ✓ - Larger coverage ✓ - 6-point mounting system ✓	click(pid="P-B1")
	[P-B1]	Fact-Check	None	- Non-adhesive options ✓ - Larger coverage ✓ - 6-point mounting system ✓ - Material composition - Potential interaction with vintage wood	chat(text="What's the exact material of these corner guards and will they stain vintage wood?")
	[P-B1]	Done	None	- non-adhesive options ✓ - Larger coverage ✓ - 6-point mounting system ✓ - Material composition ✓ - Potential interaction with vintage wood ✓	Move onto next sub-goal
Safety gates	[]	Explore	Need to understand mounting options for vintage baseboards	- No vintage baseboard damage	chat(text="What safety gates won't damage vintage baseboards when installed?")
	[P-C1, P-C2]	Compare	Installation techniques for delicate surfaces	- No vintage baseboard damage - Retractable gate ✓	chat(text="Can you compare these two retractable gates for vintage homes?")
	[P-C1, P-C2]	Compare	Installation impact on vintage woodwork	- No vintage baseboard damage - Retractable gate ✓ - Wide coverage ✓	click(pid="P-C1")
	[P-C1]	Fact-Check	Installation impact on vintage woodwork	- No vintage baseboard damage - Retractable gate ✓ - Wide coverage ✓	chat(text="How does the mounting system work with baseboards?")

Safety gates	[P-C1]	Fact-Check	Installation impact on vintage woodwork	- No vintage baseboard damage - Retractable gate ✓ - Wide coverage ✓ - Uneven floors	chat(text="Does this gate adjust for uneven floors?")
	[P-C1, P-C2]	Explore	Installation impact on vintage woodwork	- No vintage baseboard damage - Retractable gate ✓ - Wide coverage ✓ - Uneven floors ✗	chat(text="Are there any safety gates specifically designed for uneven floors?")
	[P-C1, P-C3, P-C4]	Compare	None	- No vintage baseboard damage - Retractable gate ✓ - Wide coverage ✓ - Uneven floors ✓	chat(text="Can you compare the P-C3 and P-C4 gates?")
	[P-C3, P-C4]	Compare	None	- No vintage baseboard damage - Retractable gate ✓ - Wide coverage ✓ - Uneven floors ✓	click(pid="P-C4")
	[P-C4]	Fact-Check	None	- No vintage baseboard damage - Retractable gate ✓ - Wide coverage ✓ - Uneven floors ✓	chat(text="What mounting hardware comes with this gate and how does it install?")
	[P-C4]	Done	None	- No vintage baseboard damage ✓ - Retractable gate ✓ - Wide coverage ✓ - Uneven floors ✓	Sub-goal completed.
	[P-A2], [P-B1], [P-C4]	Done	None	All satisified	end(conclusion="I have successfully identified child-proofing solutions safe for my 1950s home: P-A2 rug pad for non-slip protection, P-B1 Corner Guards for furniture protection, and P-C4 Safe Gates for doorways/stairs. ")

I.4 Generate->Critique->Refine Loop

Disclaimer: The product described below is entirely fictional. All names, specifications, and features are invented for illustrative purposes and do not correspond to any real-world brand or item.

I.4.1 Example-1

Summary: The User Simulator requested a security-feature comparison between fireproof document organizers A and B. The initial Shopping Agent response was overly long and hard to read due to deeply nested bullet lists. After Critic Agent's feedback, the Shopping Agent reformatted the answer into a side-by-side comparison table with concise recommendations that highlighted key decision factors, which passed all the checks by the Critic Agent.

User Simulator

How do fireproof document organizer [Product_A]'s security features compare with [Product_B]?

Shopping Agent (V1)

Security Features Comparison: [Product_A] vs [Product_B]

Fire Resistance

- **[Product_A]**:
  - Rated for heat exposure up to $\sim 5000^{\circ}\mathrm{F}$
  - Built with silicone-based fabric and a foil inner layer
  - Foil layer adds an extra thermal shield
- **[Product_B]**:
  - Meets a recognized flame-resistance standard (**[Cert_Fire_1]**)
  - Uses about **1.2 mm** fire-retardant material
  - Silicone-coated fiberglass shell
  - Verified by an independent lab (**[Cert_Test_2]**)

Locking Design

- **[Product_A]**:
  - Dual-zip closure
  - No passcode lock included
  - Water-sealed zipper track
- **[Product_B]**:
  - **4-digit code lock** (keyless)
  - Dual metal zipper hardware
  - Anti-tamper construction
  - Reinforced stitching

Moisture Protection

- **[Product_A]**:
  - **Fully waterproof construction**
  - Sealed zipper system designed for strong spray exposure
  - End-to-end moisture barrier design
- **[Product_B]**:
  - Water-resistant build
  - Guarded seams
  - Not specified as fully waterproof

