Think Wise, Collaborate Effectively: A Rationale-Aware LLM-Based Recommender with Reinforcement Learning from Collaborative Signals

Chung Park[1], Taesan Kim[1], Hyeongjun Yun[1, 2], Dongjoon Hong[1], Junui Hong[1], Kijung Park[1], MinCheol Cho[1], Min sung Choi[1], Jihwan Seok[1], Jaegul Choo[2]

[1]SK Telecom

[2]Korea Advanced Institute of Science and Technology

{skt.cpark, ktmountain, hjyoon, dongjoon.hong, skt.juhong, kijung.park, skt.mccho, ms.choi, seokjh}@sk.com

cpark88kr@gmail.com , jchoo@kaist.ac.kr

Abstract

Large Language Models (LLMs) have recently emerged as powerful reasoning engines in recommender systems, generating natural-language explanations that foster user engagement. However, their recommendation performance remains limited, as they lack exposure to collaborative user-item interaction patterns. In contrast, collaborative filtering (CF) models achieve strong performance by learning from these behavioral patterns at scale. To unify the strengths of **both paradigms**, we propose **TWiCE-Rec** (Think Wise, Collaborate Effectively), a rationale-aware LLM-based recommender that incorporates collaborative user-item interactions. In the first stage, we construct a rationale dataset by applying in-context learning with self-annotated curation. A state-of-the-art LLM is guided to generate persuasive rationales that explain the causal relationship between the user's interaction sequence and the ground-truth next item, resulting in a curated post-hoc training dataset. In the second stage, we perform multi-task instruction-tuned adaptation—based on the rationale-augmented training dataset—comprising item description generation and both non-reasoning and reasoning-based sequential recommendation, to equip the LLM with the ability to generate rationales that reflect how user preferences align with item characteristics. Finally, we aim to enhance the LLM's recommendation performance by incorporating user-item interaction patterns derived from the CF-Rec model. To achieve this, we propose a confidence-weighted reinforcement learning strategy that adjusts rewards in proportion to both the LLM's prediction alignment with the ground-truth and the confidence from the pretrained CF-Rec model. Our method outperforms both CF- and LLM-Rec models on Amazon datasets in terms of recommendation performance and rationale quality. In an online A/B test, it achieved about 8% higher click-through rate than existing models, demonstrating practical value.

Code — github.com/cpark88/TWiCE-Rec

Introduction

Large Language Models (LLMs) introduce a new paradigm for recommender systems by enabling natural-language reasoning, allowing them to both recommend items and explain why those recommendations are appropriate. Such rationales are not only useful for enhancing user engagement and satisfaction, but are also increasingly adopted in real-world applications—e.g., platforms like *Amazon* and *Instagram* often display explanations for recommendations to users—highlighting their practical business value.

However, a key limitation of LLM-based recommenders (LLM-Rec) is their lack of exposure to collaborative user-item interaction patterns, which are essential for accurate recommendation performance in real-world applications. As a result, while LLM-Rec can generate fluent and plausible explanations, their recommendation performance tends to fall behind that of collaborative filtering-based recommenders (CF-Rec), as shown in Fig. 1. In contrast, CF-Rec demonstrates strong recommendation performance by modeling user preferences derived from user–item co-occurrence patterns (Kang and McAuley 2018). However, these models are inherently black-box and incapable of producing human-understandable rationales, limiting their utility in applications requiring explainability.

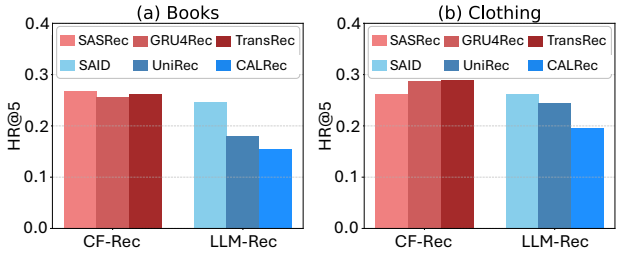

Figure 1: Comparisons between the performance of CF-Rec and LLM-Rec, under the *Books* and *Clothing* domain on Amazon Review dataset (i.e., CF-Rec>LLM-Rec).

