SIMREC: LEARNING TO SEARCH, ASK, AND RECOMMEND FROM SIMULATED INTERACTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Recommendation agents built on large language models can search a catalog, inspect items, ask the user, and revise a recommendation after the user rejects it. In practice, however, these behaviors are scripted by hand-designed workflows rather than learned, and existing training methods optimize isolated components such as query rewriting or list ranking. As a result, agents rarely verify candidates, rarely use the user's history, and almost never recover from a rejected recommendation. We present **SimRec**, a reinforcement learning framework that trains the complete interaction policy of a tool-integrated recommendation agent. SimRec introduces two components. A task-anchored user simulator provides feedback on every recommendation: conditioned on the current task need rather than on the target item or a user profile, it states what the recommended item lacks without revealing the target, and it can be an external LLM or the recommender itself under self-play. A task-aligned reward optimizes the whole trajectory with GRPO, crediting natural retrieval quality and valid tool use alongside the final hit, with training-only target injection to handle sparse hits over large catalogs. On ESCI and Amazon C4, SimRec improves the NDCG@100 of a Qwen3-4B agent from **32.23% to 35.89%** and from **22.94% to 25.39%**, outperforming training-free agents on far larger backbones and task-aligned baselines. Analysis shows that the trained agent verifies candidates, uses preferences, and recovers from rejected recommendations far more often than its untrained counterpart. Our code is available at `https://anonymous.4open.science/r/SimRec-6F0D/`.

1 INTRODUCTION

Recommender systems are moving from one-shot prediction toward interaction. Built on a large language model (LLM), a recommendation agent takes a request in natural language, retrieves candidates from the catalog, inspects those that look plausible, asks the user when the need is underspecified, and recommends an item; if the user is not satisfied, it continues from the user's reaction (Huang et al., 2025b; Zhao et al., 2024; Zhang et al., 2026a; Li et al., 2026; Lin et al., 2026; Wang et al., 2026). Recommendation quality in this setting is not the output of a single scoring model but the result of a sequence of decisions made over the course of the interaction.

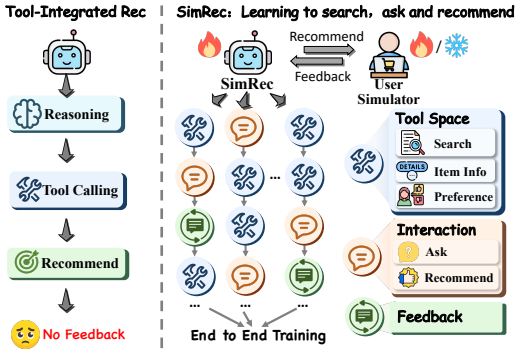


Figure 1: Compared to existing tool-integrated recommender agents (left), SimRec (right) learns to search, ask and recommend from simulated interaction.

Existing agents make these decisions by rules set in advance rather than by learning from outcomes: a hand-designed workflow prescribes when to search, when to ask, and when to recommend (Huang et al., 2025b; Fang et al., 2024; Wang et al., 2025b). Where training is used at all, it targets one step lifted out of the interaction, such as rewriting a query for the retriever (Lin et al., 2025; Yoon et al., 2025; Nguyen et al., 2025), ranking a given list (Zhu et al., 2026; Feng et al., 2025b), or generating a response to a given di-

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.

alogue (Xiang et al., 2026; Cai et al., 2026); the outcome of the whole interaction never reaches the policy that made the decisions along the way. These limitations show in behavior. Such agents seldom verify the candidates they retrieve, seldom consult the user's history, and rarely recover once a recommendation is rejected, and a larger backbone does not make up for it (Section 4.4). An interaction policy has to be trained on the interaction itself.

Reinforcement learning over complete trajectories trains a policy on exactly this: the agent acts, the environment responds, and the whole trajectory is optimized for the final outcome. For search agents the recipe is well established, with the model interleaving reasoning and search calls and the trajectory rewarded by the correctness of the final answer (Jin et al., 2025; Qian et al., 2025; Feng et al., 2025a). The search setting, however, comes with its environment given: the index answers every query the same way, search is the only tool, and the answer is either right or wrong (Figure 1). Recommendation has none of these. The environment is a user who reacts to each recommended item, the evidence comes from several tools with different costs, and the outcome is a hit that is rare and does not say where the failure occurred. Training a recommendation agent on the interaction itself therefore requires three things to be supplied: the user, the actions, and the objective.

Interaction with real users cannot supply training signal at the scale reinforcement learning needs, so the user is played by a simulator (Xiang et al., 2026). A simulator that knows the target item leaks it into the dialogue (Zhu et al., 2024; Kim et al., 2025), and one that reconstructs a full user profile may drift from the actual need over turns (Zhang et al., 2024; Meshi et al., 2026); the simulator should say only what a real user would say after a recommendation, namely whether the item fits the current need and what it lacks (Bi et al., 2019). The actions have to cover the evidence a recommendation depends on, that is, the candidates, the details of a single item, and the user's long-term preferences; each has a cost and none is always needed, so when to draw on each is left to the policy (Li et al., 2025a; Ma et al., 2024). The objective has to credit what the hit cannot see. A hit is rare, and a miss does not say whether the target was never retrieved or was retrieved and not chosen; retrieval quality and tool use therefore have to be rewarded alongside the final hit, and the reward has to remain informative when the hit is absent (Wang et al., 2026).

To meet these requirements, we propose SimRec, a framework that trains a recommendation agent end-to-end on multi-turn interaction with a simulated user (Figure 2). The user is a task-anchored simulator: it is conditioned on the current task need rather than on the target item or a profile, answers each recommendation with what the item lacks, and never names the target; it can be an external LLM or the recommender itself under shared-parameter self-play. The actions are retrieval, item inspection, preference access, a clarifying question, and a recommendation, in no fixed order; long-term preferences are thus evidence the agent fetches, not part of the simulator. The objective is a task-aligned reward over the whole trajectory that credits natural retrieval quality and valid tool use alongside the final hit, with training-only target injection so that candidate comparison can still be learned when retrieval misses. Trained with GRPO under this objective, the agent learns when to search, when to inspect, when to ask, and when to recommend.

On ESCI and Amazon C4, SimRec raises the NDCG@100 of its Qwen3-4B backbone from 32.23% to 35.89% and from 22.94% to 25.39%, above training-free agents on far larger backbones and above task-aligned baselines. The gain is in the interaction: the trained agent verifies candidates and consults preferences where the untrained one almost never does, recovers from a rejected recommendation more than twice as often, and ranks a fixed candidate set better in a controlled reranking test. The gains carry over to a sparse retriever and a smaller backbone, and training leaves general capabilities intact.

Our contributions are summarized as follows:

- We treat the interaction decisions of a recommendation agent, namely when to search, inspect, ask, and recommend, as a policy to be trained on the outcome of the whole interaction, and identify what such training requires: the user, the actions, and the objective.
- We propose SimRec, which addresses these requirements with a task-anchored user simulator, a tool-integrated action space over retrieval, item inspection, and preferences, and a task-aligned trajectory reward with training-only target injection.
- SimRec outperforms retrieval, training-free, and task-aligned baselines on ESCI and Amazon C4, and its gains trace to learned interaction behavior: candidate verification, use of preferences, recovery after a rejection, and stronger candidate comparison.

2 Related Work

2.1 Agentic Recommendation Agents

Conversational recommender systems use natural-language interaction to elicit user needs and support preference understanding and recommendation Christakopoulou et al. (2016); Jannach et al. (2022). LLM-based methods further improve intent understanding, candidate analysis, response generation, and multi-turn planning Huang et al. (2025a); Zhang et al. (2026b); Li et al. (2025a); Du et al. (2025), while conversational product search studies negative feedback, proactive questioning, query reformulation, and preference clarification Bi et al. (2019); Ma et al. (2024); Ye et al. (2025); Zou et al. (2025); Li et al. (2025b). Recent recommender agents formulate recommendation as sequential observation, reasoning, and action, often augmented with retrieval, user profiles, and external knowledge Huang et al. (2025b); Zhang et al. (2026a); Shang et al. (2026); Shi et al. (2025); Hui et al. (2026). Related work also explores proactive retrieval, query reformulation, and task-specific post-training for recommendation and tool decisions Zhao et al. (2024); Lin et al. (2026; 2025); Zhu et al. (2026); Li et al. (2026). However, existing multi-turn interaction is still largely used for need elicitation and information acquisition before recommendation. SimRec instead models a complete recommendation loop, where user responses and external evidence continuously update subsequent recommendation decisions.

2.2 User Simulation for Recommendation

User simulation has long supported conversational recommender systems (CRSs). Early work developed preference-aware and agenda-based simulators, while UserSimCRS incorporated persona, context, satisfaction, and conditional response generation Zhang & Balog (2020); Afzali et al. (2023). Subsequent studies systematize user simulation across information-access tasks and distinguish its training and evaluation roles Balog & Zhai (2023); Bernard & Balog (2024).

Recent work increasingly uses LLMs to simulate user profiles, preferences, and conversational behavior Ni et al. (2026); Wang et al. (2025a); Shi et al. (2025); Bougie & Watanabe (2025); Zhang et al. (2025b), together with dedicated evaluation protocols and benchmarks Yoon et al. (2024); Kim et al. (2025); Meshi et al. (2026). Simulated interactions have also been used to optimize multi-turn recommendation policies through preference optimization, self-distillation, and ranking-based objectives Xiang et al. (2026); Wang et al. (2026); Feng et al. (2025b); Zhu et al. (2026). In contrast, SimRec anchors simulation to the current task need and generates feedback on recommendation mismatch, while exposing long-term preferences separately through a tool.

3 Method

3.1 Problem Formulation

SimRec models recommendation as a finite-horizon interaction among a recommendation agent, external tools, and a user simulator (Figure 2). Let $\mathcal{I}$ be the product catalog, whose items have identifiers, textual descriptions, and structured attributes. A task is defined as

$$x = (u_0, h_x, z_x, \mathcal{Y}_x), \tag{1}$$

where u_0 is the initial request, h_x the interaction history, z_x the current task need, and $\mathcal{Y}_x \subseteq \mathcal{I}$ the set of target items. The simulator is conditioned on z_x, and the environment uses $\mathcal{Y}_x$ to assess success; neither is directly exposed to the agent. The agent can access long-term preferences summarized from h_x through the preference tool. When training requires one target, $y_x \in \mathcal{Y}_x$ denotes the selected item; for single-target tasks, $|\mathcal{Y}_x| = 1$.