Material & Build Quality

- **[Product_A]**:
  - Silicone-fiberglass composite
  - Foil-lined interior
  - Reinforced body structure
- **[Product_B]**:
  - Thickened silicone-coated shell
  - Low-irritation fiberglass weave
  - Dust- and stain-resistant exterior finish

Recommendation

Choose **[Product_A]** if:

- Higher heat tolerance is your top concern
- You want fully waterproof coverage
- You prefer zipper-only access

Choose **[Product_B]** if:

- You want passcode-based access control
- You prefer certification-focused fire claims
- Tamper resistance matters most

Tip: Both provide strong protection overall. **[Product_B]** adds an access-control layer via code lock, while **[Product_A]** emphasizes stronger fire/water performance claims.

Critic Agent (V1)

Step-by-Step Instructions to Improve the Response

Tool-Invocation Plan

1. Call the same tools as last iteration to fetch refreshed product details.
2. Do not skip tool invocation or directly reuse previous outputs.

Layout Plan

1. Use `layout=["text-block", "side-by-side-table", "text-block"]` structure.
2. The first text section should briefly introduce what will be compared.
3. The middle section should use a side-by-side table to present feature differences.
4. The final text section should provide recommendations and quick tips.

Response Refinement

1. Create an intro text section explaining the comparison scope.
2. Structure the table with columns:
   - Features
   - Product_A
   - Product_B
3. Include rows for:
   - Fire Resistance
   - Locking Mechanisms
   - Water Protection
   - Material & Build
4. Move recommendations to the final text section after the table.

Other Important Gaps

1. Make feature differences easier to scan with side-by-side presentation.
2. Reduce text-heavy bullet content by moving core comparisons into table format.
3. Keep only high-level recommendations in the closing text section.

Shopping Agent (V2)

Security Controls Comparison: [Product_A] vs [Product_B]

When selecting a document organizer, protection features are critical for safeguarding important records. Here's a side-by-side comparison of how these two options address different security needs.

Security Controls	Product_A	Product_B
Heat/Fire Resistance	Up to 5000°**F** with foil-lined interior	**[Fire_Std_X]** certified, ∼**1.2 mm** fire-retardant layer
Water Resistance	Fully waterproof build with sealed zippers	Water-resistant build (not fully waterproof-rated)
Locking Method	Dual-zip closure	**4-digit** code lock + dual metal zipper
Material Build	Silicone-fiber shell with foil inner lining	Silicone-coated fiberglass shell
Third-Party Validation	Not explicitly listed	**[Lab_Cert_Y]** verified
Physical Tamper Protection	Standard zipper-based closure	Tamper-aware build with reinforced seams

Recommendations by Security Priority

Choose **[Product_A]** if you prioritize:

- Higher heat tolerance (5000°F class)
- Full waterproof protection
- Lightweight zipper-only access

Choose **[Product_B]** if you prioritize:

- Standards-based fire-resistance claims
- Passcode lock security
- Tamper-resistant construction details

Tip: Both options provide solid document protection. If your top concern is **fire/water exposure**, lean toward **[Product_A]**. If your top concern is **access control**, **[Product_B]** is the better fit due to its code-lock setup.

Critic Agent (V2)

All rubrics and rule-checks passed.

I.4.2 Example-2

The User Simulator asked a Fact-Check question about the battery duration of Smart Glasses A. The initial Shopping Agent response was over-scoped, adding competitor comparisons and unnecessary details beyond the query. Across two revisions guided by Critic-Agent's feedback, the agent removed competitor product content, tightened the layout, and prioritized core battery information plus a brief usage-variation note; the final V3 response was concise, on-task, and passed Critic Agent's Checks.

User Simulator

How long does the Smart Glasses Product A battery last?

Shopping Agent (V1)

[Smart_Glasses_Product_A] Battery Performance

The **[Smart_Glasses_Product_A]** delivers **up to ∼4.5 hours of continuous use** on a single charge. Key battery details:

- **Single-Charge Runtime**: ∼3.5–4.5 hours during active use (clip capture, calls, audio playback)
- **Recharge Time**: about 55–70 minutes for a full charge
- **Charging Case Reserve**: up to ∼30 additional hours
- **Total Combined Runtime**: up to ∼34 hours with case + glasses

Important Battery Factors

Actual battery life can vary based on:

- Usage intensity (video capture typically drains faster)
- Ambient conditions (low temperatures can reduce battery efficiency)
- Playback loudness
- Frequency of assistant/voice-command usage

Here are similar products for consideration:

- `Official Smart_Glasses_Product_A` with up to ∼4.5 hours battery life, on-device assistant features, live language support, and high-resolution camera + multi-mic array.
- `Smart_Glasses_Product_B` offering extended endurance of roughly 28–48 hours total daily usage and about 9 hours of continuous music playback.
- `Smart_Glasses_Product_C` with approximately 5.5 hours continuous runtime, quick ∼65-minute charging, and splash-resistant protection.