To bridge this gap, we propose **TWiCE-Rec** (Think Wise, Collaborate Effectively), a rationale-aware LLM-based recommender that integrates LLM reasoning with collaborative signals via reinforcement learning. The name reflects our goal to unify the **two paradigms**—LLM-Rec and CF-Rec—by combining their respective strengths and effectively anchoring LLM reasoning in collaborative user-item behaviors. Our model is structured around three key stages: (1) *Self-Annotated Rationale Construction*, (2) *Rationale-Aware Knowledge Alignment*, and (3) *Collaborative Preference Alignment*. In the first stage, we adopt in-context learn-

ing with self-annotated curation to guide a state-of-the-art LLM in generating rationales grounded in the logical causality between the user behavior and the ground-truth next item. To ensure the quality of the dataset, we develop a curation process in which the LLM autonomously evaluates and filters out instances with insufficient causal relevance or weak logical coherence. Building on this process, we inject persuasive rationales into the training phase to provide richer supervision signals and enhance the reasoning capabilities of the LLM. During the second stage, rationale-aware knowledge alignment, we aim to train the model to generate coherent and persuasive rationales by aligning user interaction sequences with the semantics of recommended items. To this end, we adopt a multi-task instruction tuning strategy consisting of three tasks: item description generation, non-reasoning sequential recommendation, and reasoning-based sequential recommendation. Note that the rationale dataset constructed in Stage 1 is used to train the reasoning-based sequential recommendation task, which serves as the primary objective of this stage. The other two tasks provide the LLM with foundational knowledge of user behavior and item characteristics.

Finally, we proceed to the collaborative preference alignment stage, where the goal is to align the recommended items with collaborative user preferences. To this end, we propose a reinforcement learning with verifiable rewards framework guided by collaborative signals, where the reward strategy leverages confidence scores from a pretrained CF-Rec model. The reward is amplified when the LLM-generated recommendation both matches the ground-truth item and receives high confidence from the CF-Rec model. If the recommendation is incorrect but receives high confidence from the CF-Rec model, we assign a reward in proportion to this confidence, as such cases may represent plausible alternatives aligned with collaborative patterns. We optimize the model using a relative reward strategy based on group-wise ranking, which enables efficient and scalable reinforcement learning (Shao et al. 2024). We conducted large-scale experiments across eight Amazon Review domains, comparing our method against approximately 20 state-of-the-art CF- and LLM-Rec baselines. Our method consistently outperformed most baselines in both recommendation performance and rationale quality, as measured by multiple evaluation metrics. Furthermore, in a real-world online A/B test, our model outperformed existing models with an 8% relative improvement in click-through rate, demonstrating its practical effectiveness.

Related Work

Collaborative Filtering-based Recommendation

Traditional CF-Rec models have long served as the foundation for recommendation systems, demonstrating robust performance. For example, SASRec (Kang and McAuley 2018) employs Transformer-based self-attention mechanisms to model user-item interaction sequences for sequential recommendation. Additionally, models such as NextItNet (Yuan et al. 2019), which applies convolutional neural networks, and TransRec (He, Kang, and McAuley 2017), based on Markov chains, have explored different paradigms to capture the dynamics of user-item relationships.

LLM-based Recommendation

Recent studies have explored the use of large language models (LLMs) as reasoning engines in recommendation systems. For instance, RecExplainer (Lei et al. 2024) fine-tunes LLMs in a supervised manner using user behaviors and intentions, enabling the generation of personalized recommendation rationales. Rec-R1 (Lin, Wang, and Qian 2025) directly applies group relative policy optimization (GRPO), where the reward is defined using evaluation metrics such as Mean Reciprocal Rank (MRR). However, most existing approaches fall short of surpassing traditional CF-Rec models in recommendation performance (See Appendix B).

Methodology

Our overall three-stage procedure is illustrated in Fig. 2 and Algorithm 1 (Appendix C).

Stage 1: Self-Annotated Rationale Construction

User Sequence We assume a sequential recommendation task setting where a user $u \in \mathcal{U}$ interacts with a sequence of items $S_u = [i_1, i_2, \ldots, i_T]$, where each $i_t \in \mathcal{I}$ is an item from the whole item set $\mathcal{I}$. Each item i_t is represented by a tuple containing the item's **title**, **description**, as well as the user's **review** and **rating** for that item. Then, we set the next item $i^+ \in \mathcal{I}$ that the user interacted with following the historical interaction sequence S_u. Based on this formulation, we construct an initial training dataset $\mathcal{D} = \{(S_u, i^+)\}$, where S_u denotes the interaction sequence of user u, and i^+ is the corresponding ground-truth next item.