At step t, the state s_t contains the initial request and accumulated dialogue, including user responses and tool observations. The agent selects $a_t \sim \pi_\theta(\cdot \mid s_t)$ from clarification, tool-use, and recommendation actions. Tools support product retrieval, item inspection, and preference access. The interaction follows

$$s_t \to a_t \to o_t \to s_{t+1}, \tag{2}$$

where o_t is the resulting user response or tool observation. For a recommendation i_t, the environment checks whether $i_t \in \mathcal{Y}_x$, and the simulator generates the corresponding response. A rejection

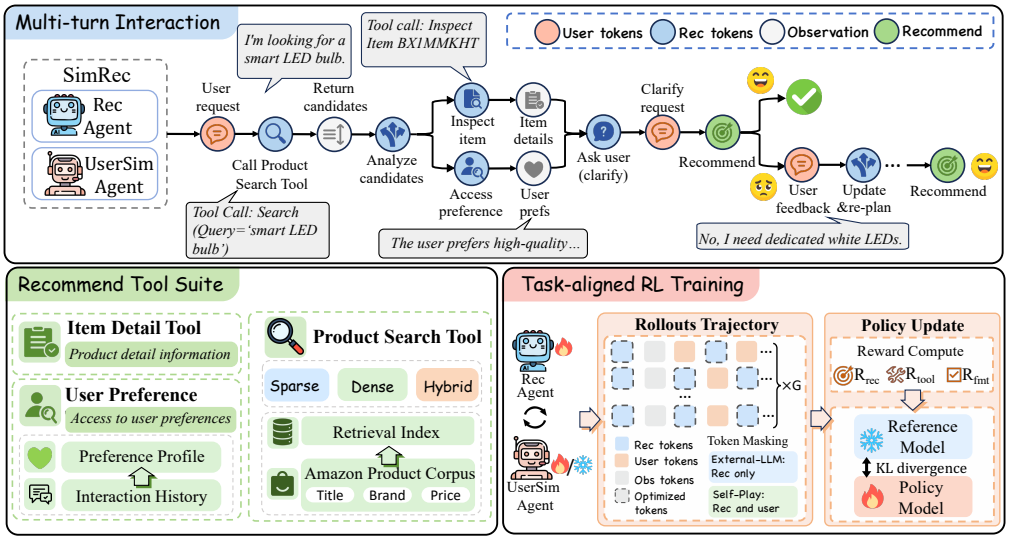


Figure 2: Overview of SimRec. The recommendation agent interacts with a user simulator and external recommendation tools, while task-aligned GRPO optimizes the policy over complete multi-turn trajectories.

continues the interaction. The episode ends upon success, after N_{rec} recommendation attempts, or at horizon $T_{\max}$, yielding a trajectory $\tau = (s_1, a_1, o_1, \ldots, s_T, a_T, o_T)$.

3.2 Tool-Augmented Recommendation

SimRec provides three external tools through a unified interface, covering the three key information needs in recommendation: **candidate discovery**, **candidate verification**, and **user preference**. Tool outputs are incorporated into the interaction state as observations.

Product Search. Given a natural-language query, the retriever returns a relevance-ranked product list. SimRec provides interchangeable retrieval backends, allowing dense or sparse retrievers to be selected for different tasks. To limit context length, search results expose compact product snippets, while full product information is accessed through the Item Detail tool.

Item Detail. The Item Detail tool returns full product information for a selected candidate, providing grounded evidence for verifying constraints such as price, size, compatibility, and functionality and for distinguishing otherwise similar candidates.

User Preference. For tasks with user history, h_x is standardized, deduplicated, chronologically ordered with target items removed, and LLM-summarized into long-term preferences. The agent accesses this summary on demand. Preference construction details are provided in the appendix C.4.

Tool use follows no fixed pipeline. At each step, the agent may retrieve, inspect items, access preferences, clarify the request, or recommend an item, making tool use part of the learned recommendation policy.

3.3 Task-Anchored User Simulation

SimRec conditions user simulation on the current task need z_x rather than a reconstructed long-term user profile. The simulator generates a response to the agent's action a_t as

$$u_t \sim \pi_{\text{sim}}(\cdot \mid z_x, a_t, \mathcal{H}_t), \tag{3}$$

where $\mathcal{H}_t$ is the interaction history. For a recommendation, a_t includes the proposed item i_t; for a clarification turn, it contains the agent's question. The simulator receives no hidden target ID and generates feedback grounded in the task need and interaction history. The environment separately determines recommendation success by checking whether $i_t \in \mathcal{Y}_x$. SimRec supports both a fixed external LLM as the simulator and shared-parameter self-play with role-specific prompts.

3.4 TOOL-INTEGRATED POLICY OPTIMIZATION

3.4.1 MULTI-TURN GRPO

We apply Group Relative Policy Optimization (GRPO) Shao et al. (2024) over complete multi-turn trajectories. For each task x, the old policy samples a group $\mathcal{G}(x) = \{\tau_1, \ldots, \tau_G\}$, where each trajectory receives reward $R_i = R(\tau_i)$. Its group-normalized advantage is $\hat{A}_i = (R_i - \mu_x)/(\sigma_x + \epsilon)$, where μ_x and σ_x are the mean and standard deviation of rewards within the group. For token $y_{i,k}$ generated under context $c_{i,k}$, we define the policy ratio as $\rho_{i,k}(\theta) = \pi_\theta(y_{i,k} \mid c_{i,k})/\pi_{\theta_{\text{old}}}(y_{i,k} \mid c_{i,k})$. The clipped objective and final optimization objective are

$$\ell_{i,k} = \min\Big[\rho_{i,k}\hat{A}_i,\ \text{clip}(\rho_{i,k}, 1-\epsilon_c, 1+\epsilon_c)\hat{A}_i\Big]. \tag{4}$$

$$J(\theta) = \mathbb{E}\left[\frac{1}{|\mathcal{M}|}\sum_{(i,k)\in\mathcal{M}}\big(\ell_{i,k} - \beta D_{\text{KL}}\big(\pi_\theta(\cdot \mid c_{i,k}) \| \pi_{\text{ref}}(\cdot \mid c_{i,k})\big)\big)\right], \tag{5}$$

where $\mathcal{M}$ denotes the optimized tokens, and all tokens within the same trajectory share the trajectory-level advantage $\hat{A}_i$. External-LLM simulation updates only recommender tokens, while shared-parameter self-play jointly optimizes both roles for recommendation success.

3.4.2 TASK-ALIGNED REWARD

The trajectory reward combines recommendation quality, natural retrieval, and effective tool use, with a unified penalty for redundant and overly long interactions:

$$R(\tau) = \text{clip}(\lambda_{\text{rec}}R_{\text{rec}} + \lambda_{\text{tool}}R_{\text{tool}} + R_{\text{fmt}} - P(\tau), R_{\min}, R_{\max}). \tag{6}$$

Recommendation Reward. We define $R_{\text{rec}} = R_{\text{hit}} + R_{\text{ret}}$, where $R_{\text{hit}} = \mathbb{I}[i_{\text{final}} \in \mathcal{Y}_x]$ and is zero if no final recommendation is produced. To provide a denser retrieval signal, let $\mathcal{Q}_{\text{nr}}(\tau)$ denote non-redundant searches, i.e., previously unseen normalized queries whose Top-K results contain at least one new item. We define

$$R_{\text{ret}}(\tau) = \max_{q_j \in \mathcal{Q}_{\text{nr}}(\tau)} \text{NDCG}\,@K(L_j, \mathcal{Y}_x), \tag{7}$$

and set $R_{\text{ret}} = 0$ if no non-redundant search occurs. We use standard binary-relevance NDCG@K, i.e., $\text{NDCG@}K = \text{DCG@}K/\text{IDCG@}K$. For single-target tasks, it reduces to the discounted gain determined by the target rank. Retrieval rewards are computed from natural, uninjected results.

Tool-Use Reward. We reward valid use of each tool category:

$$R_{\text{tool}} = \lambda_{\text{search}}R_{\text{search}} + \lambda_{\text{inspect}}R_{\text{inspect}} + \lambda_{\text{pref}}R_{\text{pref}}. \tag{8}$$

where $R_a = \mathbb{I}[N_a^{\text{valid}}(\tau) > 0]$ for $a \in \text{search}, \text{inspect}, \text{pref}$. Invalid, failed, or redundant calls do not count, so each tool type contributes at most once per trajectory.

Format Reward. R_{fmt} rewards each recommendation containing a parseable item ID and a nonempty description, regardless of target match.

Interaction Penalty. We combine query repetition and excessive interaction length into

$$P(\tau) = \lambda_{\text{repeat}}\mathbb{I}[\text{norm}(q^+) = \text{norm}(q^-)] + \min(P_{\max}, \lambda_{\text{len}}[N_{\text{turn}}(\tau) - T_{\text{free}}]_+), \tag{9}$$

where q^- and q^+ are the highest-scoring natural queries before and after the first corrective feedback; the repetition term is zero if either is unavailable, and $[a]_+ = \max(a, 0)$.

3.4.3 DYNAMIC RETRIEVAL MECHANISM

For query q_t, the retriever returns $L_t = \text{Retrieve}_K(q_t)$, but the agent sees only $V_t = L_t[:k]$, where $k \le K$. Reformulating queries across turns exposes different candidates and expands trajectory-level recall coverage. To address the sparse training signal over large catalogs, when a target remains outside the visible list we inject a selected $y_x \in \mathcal{Y}_x$ after at least N user turns:

$$n_{\text{user}}(t) \ge N, \quad y_x \notin V_t \quad \Longrightarrow \quad \widetilde{V}_t = \text{RandInsert}(V_t, y_x). \tag{10}$$

Here, $\mathrm{RandInsert}$ inserts the target at a uniformly random position in the visible candidate list, avoiding a fixed positional cue. It preserves learning opportunities for candidate comparison when retrieval misses the target, following controlled candidate-construction strategies in agentic recommendation Wang et al. (2026). Retrieval rewards are always computed from uninjected results, and target injection is disabled during evaluation to prevent information leakage.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets. We evaluate SimRec on ESCI Reddy et al. (2022), which provides query–product relevance annotations, and Amazon C4 Hou et al. (2026), which contains complex shopping requests. We use five ESCI categories and four Amazon C4 categories. For Amazon C4, we collect user histories from the original Amazon Reviews 2023 data and make them available through the preference tool.