Battery Optimization Tips

To maximize **[Smart_Glasses_Product_A]** battery life:

- **Keep the charging case powered** — it can supply roughly **7x** extra top-ups
- **Use quick top-up charging** when short on time
- **Track battery status** in the **[Companion_App_A]**
- **Account for temperature and usage patterns** that may impact runtime

If battery duration is your main priority, the alternatives above may offer longer runtime, though they may differ in camera/assistant capabilities compared with **[Smart_Glasses_Product_A]**.

Critic Agent (V1)

Step-by-Step Instructions to Improve the Answer

Tool-Execution Plan

1. Use same tools as previous iteration to retrieve product information and battery details.
2. Keep retrieval focused only on `[Product_A]`, and avoid pulling in competing smart-eyewear models.

Layout Plan

1. Use a simpler `["text-block", "text-block"]` layout.
2. Remove sections that compare alternative products.

Response Refinement

1. Keep only factual battery information for [Product_A] from official sources.
2. Remove all alternative product mentions (e.g., [Product_B], [Product_C]).
3. Focus battery optimization tips specifically on [Product_A].
4. Remove comparative claims about other products' battery performance.

Other Important Gaps

1. Ensure every battery claim is supported by official product documentation.
2. Remove speculative cross-brand performance statements.
3. Keep the response tightly focused on the original question: [Product_A] battery duration.

Shopping Agent (V2)

Smart_Glasses_Product_A Battery Endurance

Smart_Glasses_Product_A provides the following power profile:

Core Power Specs

- **Single-Charge Runtime**: Up to ∼**4.5 hours** of continuous operation
- **Case Reserve Capacity**: Up to ∼**30 extra hours**
- **Combined Runtime**: Up to ∼**34.5 hours** with a fully charged case

Charging Details

- **Full Recharge Time**: About **55–70 minutes** from empty to full
- **Charge Interface**: Magnetic contact points on both sides
- **Charge Indicator**: **Amber** while charging, **blue** when fully charged

What Impacts Battery Runtime

Actual battery performance may change based on:

- **Feature Intensity**: Clip capture, live broadcasting, and assistant functions draw more power
- **Ambient Conditions**: Lower temperatures can reduce battery efficiency
- **Audio Output Level**: Louder playback drains battery faster
- **Connection Activity**: Continuous wireless sync and active pairing increase power usage

Battery Optimization Tips

To extend battery life for **[Smart_Glasses_Product_A]**:

- Keep the charging case powered for longer day-to-day use
- Check battery status in **[Companion_App_A]**
- Place glasses back in the case when idle
- Tune volume and feature usage based on need
- Confirm both charging contacts are seated correctly

Note: Battery values reflect manufacturer-stated specifications and can vary by usage behavior, settings, and environment.

Critic Agent (V2)

Step-by-Step Instructions to Improve the Response

Tool-Execution Plan

1. Run the same tool sequence as previous iteration to fetch refreshed battery specifications for [Smart_Glasses_Product_A].
2. Reuse the same search-query intent to preserve consistency, while confirming the returned data is current.

Layout Plan

1. Keep the `["text-block"]` layout, since this is a direct product-spec question.
2. Organize into **2–3 main sections** (instead of 4–5) for better concision.

Response Refinement

1. Combine **"Core Battery Features"** and **"Charging Details"** into one **"Battery Specifications"** section.
2. Reduce battery optimization guidance to the **3 highest-impact tips**.
3. Fold **battery-affecting factors** into a short note inside the specifications section.
4. Keep only the details that directly answer battery-duration expectations.

Other Important Gaps

1. Remove overlap across sections to improve clarity.
2. Prioritize runtime values over low-priority charging-process detail.

Shopping Agent (V3)

[Smart_Glasses_Product_A] Battery Life

[Smart_Glasses_Product_A] offers:

- **Single-Charge Runtime**: ∼3.5–4.5 hours of continuous active use
- **Total Runtime (with included case)**: Up to ∼34–36 hours
- **Idle/Standby Duration**: ∼13–16 hours when features are not actively used
- **Full Recharge Time**: About 55–70 minutes

Note: Actual battery performance may vary based on usage intensity, enabled features (for example, video capture or assistant functions), and ambient conditions such as temperature.

Critic Agent (V3)

All rubrics and rule-checks passed.