Item Meta We assume the availability of item-level metadata m_i for each item i, encompassing attributes such as title, description, and detailed specifications. Note that the item-level metadata does not contain user-specific information such as reviews or ratings, which are present in the user interaction sequences S_u.

In-Context Learning with Self-Annotated Curation for Rationale Generation We constructed a rationale dataset via in-context learning, using a state-of-the-art LLM guided by carefully designed prompts (Refer to Appendix F). To this end, we constructed prompts using S_u and i^+ to guide the model in inferring plausible rationales for the user's interest in item i^+, grounded in their historical interaction sequence S_u. We employed a few-shot setting, seeding each prompt with exemplar rationales, and instructed an LLM to not only generate post-hoc rationales r_{i^+} but also assess the logical causality between the user's inferred preferences—derived from their historical interactions—and the attributes of the ground-truth item (Gu et al. 2025). Irrelevant rationales, as tagged by the LLM when no clear connection was found, were filtered out to ensure quality (*Self-Annotated Rationale Curation*), with results aligning well with human judgment. These post-hoc rationales were then combined with the corresponding S_u and i^+ to construct

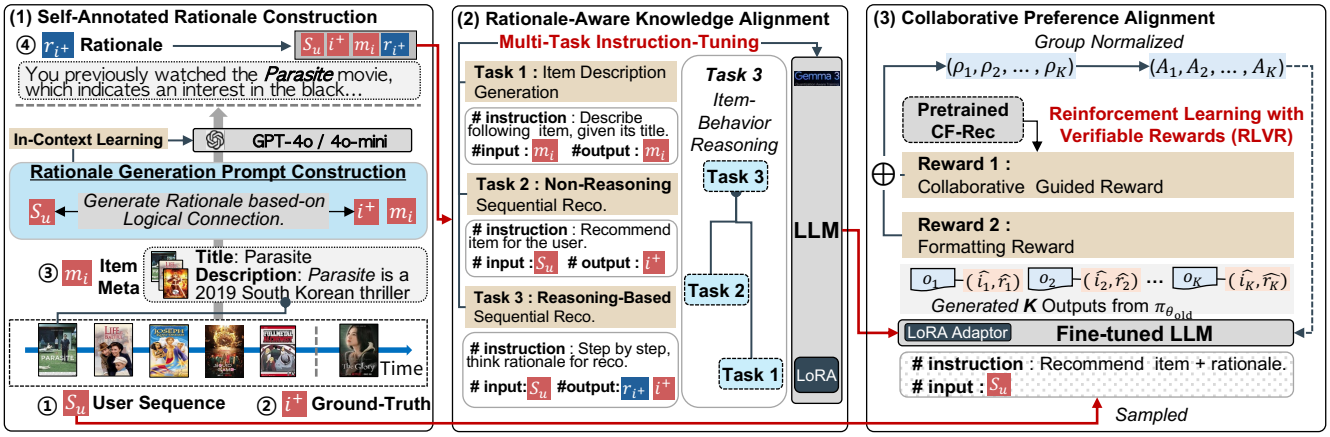

Figure 2: Overview of our training framework.

the final training data, consisting of tuples (S_u, i^+, r_{i^+}). Incorporating well-crafted rationales into the training data enriches the supervision signal, thereby enhancing both the interpretability of the model and the efficiency of train-time scaling. See Appendix I for annotation protocol details.

Problem Formulation Given a user u, their historical interaction sequence S_u, the model is expected to generate a tuple $(\hat{i}, \hat{r})$, where $\hat{i}$ is the predicted next item and $\hat{r}$ is a natural-language rationale for why $\hat{i}$ is a suitable recommendation. The following stages are designed to accurately predict the next item $\hat{i}$ and generate a plausible rationale $\hat{r}$.

Stage 2: Rationale-Aware Knowledge Alignment

In this stage, we aim to train the model to generate rationales that are both persuasive and logically consistent, based on item semantics and user preferences reflected in behavioral patterns and reviews. We jointly instruction-tune the model on three tasks (1) *item description generation*, (2) *non-reasoning*, and (3) *reasoning-based sequential recommendation*. While all tasks contribute to enhancing the model's reasoning ability, the **reasoning-based sequential recommendation task**—trained on the rationale dataset from Stage 1—is our primary objective. The other two serve as foundational tasks to improve the model's understanding of user behavior and item semantics.

Task 1: Item Description Generation We introduce an item description task, where the model generates descriptions from the title in the item metadata m_i. This helps the model internalize fine-grained item semantics, laying the groundwork for generating persuasive rationales.