Evaluation Metrics. We report instance-weighted NDCG@100 and Recall@100 for retrieval quality, and Hit@1 for the final recommendation. For agents, retrieval metrics use the Top-100 results of the last non-redundant natural search, while Hit@1 measures whether the final recommended item belongs to the target set. For standalone retrievers, Hit@1 equals Recall@1. Target injection is disabled during evaluation.

Baselines. We compare SimRec against these baselines: standalone retrievers (BM25 Robertson & Zaragoza (2009), Qwen3 Dense Zhang et al. (2025a), and BLAIR-BASE/LARGE Hou et al. (2026)); training-free LLM-based recommenders (GP-VisualPrompt Xi et al. (2025) and ChatCRS Li et al. (2025a)); trainable LLM-based recommenders (Rec-R1 Lin et al. (2025), RetPO Yoon et al. (2025), and MiniELM Nguyen et al. (2025)); and LLM backbone baselines run within the SimRec framework (Qwen3-4B Base Yang et al. (2025), Qwen-Plus[1], DeepSeek-V3.2 DeepSeek-AI et al. (2025), DeepSeek-V4-Flash DeepSeek-AI et al. (2026), Kimi-K2.6 Team et al. (2026), and GLM-5.1 GLM-5-Team et al. (2026)). Agent-based methods use Qwen3 Dense by default; BM25 variants assess sparse retrieval.

Implementation Details. SimRec uses Qwen3-4B with Qwen3 Dense as its default retriever. Each search returns $K = 100$ items, with the top $k = 20$ visible to the agent. During training, target injection is enabled after at least $N = 2$ user turns if the selected target remains absent from the visible candidates. Feedback is generated by a fixed Qwen3.6-Plus simulator[2] by default; we evaluate shared-parameter self-play separately. GRPO uses $G = 8$, a learning rate of 5×10^{-7}, clipping coefficient $\epsilon_c = 0.2$, and KL coefficient $\beta = 10^{-3}$. Agent baselines share the tool interface, search budget, candidate visibility, simulator protocol, and decoding constraints. Further details are provided in the Appendix A.

4.2 MAIN RESULTS

Overall Performance. Tool access alone does not realize the benefit of interaction. On ESCI (Table 1), SimRec-4B improves NDCG@100, Recall@100, and Hit@1 from 32.23/72.03/19.31 to 35.89/78.47/24.42, respectively, outperforming all compared methods on the aggregate metrics under the same retriever and tool setting. NDCG@100 improves over Qwen3-4B Base in all five product categories, most strongly in Electronics (+7.66 percentage points). SimRec also exceeds the strongest alternative backbone's NDCG (GLM-5.1, 34.99) and the task-aligned baselines. These results support the introduction's distinction between supplying an agent with tools and learning how to use them over a complete interaction; the behavioral analyses below examine what changes in the policy.

Retrieval coverage and final choice remain distinct bottlenecks. On Amazon C4, SimRec improves NDCG in all four categories, reaching the best aggregate NDCG@100 of 25.39 versus 22.94 for Qwen3-4B Base. The final-item Hit@1 gain (+8.44 points) accompanies a much smaller Recall@100 gain (+1.34 points). However, BLAIR-LARGE, an Amazon-trained retriever, achieves

[1] https://help.aliyun.com/en/model-studio/qwen-plus

[2] https://help.aliyun.com/en/model-studio/qwen3-6-plus

Table 1: Main results on ESCI and Amazon C4. Category columns report NDCG@100, and the AVG columns report NDCG@100, Recall@100, and Hit@1. *Elec.* denotes Electronics. *SimRec: LLM Backbone Baselines* evaluates different LLM backbones under the same SimRec framework. Unless explicitly marked with BM25, SimRec uses Qwen3 Dense as the retriever. Best and second-best distinct results in each column are **bolded** and underlined, respectively.

Method	ESCI								Amazon C4						
	NDCG@100					Avg.			NDCG@100				Avg.		
	Baby	Office	Sports	Game	Elec.	NDCG	Recall	Hit@1	Baby	Office	Sports	Game	NDCG	Recall	Hit@1
Standalone Retrieval Baselines															
BM25	15.20	24.16	20.00	12.44	22.14	19.68	53.94	12.76	6.40	8.70	7.28	7.82	7.68	21.84	2.61
Qwen3 Dense	29.06	35.17	31.74	34.64	33.07	32.62	75.69	19.98	17.34	18.66	21.77	18.49	19.80	52.32	6.45
BLAIR-BASE	16.93	22.41	20.37	14.64	14.70	19.09	55.94	9.77	20.95	17.99	22.27	19.72	20.42	57.72	5.87
BLAIR-LARGE	17.15	22.17	19.66	19.03	17.25	19.54	57.60	9.10	23.26	20.11	25.65	**25.27**	23.42	**63.01**	7.14
Training-Free LLM-based Recommenders															
GP-VisualPrompt	28.31	36.48	33.47	35.19	32.99	33.64	76.03	22.31	23.14	19.11	24.17	18.37	21.82	56.74	7.14
ChatCRS	32.46	34.64	33.60	38.02	35.15	34.36	76.80	22.31	21.87	20.08	24.65	17.82	22.12	57.96	7.61
Trainable LLM-based Recommenders															
Rec-R1	27.39	34.39	30.98	35.00	33.46	32.01	57.20	20.75	19.34	16.55	19.13	16.22	18.04	50.55	4.59
RetPO (DPO)	23.16	28.28	25.91	31.27	34.29	27.65	61.26	22.64	20.85	18.85	23.30	19.39	21.12	56.04	6.40
MiniELM (SFT+DPO)	28.66	34.57	32.02	35.85	33.92	32.81	74.70	21.42	22.29	18.62	22.59	19.21	20.92	54.35	6.97
SimRec: LLM Backbone Baselines															
Qwen3-4B Base	28.61	33.62	31.93	35.42	31.36	32.23	72.03	19.31	21.01	20.59	25.80	20.89	22.94	56.21	3.54
Qwen-Plus	29.32	32.40	30.42	36.27	33.61	31.73	70.92	12.21	17.79	16.31	20.89	16.16	18.50	48.61	12.54
DeepSeek-V3.2	22.45	31.13	32.02	34.64	31.61	31.00	70.81	11.21	**23.93**	**23.03**	26.71	21.39	24.62	58.36	13.36
DeepSeek-V4-Flash	29.94	35.15	34.21	34.65	33.53	33.89	73.14	12.87	20.71	19.46	24.28	18.40	21.64	54.88	11.50
Kimi-K2.6	30.39	35.60	33.71	**38.53**	35.75	34.50	77.25	15.65	21.28	18.81	24.14	17.46	21.36	53.19	**15.16**
GLM-5.1	**32.93**	35.56	34.42	38.37	34.95	34.99	77.80	13.87	20.86	19.61	24.22	17.93	21.65	52.90	13.06
SimRec-4B	32.67	**36.99**	**34.87**	38.09	**39.02**	**35.89**	**78.47**	**24.42**	23.39	22.69	**28.70**	22.54	**25.39**	57.55	11.98
+ Self-Play	32.46	35.39	34.39	37.59	34.23	34.72	75.81	22.42	17.88	18.83	21.69	16.90	19.77	49.54	7.14
+ BM25	21.06	25.04	23.81	21.54	28.51	24.02	65.37	14.76	15.32	14.96	16.36	17.91	15.88	40.77	5.75
+ Self-Play + BM25	24.46	29.72	29.80	22.22	26.11	26.46	56.05	12.98	15.24	15.09	15.81	15.93	15.50	39.43	6.04

Table 2: Component ablation on Amazon C4.

Variant	NDCG@100	Hit@1
Full SimRec	**25.39**	**11.98**
w/o Preference	21.17 (−4.22)	7.65 (−4.33)
w/o Item Detail	23.42 (−1.97)	6.04 (−5.94)
w/o Feedback	22.14 (−3.25)	8.06 (−3.92)

Table 3: Reward ablation on the ESCI.

Reward Setting	NDCG@100	Hit@1
Full Reward	**32.38**	**17.22**
w/o Hit@1	25.45 (−6.93)	14.28 (−2.94)
w/o Retrieval	26.18 (−6.20)	14.58 (−2.64)
w/o Tool	25.49 (−6.89)	14.34 (−2.88)

Note: Red values in parentheses denote percentage-point drops from the full setting.

higher Recall (63.01 vs. 57.55), while larger LLMs such as Kimi-K2.6 achieve higher Hit@1. SimRec nevertheless achieves the highest aggregate NDCG@100 among the compared methods. This suggests that its main advantage lies in better candidate ranking and final decisions rather than retrieval recall alone.

Different Retrieval Backends. To test robustness to the retriever, we replace Qwen3 Dense with BM25. SimRec-4B improves NDCG@100/Recall@100 from 19.68/53.94 to 24.02/65.37 on ESCI and from 7.68/21.84 to 15.88/40.77 on Amazon C4, showing that its gains persist with a weaker sparse backend. The larger relative gains further suggest that interaction can partly compensate for weaker retrieval. Retrieval gains also extend to the 1.7B backbone (Appendix A.4); the evaluated general benchmarks show no systematic degradation after post-training (Appendix A.5).

4.3 Ablation Study

Different evidence sources address different decision bottlenecks. We conduct component ablations on Amazon C4 (Table 2). Removing the User Preference Tool causes the largest drop in NDCG@100 (4.22 points), whereas removing the Item Detail Tool causes the largest drop in Hit@1 (5.94 points). This pattern suggests that user preferences primarily help guide search, whereas item details are particularly useful for verifying candidates before the final decision. Removing multi-

turn feedback lowers NDCG@100 and Hit@1 by 3.25 and 3.92 points, suggesting that item-detail and preference do not recover the full benefit of the user feedback. Further domain-level analyses are provided in Appendix B.

Intermediate rewards and the final hit serve complementary purposes. During the first 500 training steps on ESCI (Table 3), removing retrieval or tool-use rewards reduces Hit@1 by 2.64 or 2.88 points, despite retaining the final-hit objective. Thus, rewarding evidence acquisition benefits the eventual decision, not just the rewarded intermediate behavior. Conversely, removing the hit reward lowers retrieval NDCG by 6.93 points as well as Hit@1 by 2.94 points; auxiliary signals do not replace the task outcome. Together, the results support combining evidence-acquisition and outcome signals in a single trajectory objective.

4.4 Analysis

We examine whether the aggregate gains reflect the interaction behaviors highlighted in the introduction: useful feedback, recovery after rejection, tool use, and candidate ranking. Training diagnostics then examine how natural retrieval evolves alongside target injection.

4.4.1 User Simulation Modes

Self-play is viable, but its utility depends on the interaction setting. The main results compare training with a fixed external simulator and shared-parameter self-play (Table 1). With dense retrieval, self-play is close to external simulation on ESCI (34.72 vs. 35.89 NDCG@100), but not on Amazon C4 (19.77 vs. 25.39); with BM25, it leads on ESCI (26.46 vs. 24.02) and nearly matches external simulation on Amazon C4 (15.50 vs. 15.88). A fixed simulator may provide a more stationary feedback source, but model capacity and parameter sharing also differ, so the comparison does not isolate stability. Self-play offers a way to train without a separate simulator, not a universally equivalent replacement (Appendix B).

Table 4: Feedback-mode comparison on ESCI.

Feedback Mode	NDCG@10 (%) ↑	Hit@1 (%) ↑	Invalid Rate‡ (%) ↓
Fixed Feedback†	<u>31.52</u>	18.65	**11.90**
Qwen3-4B Base	31.10	<u>19.53</u>	30.70
SimRec-4B (Self-Play)	**32.15**	**20.64**	<u>14.00</u>

† Fixed Feedback always returns: "This recommendation does not meet my needs." ‡ Invalid Rate: malformed outputs or invalid tool calls.

The self-play simulator balances recommendation quality and interaction stability. In the separate ESCI feedback-mode comparison (Table 4), Fixed Feedback minimizes invalid recommender outputs or tool calls (11.90%), yet self-play achieves better NDCG@10 and Hit@1 (32.15 and 20.64 vs. 31.52 and 18.65). Self-play also reduces invalid outputs from 30.70% with the base simulator to 14.00%. Its shorter median feedback (114 vs. 195 characters) is descriptive, not evidence that length causes the difference. The useful distinction is task guidance versus mere continuation of dialogue.

Simulator choice affects recommendation performance. We evaluate SimRec with GLM-5.1 and Kimi-K2.6 in place of the default Qwen3.6-Plus simulator, keeping the remaining setup fixed. GLM-5.1 improves NDCG, Recall and Hit@1 to 36.93%, 79.02%, and 26.42%, respectively. Kimi-K2.6 slightly improves Recall@100 but lowers NDCG@100 and Hit@1 (Appendix B.2). These results show that recommendation performance is sensitive to the choice of simulator. In particular, more informative simulated feedback may improve downstream interaction quality.

4.4.2 Recovery Analysis

SimRec shows stronger recovery from initial misses. We assess whether policy training improves the ability to recover after an initial miss. We measure recovery as the fraction of first-attempt misses that become hits by the final recommendation. On ESCI, Hit@1 rises from 18.09% to 19.31% for Qwen3-4B Base, and from 20.64% to 24.42% for SimRec-4B (Figure 3). Among first-attempt misses, SimRec-4B recovers 4.76% by the final recommendation, compared with 1.49% for Qwen3-4B Base, consistent with more effective revision after user feedback. The improvement from the first to the final

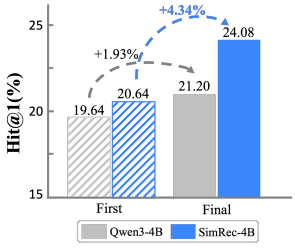


Figure 3: Recovery analysis.

recommendation shows that SimRec can recover from initial misses, and policy training further strengthens this capability.

4.4.3 CONTROLLED RERANKING

Interaction training also improves reranking ability. To disentangle retrieval recall from candidate reranking ability, we construct a controlled generative reranking task on ESCI. For each query, all models receive the same 16 candidate items and rerank them from a randomized input order, while retrieval scores and source information are hidden. SimRec-4B improves overall NDCG@10 from 24.75 to **28.06** over Qwen3-4B Base, while self-play reaches 25.57 (Table 5). With the fixed candidate set, this gain indicates SimRec improves candidate comparison and product-level ranking based on query and item evidence through training. Dataset construction details are provided in Appendix A.3.

Table 5: Controlled ESCI reranking results (NDCG@10, %).

Model	Baby	Office	Sports	Video Games	Electronics	Overall
Random-shuffled Input	20.91	26.05	18.39	14.49	13.56	19.73
Qwen3-4B Base	29.27	27.33	23.51	23.57	21.08	24.75
SimRec-4B (Self-Play)	33.63	28.18	22.99	**27.63**	21.97	25.57
SimRec-4B	**35.06**	**31.48**	**25.54**	24.07	**27.21**	**28.06**

4.4.4 TRAINING DYNAMICS

Natural retrieval improves while target injection declines. From the first to the last training window, the Top-10 hit rate rises from 61.81% to 68.52%, indicating that targets are increasingly ranked near the top. Meanwhile, the Top-100 miss rate falls from 19.70% to 14.50%, indicating improved retrieval coverage. Early in training, 38.18% of eligible trajectories require target injection; even after multiple turns, the target is still absent from the candidates visible to the agent. The injection rate subsequently declines to 31.48% as natural retrieval improves. Detailed statistics are provided in Appendix B.3.

SimRec learns a more active tool-use policy. SimRec also exhibits a more active evidence-seeking policy, with average tool use increasing from 2.43 to 5.19 calls per session. Most of this increase comes from item-detail inspection (0.58 to 1.83 calls) and preference-tool calls (0.02 to 0.86), indicating greater use of both product- and user-specific evidence. Moreover, sessions with more calls have lower average task reward (Appendix B.4), consistent with harder sessions prompting further investigation. Together with the component ablations, these results suggest that training changes how SimRec gathers evidence (Section 4.3) during interaction.

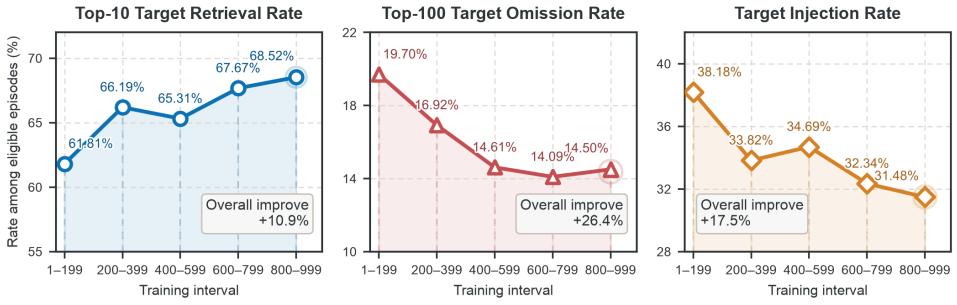


Figure 4: Training dynamics of natural retrieval and target injection.

5 CONCLUSION

We present **SimRec**, which trains a tool-integrated recommendation agent over complete interactions using task-anchored feedback and trajectory-level rewards. SimRec outperforms retrieval, training-free, and task-aligned baselines in overall NDCG@100 on ESCI and Amazon C4, while improving over its untrained 4B backbone. Fixed-candidate reranking and recovery analyses further show better product selection and more effective revision after rejection. These results demonstrate the value of learning to gather evidence, recommend, and revise within a single interaction policy.

AI USE STATEMENT

Generative AI models were used as components of the experimental framework in this work. In particular, LLMs were used to simulate user responses during multi-turn recommendation interactions, including the external user simulator and the self-play setting described in Section 3. Simulated feedback was generated during both training and evaluation according to the task-specific simulator prompts. Generative AI was also used in the construction of long-term user-preference profiles by summarizing the purchase histories provided to the preference tool. These uses are part of the proposed experimental system and are described in the corresponding methodology and prompt templates. Separately, generative AI tools were used to assist the research process, including polishing the writing of the manuscript for clarity and readability, supporting literature search and discovery, research brainstorming, and assisting with software code through editing, debugging, and refinement. Generative AI was not used to determine ground-truth relevance labels, formulate the reported experimental results, or make decisions about their interpretation. All AI-assisted research outputs were reviewed by the authors. Information and literature identified with AI assistance were checked against the original sources; AI-assisted code was reviewed, tested, and validated; and generated manuscript edits were checked to ensure that they accurately reflect the methodology, experiments, and conclusions reported in the paper. The authors take responsibility for the final text, code, experimental results, and other artifacts presented in this work.

ETHICS STATEMENT

This work evaluates recommendation agents using the ESCI and Amazon C4 benchmarks. For Amazon C4, histories from Amazon Reviews 2023 are processed into long-term preference summaries after deduplication and leakage filtering. Agent interactions during training and evaluation use simulated user feedback; no participants were recruited, and the experiments do not make purchases or interact with live shoppers. The results therefore do not establish how SimRec would perform or behave with real users. Deployment would require additional evaluation of recommendation accuracy, potentially biased or misleading product claims, and the privacy implications of using shopping histories, together with appropriate user controls and oversight.

REPRODUCIBILITY STATEMENT

Section 3 specifies SimRec's tool interface, task-anchored user simulation, trajectory reward, and GRPO training procedure, including training-only target injection, which is disabled during evaluation. Section 4 describes the ESCI and Amazon C4 evaluation, model backbones, retrieval backends, baselines, metrics, and shared agent settings. The appendix provides implementation hyperparameters, category-level results, controlled reranking construction, additional ablations and diagnostics, and the prompt templates used for the agent, user simulator, reranking task, and preference-profile construction. Our code is available at `https://anonymous.4open.science/r/SimRec-6F0D/`.