Task 2: Non-Reasoning Sequential Recommendation The goal of this task is to predict the next item in a user's interaction sequence. In our prompt design with (S_u, i^+), we instructed the model to wrap the predicted item within `<item>` and `</item>` tags, so that the output can be easily parsed and utilized in real-world production systems. The model is trained to autoregressively predict the correct item i^+ given the user sequence S_u. This task allows the model to learn item co-occurrence common in user behavior data. Prompts for tasks 1 and 2 are shown in Appendix F.

Task 3: Reasoning-Based Sequential Recommendation In this task, the model is prompted not only to predict the next item, but also to generate a natural-language rationale explaining the recommendation. Through task 1 and 2, the model learns item-level knowledge and user interests grounded in historical interactions, respectively. Building upon this foundation, task 3 aims to capture the underlying causal relationships between item characteristics and user preferences. Using the previously constructed dataset of tuples (S_u, i^+, r_{i^+}), we created instruction data for this task by prompting the model to generate both the rationale r_{i^+} and the next item i^+, given the user history S_u. We instructed the model to follow a rationale-first decoding process: first infer user preferences from historical interaction sequences and reviews, then generate a rationale before recommending the next item—effectively injecting a chain-of-thought (CoT) structure to guide its reasoning (Fig. 3). To ensure compatibility with downstream applications, we also trained the model to follow a specific output format: the rationale is enclosed within `<think>` `</think>` tags, and the recommended item within `<item>` `</item>` tags.

#Instruction

Based on **[Purchase History]**, use your logical reasoning process to identify the most suitable item for this customer from the **[Candidate List]**. Then, include your reasoning inside the `<think></think>`tag and the recommended item inside the `<item></item>`tag.

#Input

[Purchase History] (1)Title: Cetaphil baby mommy and me travel kit (2)Title: Summer infant 2-in-1 carry & cover infant car seat cover,...(truncated) $\leftarrow S_u$

[Candidate List]: (omitted for brevity)

#Output

`<think>`The customer has previously purchased items that are primarily focused on baby care and comfort,... (truncated) `</think>`$\leftarrow r_{i^+}$

`<item>`BB my best buddy toddler kids pillowcases `</item>`$\leftarrow i^+$

Figure 3: An example prompt for Task 3

Stage 3: Collaborative Preference Alignment

Building on Stage 2, we applied reinforcement learning (RL) to make further adjustments to align the LLM with accurate and collaborative recommendation behavior. Although instruction-tuning enables the model to produce semantically relevant recommendations, the lack of alignment with collaborative preferences often hinders generalization and degrades recommendation performance (Lin, Wang, and Qian 2025). To address this, we argue that aligning the model with collaborative signals and ground-truth behaviors is key to improving recommendation performance.

Reinforcement Learning with Verifiable Rewards (RLVR) Reinforcement learning allows the model to continuously adapt its behavior to maximize performance on the downstream recommendation task (Lin, Wang, and Qian 2025). While supervised fine-tuning directly teaches the model to match ground-truth answers, it often fails to induce reasoning skills required for complex tasks. In contrast, Reinforcement Learning with Verifiable Rewards (RLVR) enables the model to develop a chain-of-thought by learning solely from reward-based feedback through repeated trial and error (Shao et al. 2024). Following DeepSeek-R1 (Guo et al. 2025), we implement RLVR using Group Relative Policy Optimization (GRPO) with a novel reward design, to further optimize the LLM from Stage 2. Compared to traditional reinforcement algorithms such as Proximal Policy Optimization (PPO) (Schulman et al. 2017), GRPO significantly reduces memory consumption during training while maintaining competitive performance.

For each prompt q constructed with S_u, GRPO samples a group of outputs $\{o_1, o_2, \ldots, o_K\}$ from the old LLM policy $\pi_{\theta_{\text{old}}}$. The prompt q is designed to generate both a rationale and a recommended item based on the user sequence S_u. Therefore, the k-th generated output o_k consists of the rationale $\hat{r_k}$ and the recommended item $\hat{i_k}$. Then, it optimizes the LLM by maximizing the following objective:

$$\begin{aligned}&\mathcal{J}_{\text{GRPO}}(\theta) = \mathbb{E}_{q\sim P(Q),\{o_k\}_{k=1}^{K}\sim\pi_{\theta_{\text{old}}}(O|q)} \\ &= \frac{1}{K}\sum_{k=1}^{K}\left\{\min\left[\frac{\pi_\theta(o_k|q)}{\pi_{\theta_{\text{old}}}(o_k|q)}A_k, \text{clip}\left(\frac{\pi_\theta(o_k|q)}{\pi_{\theta_{\text{old}}}(o_k|q)}, 1-\varepsilon, 1+\varepsilon\right)A_k\right]\right\},\end{aligned} \tag{1}$$

where ε is hyper-parameter, and A_i represents the advantage computed using relative rewards within each group. To compute A_k, we first apply the reward function to a set of group outputs $\{o_1, o_2, \ldots, o_K\}$, obtaining a corresponding set of reward scores $\boldsymbol{\rho} = \{\rho_1, \rho_2, \ldots, \rho_K\}$. These scores are then standardized by subtracting their mean and dividing by their standard deviation. This outcome-based supervision yields normalized advantages for each output as follows: $A_k = \frac{\rho_k - \text{mean}(\boldsymbol{\rho})}{\text{std}(\boldsymbol{\rho})}$. The reward is the source of the training signal, which decides the optimization direction of RL. We train our model using two custom-designed reward functions: (1) Collaborative-Guided Reward and (2) Formatting Reward.

(A) Collaborative Guided Reward To combine the strengths of ground-truth correctness and collaborative preference alignment, we define a **composite reward function** for each generated output $\hat{o_k} = (\hat{r_k}, \hat{i_k})$ from the LLM. This function incorporates two distinct reward signals—(1) the binary item matching reward $\rho^{\text{match}} \in \{0, 1\}$ and (2) the collaborative filtering reward $\rho^{\text{cf}} \in [0, 1]$.

(A-1) Item Matching Reward (ρ^{match}) : The k-th output o_k of the LLM includes the rationale $\hat{r_k}$ and the recommended item $\hat{i_k}$. A binary reward indicating whether the generated item $\hat{i_k}$ matches the ground-truth item i^+. This binary reward serves as a precise signal to anchor the model's behavior to actual user preferences, i.e., $\mathbb{I}[\hat{i_k} = i^+]$. It provides a stable target signal that complements more flexible reward components. However, using only item-matching reward provides sparse feedback, as exact item matches are rare, limiting the richness of learning signals necessary for stable policy optimization. Therefore, we introduce the following sub-reward to address this issue.

(A-2) Collaborative Filtering Reward (ρ^{cf}) : This reward measures how confidently a pretrained CF-Rec (e.g., SASRec) supports the LLM-generated item. This encourages the LLM to align its outputs with collaborative patterns, enabling it to learn CF-style reasoning through soft supervision. To compute this reward, we use a pretrained and frozen collaborative filtering model f_ϕ, which estimates user-item compatibility based on sequential interactions. Given a user's interaction sequence S_u and a recommended item $\hat{i_k}$, we compute their similarity $z_{\hat{i_k}} = f_\phi(S_u, \hat{i_k})$, where f_ϕ measures the similarity between their corresponding embeddings. We then apply a sigmoid transformation to obtain a normalized reward: $\rho^{\text{cf}} = \frac{1}{1+\exp(-z_{\hat{i_k}})}$.

Finally, this reward considers ρ^{match} and ρ^{cf}, and two hyperparameters: amplification factor λ and threshold τ. The reward of k-th output is computed as follows:

$$\rho_k^{cg} = \begin{cases} 1 + \lambda \cdot \rho_k^{\text{cf}} & \text{if } \rho_k^{\text{match}} = 1 \text{ and } \rho_k^{\text{cf}} \geq \tau \\ 1 & \text{if } \rho_k^{\text{match}} = 1 \text{ and } \rho_k^{\text{cf}} < \tau \\ \rho_k^{\text{cf}} & \text{if } \rho_k^{\text{match}} = 0 \text{ and } \rho_k^{\text{cf}} \geq \tau \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

This reward selectively amplifies the confidence of the pretrained CF-Rec only when the LLM-generated item matches the ground-truth and is assigned high confidence by the CF model (i.e., $\rho^{\text{cf}} \geq \tau$). This design ensures that collaborative signals are emphasized only when they reflect correct user behavior. Conversely, if the LLM generates an incorrect item that still receives high confidence from the CF-Rec (i.e., $\rho^{\text{match}} = 0$ but $\rho^{\text{cf}} \geq \tau$), it is assigned only ρ^{cf}, encouraging the model to learn from plausible but imperfect predictions rather than reinforcing errors. This reward design incentivizes the model to generate items that are not only behaviorally plausible according to collaborative patterns, but also grounded in true user preferences, thereby aligning the generation with both accuracy and collaborative consistency.