REFERENCES

Jafar Afzali, Aleksander Mark Drzewiecki, Krisztian Balog, and Shuo Zhang. UserSimCRS: A User Simulation Toolkit for Evaluating Conversational Recommender Systems. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pp. 1160–1163. ACM, 2023. ISBN 978-1-4503-9407-9. doi: 10.1145/3539597.3573029. URL https://dl.acm.org/doi/10.1145/3539597.3573029.

Krisztian Balog and ChengXiang Zhai. User Simulation for Evaluating Information Access Systems. In *Proceedings of the Annual International ACM SIGIR Conference on Research and Development in Information Retrieval in the Asia Pacific Region*, pp. 302–305. ACM, 2023. ISBN 979-8-4007-0408-6. doi: 10.1145/3624918.3629549. URL https://dl.acm.org/doi/10.1145/3624918.3629549.

Nolwenn Bernard and Krisztian Balog. Towards a Formal Characterization of User Simulation Objectives in Conversational Information Access. In *Proceedings of the 2024 ACM SIGIR Inter-*

national Conference on Theory of Information Retrieval, pp. 185–193. ACM, 2024. doi: 10.1145/3664190.3672529. URL https://dl.acm.org/doi/10.1145/3664190.3672529.

Keping Bi, Qingyao Ai, Yongfeng Zhang, and W. Bruce Croft. Conversational Product Search Based on Negative Feedback. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 359–368. ACM, 2019. doi: 10.1145/3357384.3357939. URL https://dl.acm.org/doi/10.1145/3357384.3357939.

Nicolas Bougie and Narimawa Watanabe. 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)*, pp. 43–60, 2025. URL https://aclanthology.org/2025.acl-industry.5/.

Shihao Cai, Chongming Gao, Haoyan Liu, Wentao Shi, Jianshan Sun, Ruiming Tang, and Fuli Feng. MGFRec: Towards Reinforced Reasoning Recommendation with Multiple Groundings and Feedback. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, KDD '26, pp. 49–58, New York, NY, USA, April 2026. Association for Computing Machinery. doi: 10.1145/3770854.3780206.

Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. Towards Conversational Recommender Systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 815–824. ACM, 2016. doi: 10.1145/2939672.2939746. URL https://dl.acm.org/doi/10.1145/2939672.2939746.

DeepSeek-AI, Aixin Liu, Aoxue Mei, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, et al. DeepSeek-V3.2: Pushing the Frontier of Open Large Language Models, 2025. URL http://arxiv.org/abs/2512.02556.

DeepSeek-AI, Anyi Xu, Bangcai Lin, Bing Xue, Bingxuan Wang, Bingzheng Xu, Bochao Wu, Bowei Zhang, Chaofan Lin, Chen Dong, Chenchen Ling, Chengda Lu, et al. DeepSeek-V4: Towards Highly Efficient Million-Token Context Intelligence, 2026. URL http://arxiv.org/abs/2606.19348.

Hanwen Du, Bo Peng, and Xia Ning. SAPIENT: Mastering multi-turn conversational recommendation with strategic planning and Monte Carlo tree search. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 2629–2648, 2025. URL https://aclanthology.org/2025.naacl-long.133/.

Jiabao Fang, Shen Gao, Pengjie Ren, Xiuying Chen, Suzan Verberne, and Zhaochun Ren. A multi-agent conversational recommender system, 2024.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *Advances in Neural Information Processing Systems*, 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. URL https://aclanthology.org/2025.findings-acl.307/.

GLM-5-Team, Aohan Zeng, Xin Lv, Zhenyu Hou, Zhengxiao Du, Qinkai Zheng, Bin Chen, et al. GLM-5: From Vibe Coding to Agentic Engineering, February 2026.

Yupeng Hou, Jiacheng Li, Xiangjun Fu, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation: Benchmarking LLMs as semantic encoders. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 3251–3265, 2026.

Chengkai Huang, Hongtao Huang, Tong Yu, Kaige Xie, Junda Wu, Shuai Zhang, Julian Mcauley, Dietmar Jannach, and Lina Yao. A Survey of Foundation Model-Powered Recommender Systems: From Feature-Based, Generative to Agentic Paradigms, 2025a. URL http://arxiv.org/abs/2504.16420.

Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. Recommender AI Agent: Integrating Large Language Models for Interactive Recommendations. *ACM Transactions on Information Systems*, 43(4):1–33, 2025b. ISSN 1046-8188, 1558-2868. doi: 10.1145/3731446. URL https://dl.acm.org/doi/10.1145/3731446.

Zheng Hui, Xiaokai Wei, Yexi Jiang, Kevin Gao, Chen Wang, Se-eun Yoon, Rachit Pareek, and Michelle Gong. Toward safe and human-aligned game conversational recommendation via multi-agent decomposition. In *Findings of the Association for Computational Linguistics: EACL 2026*, pp. 4568–4584, 2026. URL https://aclanthology.org/2026.findings-eacl.238/.

Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. A Survey on Conversational Recommender Systems. *ACM Computing Surveys*, 54(5):1–36, 2022. ISSN 0360-0300, 1557-7341. doi: 10.1145/3453154. URL https://dl.acm.org/doi/10.1145/3453154.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-R1: Training LLMs to reason and leverage search engines with reinforcement learning, 2025.

SungHwan Kim, Kwangwook Seo, Tongyoung Kim, Jinyoung Yeo, and Dongha Lee. Stop Playing the Guessing Game! Evaluating Conversational Recommender Systems via Target-free User Simulation. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 19588–19605, 2025. URL https://aclanthology.org/2025.findings-emnlp.1067/.

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, 2025a. URL https://aclanthology.org/2025.findings-naacl.17/.

Tianyi Li, Zixuan Wang, Guidong Lei, Xiaodong Li, and Hui Li. AgenticRec: A Recommendation-Oriented Agentic Framework with Progressive Tool-Integrated Reasoning Optimization, 2026. URL http://arxiv.org/abs/2603.21613.

Xiangci Li, Zhiyu Chen, Jason Ingyu Choi, Nikhita Vedula, Besnik Fetahu, Oleg Rokhlenko, and Shervin Malmasi. 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)*, pp. 13095–13120, 2025b. URL https://aclanthology.org/2025.acl-long.641/.

Jiacheng Lin, Tian Wang, and Kun Qian. Rec-R1: Bridging Generative Large Language Models and User-Centric Recommendation Systems via Reinforcement Learning, 2025. URL http://arxiv.org/abs/2503.24289.

Xinyu Lin, Yashar Deldjoo, Sunhao Dai, Honghui Bao, Xiaopeng Ye, Fatemeh Nazary, Wenjie Wang, Tommaso Di Noia, Jun Xu, and Tat-Seng Chua. Autonomous Information Seeking: A Roadmap for Agentic Recommender Systems, 2026. URL http://arxiv.org/abs/2607.04433.

Heli Ma, Jie Zou, Mohammad Aliannejadi, Evangelos Kanoulas, Yi Bin, and Yang Yang. Ask or Recommend: An Empirical Study on Conversational Product Search. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pp. 3927–3931. ACM, 2024. doi: 10.1145/3627673.3679875. URL https://dl.acm.org/doi/10.1145/3627673.3679875.

Ofer Meshi, Krisztian Balog, Sally Goldman, Avi Caciularu, Guy Tennenholtz, Jihwan Jeong, Amir Globerson, and Craig Boutilier. Convapparel: A benchmark dataset and validation framework for user simulators in conversational recommenders. In *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5270–5304, 2026. URL https://aclanthology.org/2026.eacl-long.244/.

Duy A. Nguyen, Rishi Kesav Mohan, Shimeng Yang, Pritom Saha Akash, and Kevin Chen-Chuan Chang. MiniELM: A Lightweight and Adaptive Query Rewriting Framework for E-Commerce

Search Optimization. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 6952–6964. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-acl.363. URL https://aclanthology.org/2025.findings-acl.363/.

Bo Ni, Yu Wang, Leyao Wang, Branislav Kveton, Franck Dernoncourt, Yu Xia, Hongjie Chen, Reuben Luera, Samyadeep Basu, Subhojyoti Mukherjee, Puneet Mathur, Nesreen K. Ahmed, Junda Wu, Li Li, Huixin Zhang, Ruiyi Zhang, Tong Yu, Sungchul Kim, Jiuxiang Gu, Zhengzhong Tu, Alexa Siu, Zichao Wang, Seunghyun Yoon, Nedim Lipka, Namyong Park, Zihao Lin, Trung Bui, Yue Zhao, Tyler Derr, and Ryan A. Rossi. A Survey on LLM-based Conversational User Simulation. In Vera Demberg, Kentaro Inui, and Lluís Marquez (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 4266–4301, Rabat, Morocco, March 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.eacl-long.200.

Cheng Qian, Emre Can Acikgoz, Qi He, Hongru Wang, Xiusi Chen, Dilek Hakkani-Tür, Gokhan Tur, and Heng Ji. ToolRL: Reward is all tool learning needs. In *Advances in Neural Information Processing Systems*, 2025.

Chandan K. Reddy, Lluís Màrquez, Fran Valero, Nikhil Rao, Hugo Zaragoza, Sambaran Bandyopadhyay, Arnab Biswas, Anlu Xing, and Karthik Subbian. Shopping Queries Dataset: A Large-Scale ESCI Benchmark for Improving Product Search, June 2022.

Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework: BM25 and beyond. *Foundations and trends® in information retrieval*, 4(1–2):1–174, 2009. URL https://www.emerald.com/ftinr/article-abstract/4/1-2/1/1326508.

Yu Shang, Peijie Liu, Yuwei Yan, Zijing Wu, Leheng Sheng, Yuanqing Yu, Chumeng Jiang, An Zhang, Fengli Xu, and Yu Wang. Agentrecbench: Benchmarking llm agent-based personalized recommender systems. *Advances in Neural Information Processing Systems*, 38, 2026. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/e2d6f7249add096e26679eade1b4cc6f-Abstract-Datasets_and_Benchmarks_Track.html.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models, 2024. URL http://arxiv.org/abs/2402.03300.

Yunxiao Shi, Wujiang Xu, Zhang Zeqi, Xing Zi, Qiang Wu, and Min Xu. PersonaX: A recommendation agent-oriented user modeling framework for long behavior sequence. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 5764–5787, 2025. URL https://aclanthology.org/2025.findings-acl.300/.

Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, S. H. Cai, Yuan Cao, Ziwei Chai, Y. Charles, H. S. Che, et al. Kimi K2.5: Visual Agentic Intelligence, August 2026.

Kuang Wang, Xianfei Li, Shenghao Yang, Li Zhou, Feng Jiang, and Haizhou Li. 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)*, pp. 21082–21107, 2025a. URL https://aclanthology.org/2025.acl-long.1025/.

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*, 2025b.

Zongwei Wang, Min Gao, Hongzhi Yin, Junliang Yu, Tong Chen, Quoc Viet Hung Nguyen, Shazia Sadiq, and Tianrui Li. Self-Distilled Reinforcement Learning for Co-Evolving Agentic Recommender Systems, 2026. URL http://arxiv.org/abs/2604.10029.

Nan Xi, Jingjing Meng, Yietian Chen, Chaosheng Dong, Yan Gao, Yi Sun, and Junsong Yuan. Generative prompting for complex product retrieval. In *WWW Companion 2025 – Companion Proceedings of the ACM Web Conference 2025*, pp. 1412–1416. Association for Computing Machinery, 2025. doi: 10.1145/3701716.3715546. URL https://doi.org/10.1145/3701716.3715546.

Xingyuan Xiang, Xiangchen Pan, and Wei Wei. User Simulator-Guided Multi-Turn Preference Optimization for Reasoning LLM-based Conversational Recommendation, 2026. URL http://arxiv.org/abs/2604.03671.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, et al. Qwen3 Technical Report, May 2025.

Jingheng Ye, Yong Jiang, Xiaobin Wang, Yinghui Li, Yangning Li, Pengjun Xie, and Fei Huang. Productagent: Benchmarking conversational product search agent with asking clarification questions. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 383–398, 2025. URL https://aclanthology.org/2025.emnlp-industry.25/.

Chanwoong Yoon, Gangwoo Kim, Byeongguk Jeon, Sungdong Kim, Yohan Jo, and Jaewoo Kang. Ask optimal questions: Aligning large language models with retriever's preference in conversation. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 5914–5936, 2025. URL https://aclanthology.org/2025.findings-naacl.328/.

Se-eun Yoon, Zhankui He, Jessica Echterhoff, and Julian McAuley. Evaluating Large Language Models as Generative User Simulators for Conversational Recommendation. In Kevin Duh, Helena Gomez, and Steven Bethard (eds.), *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 1490–1504, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.83.

An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. On generative agents in recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2024.

Haobo Zhang, Yutao Zhu, Kelong Mao, Tianhao Li, and Zhicheng Dou. RecThinker: An Agentic Framework for Tool-Augmented Reasoning in Recommendation, 2026a. URL http://arxiv.org/abs/2603.09843.

Shuo Zhang and Krisztian Balog. Evaluating Conversational Recommender Systems via User Simulation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1512–1520. ACM, 2020. ISBN 978-1-4503-7998-4. doi: 10.1145/3394486.3403202. URL https://dl.acm.org/doi/10.1145/3394486.3403202.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, Fei Huang, and Jingren Zhou. Qwen3 Embedding: Advancing Text Embedding and Reranking Through Foundation Models, June 2025a.

Yu Zhang, Shutong Qiao, Jiaqi Zhang, Tzu-Heng Lin, Chen Gao, and Yong Li. A Survey of Large Language Model Empowered Agents for Recommendation and Search: Towards Next-Generation Information Retrieval. *ACM Trans. Inf. Syst.*, September 2026b. ISSN 1046-8188. doi: 10.1145/3842663.

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

Yuyue Zhao, Jiancan Wu, Xiang Wang, Wei Tang, Dingxian Wang, and Maarten De Rijke. Let Me Do It For You: Towards LLM Empowered Recommendation via Tool Learning. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1796–1806. ACM, 2024. doi: 10.1145/3626772.3657828. URL https://dl.acm.org/doi/10.1145/3626772.3657828.

Lixi Zhu, Xiaowen Huang, and Jitao Sang. How reliable is your simulator? Analysis on the limitations of current LLM-based user simulators for conversational recommendation, 2024.

Yaochen Zhu, Harald Steck, Dawen Liang, Yinhan He, Vito Ostuni, Jundong Li, and Nathan Kallus. Rank-grpo: Training llm-based conversational recommender systems with reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 155483–155512, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/fb8f2d4e90018c0fae060dff09a91ce3-Abstract-Conference.html.

Jie Zou, Mohammad Aliannejadi, Evangelos Kanoulas, Shuxi Han, Heli Ma, Zheng Wang, Yang Yang, and Heng Tao Shen. PSCon: Toward Conversational Product Search. *CoRR*, 2025.

A Additional Experimental Details

A.1 Implementation

The recommender is initialized from Qwen3-4B. The retrieval backend returns $K = 100$ items, with the top $k = 20$ visible by default. Training-only target injection is enabled after $N = 2$ user turns if the target remains absent from the visible list. GRPO uses learning rate 5×10^{-7}, rollout batch size 8, group size $G = 8$, temperature 1.0, top-p 1.0, clipping coefficient 0.2, and KL coefficient 10^{-3}. The external simulator is Qwen3.6-Plus, with temperature 0.2, maximum output length 220 tokens and seed 42. Rewards are clipped to $[0, 2.5]$. Duplicate queries receive no additional valid-search bonus; unchanged feedback queries incur a 0.05 penalty. Length penalties start after four turns, increase by 0.02 per additional turn and are capped at 0.20. The recommendation reward weight and retrieval NDCG scale are both 1.0. The format reward is 0.01; valid search, item-detail, and preference bonuses are 0.03, 0.01, and 0.01.

A.2 All Results

A.3 Controlled Reranking Construction

We construct the ESCI reranking set with BM25 as the only upstream retriever. BM25 searches the product `contents` field with $k_1 = 1.2$ and $b = 0.75$, returns the Top-20 candidates, and filters out instances whose upstream NDCG@20 is zero. We retain the first 16 candidates after filtering, resulting in 345 valid examples from the 901-instance local ESCI test set across Baby Products, Office Products, Sports and Outdoors, Video Games, and Electronics. To prevent the model from copying the upstream order, each candidate list is shuffled with random seed 42, while `retrieval_score` and `retrieval_type` are hidden. The random input order serves as the no-reranking baseline.

Table 6: ESCI results across five categories. Category columns report NDCG@100 and Recall@100; AVG additionally reports Hit@1. Agent methods use Qwen3 Dense unless marked *(BM25)*; Standalone retrievers are listed by method name. Best and second-best distinct results in each metric column are bolded and underlined, respectively; ties receive the same formatting.

Method	Baby		Office		Sports		Video Games		Electronics		AVG		
	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	Hit@1
Retrieval Baselines													
BM25	15.20	45.63	24.16	63.04	20.00	53.92	12.44	41.75	22.14	58.25	19.68	53.94	12.76
Qwen3 Dense	29.06	70.87	35.17	78.80	31.74	75.25	34.64	78.64	33.07	73.79	32.62	75.69	19.98
BLAIR-BASE	16.93	54.37	22.41	63.59	20.37	58.09	14.64	41.75	14.70	49.51	19.09	55.94	9.77
BLAIR-LARGE	17.15	54.37	22.17	63.04	19.66	58.82	19.03	51.46	17.25	52.43	19.54	57.60	9.10
LLM Agent Baselines													
Qwen3-4B Base	28.61	67.96	33.62	72.28	31.93	72.55	35.42	74.76	31.36	70.87	32.23	72.03	19.31
Qwen-Plus	29.32	68.93	32.40	70.65	30.42	69.36	36.27	75.73	33.61	74.76	31.73	70.92	12.21
DeepSeek-V3.2	22.45	60.19	31.13	71.74	32.02	70.83	34.64	77.67	31.61	72.82	31.00	70.81	11.21
DeepSeek-V4-Flash	29.94	69.90	35.15	72.28	34.21	72.79	34.65	73.79	33.53	78.64	33.89	73.14	12.87
Kimi-K2.6	30.39	72.82	35.60	77.17	33.71	77.45	**38.53**	79.61	35.75	78.64	34.50	77.25	15.65
GLM-5.1	**32.93**	73.79	35.56	78.80	34.42	77.45	38.37	79.61	34.95	79.61	34.99	77.80	13.87
GP-VisualPrompt	28.31	70.87	36.48	**79.35**	33.47	76.23	35.19	71.84	32.99	78.64	33.64	76.03	22.31
ChatCRS	32.46	**75.73**	34.64	73.37	33.60	76.96	38.02	**80.58**	35.15	79.61	34.36	76.80	22.31
Task-Aligned Recommendation Methods													
Rec-R1	27.39	52.10	34.39	58.74	30.98	55.43	35.00	62.59	33.46	61.19	32.01	57.20	20.75
RetPO (DPO)	23.16	52.43	28.28	58.70	25.91	59.31	31.27	66.99	34.29	76.70	27.65	61.26	22.64
MiniELM (SFT+DPO)	28.66	69.90	34.57	77.72	32.02	74.51	35.85	74.76	33.92	74.76	32.81	74.70	21.42
SimRec-4B	32.67	74.76	**36.99**	77.72	**34.87**	**78.19**	38.09	**80.58**	**39.02**	**82.52**	**35.89**	**78.47**	**24.42**
SimRec-4B (Self-Play)	32.46	72.82	35.39	76.09	34.39	76.72	37.59	74.76	34.23	75.73	34.72	75.81	22.42
SimRec-4B (BM25)	21.06	61.17	25.04	69.57	23.81	65.69	21.54	55.34	28.51	70.87	24.02	65.37	14.76
SimRec-4B (Self-Play, BM25)	24.46	57.28	29.72	61.96	29.80	61.03	22.22	46.60	26.11	53.40	26.46	56.05	12.98

Table 7: Amazon C4 results across four categories. Category columns report NDCG@100 and Recall@100; AVG additionally reports Hit@1. Agent methods use Qwen3 Dense unless marked *(BM25)*; Standalone retrievers are listed by method name. Best and second-best results are bold and underlined.