(B) Formatting Reward This reward is defined as the sum of two sub-rewards described below, to guide the LLM to produce outputs that conform to the expected format.

(B-1) Structural Correctness Reward (ρ^{struct}) : This binary reward checks whether both the generated item and rationale in the k-th output o_k appear in the correct format:

$$\rho_k^{\text{struct}} = \mathbb{I}[\exists \texttt{<item>} \cdot \text{recommend item} \cdot \texttt{</item>} \wedge \exists \texttt{<think>} \cdot \text{rationale} \cdot \texttt{</think>} \in o_k]. \tag{3}$$

To handle repeated tags despite correct structural formatting, an additional sub-reward is introduced as follows.

(B-2) Tag Presence Reward (ρ^{tag}) : This sub-reward checks whether each of the four required tags—<item>, </item>, <think>, and </think>—is present in the k-th generated output o_k:

$$\rho_k^{\text{tag}} = 0.25 \cdot \mathbb{I}[\texttt{<item>} \in o_k] + 0.25 \cdot \mathbb{I}[\texttt{</item>} \in o_k] + 0.25 \cdot \mathbb{I}[\texttt{<think>} \in o_k] + 0.25 \cdot \mathbb{I}[\texttt{</think>} \in o_k]. \tag{4}$$

The final formatting reward is computed as the sum:

$$\rho_k^{\text{fmt}} = \rho_k^{\text{struct}} + \rho_k^{\text{tag}} \tag{5}$$

(C) Final Reward We compute the final reward ρ_k for the k-th generated output o_k in the GRPO algorithm by aggregating ρ_k^{cg} and ρ_k^{fmt}. Subsequently, in the GRPO framework, we normalize the final rewards $\boldsymbol{\rho} = \{\rho_1, \rho_2, \ldots, \rho_K\}$ generated by the LLM policy to compute the corresponding advantages $\mathbf{A} = \{A_1, A_2, \ldots, A_K\}$. These advantages are then used to update the LLM policy according to Eq. 1.

Gradient Analysis We conducted the gradient analysis from the perspective of the Collaborative Guided Reward. The gradients of our GRPO objective (Eq. 1) can be succinctly described as follows:

$$\nabla_\theta \mathcal{J}_{\text{GRPO}} = \frac{1}{K} \sum_{k=1}^{K} \left\{ \text{clip}\left(\frac{\pi_\theta(o_k|q)}{\pi_{\theta_{\text{old}}}(o_k|q)}, 1-\varepsilon, 1+\varepsilon \right) A_k \cdot \nabla_\theta \log \pi_\theta(o_k \mid q) \right\}. \tag{6}$$

The gradient term $\nabla_\theta \log \pi_\theta(o_k \mid q)$ increases the likelihood of the output o_k, and the sign of A_k determines whether this likelihood is increased or decreased. Specifically, if $A_k > 0$, the gradient update increases $\pi_\theta(o_k \mid q)$; if $A_k < 0$, it decreases it. Outputs that align well with collaborative filtering (CF) signals tend to have higher ρ_k, resulting in larger positive A_k and stronger gradient updates. To further promote this alignment, we scale the rewards within ρ_k using $1 + \lambda \cdot \rho_k^{\text{cf}}$, where λ controls the influence of CF-based confidence.

Experiments

Our experiments aim to answer the following questions:
(RQ1): Does our model outperform existing CF- and LLM-Rec models in terms of recommendation performance?
(RQ2): What is the quality of the recommendation rationales generated by our model?
(RQ3): Does each training stage improve recommendation performance?
(RQ4): What is the optimal confidence weight λ of the collaborative signal in the reward function of Stage 3?
(RQ5): Are the online performance and latency of our model within a reasonable and deployable range?
(RQ6): Does the use of rationales during both training and inference improve the recommendation performance?

Datasets

We experimented with eight domains in **Amazon Review Dataset 2023** (Hou et al. 2024), and divided them into two types based on the average number of purchases per item:
Type A (High-frequency, demand-driven) includes *Grocery and Gourmet Food*, *Industrial and Scientific*, *Musical Instruments*, and *Office Products*, which exhibit high repeat purchase rates and reflect task-specific consumption.
Type B (Low-frequency, preference-driven) consists of *Amazon Fashion*, *Baby Products*, *Clothing Shoes and Jewelry*, and *Health and Household*, where consumption tends to be selective, taste-oriented, or life-stage dependent. Detailed data statistics are provided in Appendix A.