Method	Baby		Office		Sports		Video Games		AVG		
	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	NDCG	Recall	Hit@1
Retrieval Baselines											
BM25	6.40	21.05	8.70	24.26	7.28	20.45	7.82	20.67	7.68	21.84	2.61
Qwen3 Dense	17.34	51.82	18.66	48.53	21.77	55.61	18.49	51.33	19.80	52.32	6.45
BLAIR-BASE	20.95	61.94	17.99	52.17	22.27	62.03	19.72	50.67	20.42	57.72	5.87
BLAIR-LARGE	23.26	**65.18**	20.11	**58.75**	25.65	**65.11**	**25.27**	**65.33**	23.42	**63.01**	7.14
LLM Agent Baselines											
Qwen3-4B Base	21.01	55.87	20.59	52.86	25.80	59.89	20.89	51.33	22.94	56.21	3.54
Qwen-Plus	17.79	48.99	16.31	46.79	20.89	51.34	16.16	41.33	18.50	48.61	12.54
DeepSeek-V3.2	**23.93**	57.89	**23.03**	56.67	26.71	61.76	21.39	48.67	24.62	58.36	13.36
DeepSeek-V4-Flash	20.71	57.09	19.46	51.82	24.28	58.69	18.40	44.00	21.64	54.88	11.50
Kimi-K2.6	21.28	52.23	18.81	50.43	24.14	57.49	17.46	44.00	21.36	53.19	**15.16**
GLM-5.1	20.86	53.85	19.61	51.13	24.22	56.15	17.93	42.00	21.65	52.90	13.06
GP-VisualPrompt	23.14	60.32	19.11	53.03	24.17	59.49	18.37	51.33	21.82	56.74	7.14
ChatCRS	21.87	56.28	20.08	55.98	24.65	62.03	17.82	48.00	22.12	57.96	7.61
Task-Aligned Recommendation Methods											
Rec-R1	19.34	56.74	16.55	49.38	19.13	50.43	16.22	45.47	18.04	50.55	4.59
RetPO (DPO)	20.85	58.70	18.85	52.69	23.30	59.09	19.39	49.33	21.12	56.04	6.40
MiniELM (SFT+DPO)	22.29	56.68	18.62	51.99	22.59	56.15	19.21	50.67	20.92	54.35	6.97
SimRec-4B	23.39	55.87	22.69	55.98	**28.70**	61.10	22.54	48.67	**25.39**	57.55	11.98
SimRec-4B (Self-Play)	17.88	46.56	18.83	49.57	21.69	51.87	16.90	42.67	19.77	49.54	7.14
SimRec-4B (BM25)	15.32	43.32	14.96	39.17	16.36	40.91	17.91	42.00	15.88	40.77	5.75
SimRec-4B (Self-Play, BM25)	15.24	41.70	15.09	38.99	15.81	38.90	15.93	40.00	15.50	39.43	6.04

Table 8: ESCI results with the 1.7B Qwen backbone. Retrieval metrics are percentages; Turn is the average number of interaction turns. Dense denotes the Qwen3 dense retrieval backend.

Setting	NDCG@100	Recall@100	Hit@1	Turns
Qwen3-1.7B Base + Dense	33.47	75.92	–	8.23
Qwen3-1.7B Base + BM25	19.63	53.50	–	7.49
SimRec + Dense	34.17	77.03	21.31	4.79
SimRec + BM25	29.91	71.81	18.20	3.84
SimRec (Self-Play) + Dense	34.42	76.25	21.53	4.00
SimRec (Self-Play) + BM25	25.76	65.48	17.20	4.02

A.4 Additional 1.7B Results

We additionally evaluate SimRec with a 1.7B Qwen backbone on ESCI. Table 8 reports results under dense and BM25 retrieval for both external simulation and shared-parameter self-play. These runs extend the evaluation to a smaller model scale. Turns denotes the average number of interaction turns and is reported as a descriptive efficiency statistic. – indicates that none of the 901 samples produces a natural recommendation, leaving no valid sample for Hit@1 computation.

A.5 General Benchmark Retention

Recommendation-oriented training retains performance across the evaluated general benchmarks, with higher scores on four of five. Performance improves on MMLU, GSM8K, IFEval, and MBPP, with HumanEval remaining within 0.61 percentage points of the base score.

Table 9: General-capability performance before and after recommendation training. Values are percentages; Δ denotes RL$-$Base in percentage points.

Benchmark	Base	RL	Δ
MMLU	55.83	**57.93**	$+2.10$
GSM8K	78.92	**82.49**	$+3.57$
IFEval	79.11	**81.70**	$+2.59$
MBPP pass@1	44.00	**52.00**	$+8.00$
HumanEval pass@1	**71.95**	71.34	-0.61

A.6 Domain-level Ablations and Tool Behavior

Across the four Amazon C4 categories, removing user preferences or multi-turn feedback generally reduces NDCG@100 and Recall@100, suggesting that both provide useful signals for interactive recommendation. Removing item-detail inspection does not show the same degradation in this ablation, indicating that its contribution may be more dependent on the specific interaction context (Table 10).

Table 10: Amazon C4 component ablations across four categories. Values are percentages reported as NDCG@100 / Recall@100; ALL is sample-weighted over 1,722 examples.

Ablation	Baby	Office	Sports	Video Games	ALL
w/o `get_user_preference`	20.72 / 55.06	19.09 / 51.65	23.30 / 56.82	19.34 / 46.00	21.17 / 53.89
w/o `get_item_details`	**24.88 / 69.64**	**21.24 / 60.49**	**25.13 / 64.71**	**20.93 / 60.67**	**23.42 / 63.65**
w/o multi-turn feedback	20.03 / 54.66	20.24 / 54.42	24.56 / 57.62	20.86 / 50.00	22.14 / 55.46

Table 11: Behavioral diagnostics for the Amazon C4 component ablations. These metrics characterize behavioral differences accompanying the main NDCG@100 and Recall@100 results.

Ablation	Turns	Search	Details	Preference	Eff. pref.	Feedback	Len. pen.
w/o `get_user_preference`	4.99	99.88%	4.53%	0%	0%	0.26%	0.0213
w/o `get_item_details`	6.71	100%	0%	100%	19.11%	0.58%	0.0542
w/o multi-turn feedback	3.99	99.77%	99.25%	76.77%	2.33%	0%	0

Table 12: Self-play optimization variants at step 500.

Setting	NDCG@100	Hit@1
Train user, no reward sharing	**21.75**	**9.38**
Train user, reward sharing	20.97	7.03
No user-token training	20.57	7.81

B Additional Ablation Analyses

B.1 Self-Play Optimization Analysis

We further study whether user-role tokens should be optimized and whether trajectory reward should be explicitly shared with the user role. As shown in Table 12, disabling user-token training gives the lowest NDCG@100, indicating that jointly updating the user role is beneficial for self-play optimization. In contrast, the difference between training with and without explicit reward sharing is comparatively limited and varies across metrics, with the no-sharing variant performing best at step 500. This suggests that updating user-role tokens is the more important design choice, whereas explicit reward sharing contributes less consistently.

B.2 Cross-Simulator Evaluation

To examine how the choice of user simulator affects SimRec, we conduct a cross-simulator evaluation using Qwen3.6-Plus, Kimi-K2.6, and GLM-5.1 as user simulators while keeping the remaining experimental configuration unchanged. Table 13 reports the average performance across categories.

The results show a clear sensitivity to the choice of user simulator. Relative to Qwen3.6-Plus, Kimi-K2.6 slightly increases Recall@100 by 0.22 percentage points, but decreases NDCG@100 by 1.48 points and Hit@1 by 2.00 points. In contrast, GLM-5.1 consistently improves all three metrics, yielding gains of 1.04, 0.55, and 2.00 percentage points in NDCG@100, Recall@100, and Hit@1, respectively.

These results indicate that the user simulator is not merely an interchangeable component: its quality can meaningfully affect the recommendation policy learned or evaluated through interaction. In particular, the stronger results obtained with GLM-5.1 suggest that higher-quality simulated feedback can provide a more effective interaction signal and translate into improved downstream recommendation performance. This cross-simulator comparison therefore further supports the role of the user simulator as an important component of SimRec, while also showing that the framework is compatible with multiple simulator backbones.

Table 13: Cross-simulator evaluation of SimRec. Δ denotes the absolute percentage-point change relative to Qwen3.6-Plus.

User Simulator	NDCG@100	Δ	Recall@100	Δ	Hit@1	Δ
Qwen3.6-Plus	35.89	–	78.47	–	24.42	–
Kimi-K2.6	34.41	-1.48	78.69	$+0.22$	22.42	-2.00
GLM-5.1	**36.93**	$\mathbf{+1.04}$	**79.02**	$\mathbf{+0.55}$	**26.42**	$\mathbf{+2.00}$

B.3 Additional Training Diagnostics

Table 14: Training-stage dynamics of target injection. Eligible episodes satisfy the minimum user-turn condition. The injection rate additionally requires that the target be absent from the model-visible top $k = 20$.

Training steps	Episodes	Eligible	Top-10 hit	Rank 11–100	Top-100 miss	Injection	Inj./eligible
1–199	6,368	64.62%	61.81%	18.49%	19.70%	24.67%	38.18%
200–399	6,400	71.47%	66.19%	16.90%	16.92%	24.17%	33.82%
400–599	6,400	72.92%	65.31%	20.08%	14.61%	25.30%	34.69%
600–799	6,400	70.30%	67.67%	18.24%	14.09%	22.73%	32.34%
800–999	6,240	72.18%	68.52%	16.98%	14.50%	22.72%	31.48%

Across 994 rollout files, there are 31,808 episodes and 54,352 searches; 7,610 episodes contain target injection. Comparing the first and last 50 rollout files, natural Top-10 hit rises from 56.29% to 65.20%, natural Top-100 hit rises from 75.14% to 84.45%, and episode-level injection falls from 28.62% to 24.75%. These windows provide a complementary view of the stage-level training dynamics.

B.4 Tool-Use Behavior Analysis

We further analyze the tool-use behavior of SimRec on the test sessions. We compare against the Qwen3-4B base policy evaluated on the same test set.