Experimental Setup

Baselines and Evaluation Settings We assessed the performance of our model by comparing it to two categories of baselines: (1) CF-based (**CF-Rec**) and (2) LLM-based Recommendation (**LLM-Rec**) as shown in Table 1. We adopted `Gemma-3-1B-it` (Team et al. 2025) as the LLM backbone, taking into account deployment constraints in real-world environments, and observed a clear scale-law trend (e.g., 1B, 4B, 12B) in larger models (Appendix J). We evaluated recommendation performance using the *leave-one-out* method. For each user sequence, the last item was used for testing, the second-to-last for validation, and the rest for training. We adopted a common approach by pairing the ground-truth item (positive sample) with negative samples the user has not interacted with. These items form the candidate set and are incorporated into the prompt construction for inference. We focused on standard evaluation metrics, including *Hit Ratio* (**HR**) and *Normalized Discounted Cumulative Gain* (**NDCG**). The implementation details are described in Appendix A.

Performance Evaluation (RQ1)

(1) Our model showed overall improved recommendation performance for most domains in both datasets compared to the state-of-the-art CF-Rec and LLM-Rec models (Table 1). Our model consistently outperformed all baselines in terms of HR@1 across eight domains. Specificially, our model demonstrated significant improvements over the best-performing baseline (second-best for each domain) in HR@1, achieving gains of **+63.31%** (*Fashion*), **+13.87%** (*Baby Products*), **+32.31%** (*Grocery*), **+1.97%** (*Scientific*), **+2.92%** (*Clothing*), **+38.94%** (*Musical Instruments*), **+4.62%** (*Health*), and **+5.84%** (*Office Products*). Although our model is not always the best in HR@5 and NDCG@5, its performance remains within 1%p of the top models in most domains.
(2) Our model demonstrates consistently strong performance across both Type A —characterized by frequent