Tool-use frequency. Table 15 summarizes the average number of tool calls and interaction turns per session. SimRec invokes external tools more actively than the base policy, with 5.19 tool calls per session compared with 2.43 for Qwen3-4B. The largest increases come from item-detail inspection (1.83 vs. 0.58 calls per session) and preference queries (0.86 vs. 0.02), while the increase in search calls is more moderate (2.50 vs. 1.83). SimRec also produces longer interaction trajectories on average (9.05 vs. 6.28 turns).

Importantly, the tool-use reward in SimRec is not assigned independently to every tool invocation. Instead, it serves as a trajectory-level binary shaping signal for each tool type: once a tool type has been invoked validly within a session, subsequent calls to the same tool type do not provide additional tool-use reward. Multiple calls remain permissible when they are needed for task completion—for example, the policy may inspect several different candidate items using the item-detail tool—but these additional calls do not accumulate extra R_{tool}. The purpose of this reward component is therefore to encourage the policy to learn valid use of the available tools, rather than to reward a larger number of calls.

Tool-call count and task performance. To examine whether the higher tool-use frequency could itself provide a systematic reward advantage, we stratify SimRec trajectories by their total number of tool calls. The results are shown in Table 16. Recommendation quality and final trajectory reward do not increase with the number of tool calls. Sessions with zero to two calls achieve the highest NDCG@100, Recall@100, Hit@1, and average reward, whereas sessions involving five or more calls are substantially more difficult.

The strong association between longer tool-use trajectories and lower task performance should not be interpreted as evidence that additional tool use causes worse recommendations. A more plausible

Table 15: Tool-use statistics on the Amazon session test set. SimRec uses external tools more actively than the Qwen3-4B base policy, particularly the item-detail and preference tools.

Metric	SimRec	Qwen3-4B Base
Tool calls / session	5.19	2.43
Search calls / session	2.50	1.83
Item-detail calls / session	1.83	0.58
Preference calls / session	0.86	0.02
Turns / session	9.05	6.28

Table 16: SimRec performance stratified by the total number of tool calls in a session. The groups are observational rather than controlled: easier sessions tend to terminate after fewer calls, whereas unresolved or more difficult sessions naturally induce longer trajectories and additional tool use. The table should therefore not be interpreted as estimating the causal effect of additional tool calls on recommendation performance.

Tool Calls	# Sessions	NDCG@100	Recall@100	Hit@1	Avg. Reward
0–2	171	0.8913	0.9942	0.8012	1.9058
3–4	428	0.3011	0.7640	0.0467	0.4799
≥ 5	1323	0.1254	0.4497	0.0098	0.2955

explanation is selection by task difficulty: easy sessions can often be resolved after the initial retrieval or a small number of inspections, while difficult or unresolved sessions trigger further search, item inspection, and preference consultation. Nevertheless, this analysis rules out a simple interpretation in which repeatedly invoking tools provides a reliable way to increase the trajectory-level objective. In particular, repeated calls to an already used tool type cannot accumulate additional R_{tool}, and empirically the final reward does not increase with total tool-call count.

Overall, the analysis suggests that post-training changes not only the final ranking behavior but also the interaction policy: SimRec is more willing to consult external evidence, especially item details and user preferences, when the current recommendation remains unresolved. At the same time, R_{tool} should be understood as a lightweight tool-validity shaping signal rather than a direct measure of the marginal utility of each individual tool invocation.

C Prompts Used in the Recommendation Framework

This appendix reports the prompts used for the recommendation agent, user simulator, product reranking, and user-preference-profile construction. Text enclosed in braces (e.g., {`reference_query`}) denotes a runtime placeholder that is replaced with the corresponding instance-specific value. The recommendation and auxiliary reranking tasks use distinct output protocols; the prompts below are reproduced verbatim.

C.1 Recommendation Agent

Recommendation Agent Prompt

```
You are the recommender in a shopping dialogue.
Your goal is to find the single best catalog item for the shopper.
Use retrieval evidence before recommending. Do not guess product
    facts.
Keep each tool decision concise and reserve enough space for the
    final structured recommendation.

Valid output modes:
```

```
1. Tool-call mode
- Use this mode when you need search results or item details.
- Output exactly one JSON tool call and nothing else.
- Do not add explanation, markdown, role labels, or natural language
    around the tool call.
- search format: {"name":"search_products","arguments":{"query":"...
    "}}
- item-details format: {"name":"get_item_details","arguments":{"
    item_id":"B0XXXXXXXX"}}
- user-preference format: {"name":"get_user_preference","arguments"
    :{}}

2. Shopper-facing reply mode
- Use this mode only after you have enough retrieved evidence to
    answer.
- Write plain English, including the required field labels below.
- Never expose tool names, JSON, code, commands, role tokens, XML
    tags, or hidden reasoning.
- Never end with an unfinished sentence, ellipsis, or placeholder.

3. Clarification/dialogue mode
- Use this mode only when the shopper's request lacks a decisive
    constraint and no reasonable search query can be formed.
- Ask exactly one short question and nothing else.
- Do not emit a tool call, recommendation fields, JSON, or internal
    reasoning in this mode.
- After the shopper answers, return to tool-call mode and search
    with the clarified constraint.

Recommendation output contract:
- Recommend exactly one item unless the shopper explicitly asks for
    alternatives.
- Every recommendation must use exactly this field structure:
  recommended_item_id: <ITEM_ID>
  recommended_item_description: <2-4 complete English sentences>
- The description is the most important field. It must explain why
    this item matches the shopper's requested attributes.
- The description should briefly explain why the recommended item
  matches the shopper's stated constraints and preferences.
- Mention only user-relevant attributes supported by retrieved
    evidence.
- Do not add unsupported attributes or keyword lists.
- Mention only claims supported by retrieved evidence.
- Any shopper-facing answer after tool use must still use the
    recommendation field structure; do not answer follow-up
    questions as free-form prose.

Decision and tool policy:
- Build search queries from the current request's product type and
    discriminative constraints. Keep them concise; use at most {N}
    searches total.
- Use get_item_details selectively to verify plausible candidates
    and hard constraints; do not inspect the same item twice.
- Treat get_user_preference as optional ranking evidence, never as a
    required first step. Use it at most once and only when one
    missing soft preference could change a specific search term or
    break a tie among relevant candidates.
- Useful cases are broad or taste-driven requests, unresolved soft
    trade-offs, or several similarly relevant candidates. Skip it
    when explicit attributes already determine the search or when
    search/item details can resolve the uncertainty.
```

```
- When top search results are similar, or shopper feedback adds
    specific attributes such as brand, model, size, edition,
    material, compatibility, or included accessories, inspect one
    plausible candidate with get_item_details before recommending.
- Recommend only when retrieved evidence supports the shopper's
    requested product type and attributes.
- After shopper feedback, do not answer in prose; search with the
    corrected constraints, inspect useful evidence if needed, then
    recommend using the required fields.
- Never repeat a previous search query. If no meaningfully different
     query is possible or the tool budget is nearly exhausted,
    recommend the best grounded retrieved item.
- Ask one short clarification question only when no reasonable
    search query can be formed.
```

C.2 User Simulator Prompt

C.2.1 User Simulator: Initial Shopping Request

Initial Shopping Request Prompt

```
You are simulating the shopper in a recommendation dialogue. State
    your shopping need clearly in natural English. Keep it concise,
    realistic, and specific enough for a recommender to search.
    Never output, quote, or reveal any product or item ID, even if
    it appears in the context.
Reference query: {reference_query}
```

C.2.2 User Simulator: Conversational Feedback

Conversational Feedback Prompt

```
You are simulating the shopper in a recommendation dialogue. Stay in
     character as a shopper with a clear product need. Continue the
    conversation naturally in plain English. If the assistant asks a
     question, answer it directly. Give corrective feedback with one
     or two concrete missing requirements so the recommender can
    search again. Do not use rigid labels or templates; sound like a
     normal shopper. Do not output JSON, role labels, tags, hidden
    metadata, or any product/item ID. Never reveal, quote, or repeat
     an Item ID, even if it appears in the context.
```

C.2.3 User Input Template

User Input Template

```
Your shopping need is based on:
- Reference query: {reference_query}
Assistant recommended item Ids: {recommended_item_ids}
```

```
Recommendation was grounded in recent tool results: {grounded}

Recent dialogue:
{dialogue}

Latest assistant message:
{assistant_text}

Write the next shopper reply only.
```

C.3 Listwise Product Reranking

C.3.1 System Prompt

Listwise Product Reranking

```
You are a helpful AI assistant. You first think about the reasoning
    process in the mind and then provide the user with the answer.
```

C.3.2 User Input Template

User Input Template

```
You are an expert in product reranking. Given a customer query and a
     list of candidate products, rerank the products so the most
    relevant items are placed at the top. Consider semantic meaning,
     product attributes, and customer intent.

Below are the candidate products (item_id and metadata):
```json
{candidate_block}
```

Below is the query:
```text
{query}
```

Show your work in <think> tags. Your final response must be in JSON
    format within <answer> </answer> tags. The JSON must contain the
     reranked list of item_ids in order of relevance under the key "
    reranked_items". The length of the reranked list should match
    the number of input candidates.
```

C.4 User Preference Profile Construction

C.4.1 Prompt Template

We strictly remove the target item of the current task from the user history before preference summarization, preventing the preference profile from directly exposing the ground-truth recommendation.

User Preference Profile Construction Prompt

```
Based on the user's purchase history below, generate a concise user
    preference profile.

## User Purchase History (chronological order):
{purchase_history}

## Statistics:
- Average rating: {avg_rating}

## Requirements:
1. Summarize the user's likely purchase preferences and favorite
    product types.
2. If low-rated items clearly indicate dislikes, mention that
    briefly.
3. Keep the profile to exactly one short natural sentence of 20--30
    words.
4. Output ONLY in the <answer> tag, no other explanatory text.
5. Do not include unsupported details, user IDs, bullet points, or
    multiple sentences.

## Output Format:
<answer>
The user likely prefers practical, durable products in their
    favorite categories and avoids items with clear quality or
    usability issues.
</answer>
```