Domain		Fashion			Baby Products			Grocery			Scientific			Clothing			Musical Inst.			Health			Office Products		
Types	**Models**	H@1	H@5	N@5	H@1	H@5	N@5	H@1	H@5	N@5	H@1	H@5	N@5	H@1	H@5	N@5	H@1	H@5	N@5	H@1	H@5	N@5	H@1	H@5	N@5
CF Rec	SASRec	2.70	14.40	8.43	2.56	12.91	7.61	2.88	14.75	8.72	2.49	13.58	7.86	3.16	13.53	8.24	2.82	14.27	8.40	2.52	13.69	7.99	2.92	13.76	8.19
	BERT4Rec	1.41	6.67	3.98	2.41	12.39	7.28	2.78	14.29	8.40	2.44	12.98	7.56	3.16	12.45	7.73	2.92	14.08	8.34	2.41	12.87	7.52	3.07	13.21	8.04
	S3Rec	1.58	9.16	5.22	2.31	12.67	7.34	2.62	14.10	8.22	3.75	17.04	10.24	2.62	11.67	7.09	3.15	13.90	8.39	2.54	12.25	7.29	2.77	12.63	7.60
	NextItNet	1.85	6.21	4.06	2.52	13.05	7.66	3.16	14.59	8.73	2.50	13.73	7.97	3.09	13.91	8.39	2.88	14.20	8.43	2.57	13.59	7.96	3.15	13.45	8.22
	SINE	2.39	10.42	6.38	2.58	13.03	7.67	3.07	14.59	8.70	2.67	14.00	8.18	4.45	**18.48**	8.43	2.64	14.86	8.65	2.81	**14.45**	**8.48**	2.93	13.51	8.08
	DFM	2.58	10.40	6.43	2.66	**13.07**	**7.70**	2.88	15.00	8.79	2.56	13.85	8.06	3.33	17.71	**10.25**	3.21	14.79	8.86	2.58	13.57	7.90	3.04	13.57	8.21
	TransRec	1.59	7.92	4.65	2.35	12.92	7.50	3.24	**15.25**	**9.12**	3.65	16.33	9.11	3.29	13.97	8.52	3.17	**14.91**	8.92	2.56	13.80	8.02	2.96	14.27	8.50
	LightSANs	3.38	15.92	9.47	2.55	12.91	7.60	3.08	14.61	8.75	2.49	13.66	7.93	2.83	13.67	8.12	2.89	14.45	8.50	2.65	13.76	8.06	3.03	13.53	8.15
	FOSSIL	1.93	6.13	4.06	2.52	12.74	7.46	3.04	14.68	8.75	2.58	13.49	7.89	3.38	13.98	8.57	3.04	14.56	8.70	2.70	13.34	7.89	2.95	13.22	7.97
LLM Rec	SAID	2.84	6.18	4.39	0.55	2.30	1.35	0.94	3.97	2.35	1.13	8.25	4.68	1.32	6.22	3.67	1.05	4.99	3.00	2.18	4.80	3.47	0.96	3.64	2.19
	InstructRec	1.45	3.03	2.35	2.01	3.79	3.03	1.39	3.64	2.64	2.69	7.11	5.08	1.35	3.68	2.65	1.55	4.11	2.95	1.52	4.29	3.02	1.92	4.46	3.35
	Intuitor	2.67	7.47	5.28	2.34	4.12	3.36	2.04	3.39	2.83	0.96	1.21	1.11	2.69	4.35	3.65	2.25	3.45	2.97	1.32	1.74	1.58	1.16	1.60	1.43
	STREAM-Rec	2.70	14.39	8.56	1.32	11.35	6.13	2.96	14.64	8.74	3.72	**23.94**	13.68	2.06	10.90	6.45	2.65	12.72	7.66	3.90	12.04	8.27	2.76	15.44	8.97
	S-DPO	2.93	15.01	9.02	2.60	9.69	6.16	2.59	11.23	6.91	3.95	22.05	13.24	2.25	9.91	6.09	2.37	11.48	6.92	3.75	16.32	9.93	2.59	14.54	8.47
	CALRec	1.47	6.79	3.89	0.20	4.18	2.16	0.76	5.44	2.86	0.93	15.92	9.29	0.66	3.36	2.12	1.00	3.17	2.00	0.98	4.40	2.54	0.20	5.54	2.58
	RosePO	3.05	14.72	8.60	2.56	8.47	5.63	1.95	8.57	5.07	5.59	7.17	6.55	2.27	5.88	4.16	2.75	10.38	6.61	2.52	10.27	6.42	2.96	11.58	7.33
	Rec-R1	1.91	5.72	3.95	2.74	7.28	5.25	1.44	4.91	3.23	4.11	9.82	7.18	2.83	8.35	5.72	2.13	5.59	3.96	2.22	7.49	5.01	2.53	8.79	5.81
	Rec-SAVER	3.35	14.07	8.72	2.09	12.14	7.12	3.25	13.32	8.19	2.95	22.27	12.84	2.51	11.49	6.96	2.47	10.88	6.66	2.99	11.59	7.38	5.14	16.11	10.78
	Ours	**5.52**	**16.10**	**11.10**	**3.12**	12.27	7.54	**4.30**	14.02	8.81	**5.70**	23.16	**14.35**	**4.58**	11.44	8.25	**4.46**	13.95	**9.51**	**4.08**	12.16	8.37	**5.44**	**16.44**	**11.07**

Table 1: Model performance comparison on ***Amazon***, with the top two methods highlighted in **bold** and **underline**, respectively.

and consumable item usage—and Type B —centered on one-off consumption. This indicates that the model effectively incorporates both the semantic understanding capabilities of LLMs (Type B), which capture user-specific preferences, and the collaborative signals derived from shared behavioral patterns among users (Type A).

Discussion of Rationales Quality (RQ2)

We quantitatively assessed the quality of recommendation rationales generated by our model using the same prompt across `Gemma-3-1B`, `Gemma-3-12B`, Rec-SAVER, and Rec-R1. For a comprehensive evaluation, we sampled 300 instances from each of the eight domains, resulting in a total of 2,400 samples used for extensive assessment. The evaluation criteria are formulated with a two-step approach: initially assessing correctness of the recommendation, followed by the assessment of rationale quality. Correctness is defined by the inclusion of the recommended item in the item vocabulary set, a requirement essential for detecting hallucinated outputs. Inspired by Wang et al. (2022), ChatGPT evaluated the outputs for a four-level scoring system, i.e., **0**: Incorrect, **1**: Correct and Insufficient Rationale, **2**: Correct and Acceptable Rationale, **3**: Correct and Satisfying Rationale. As shown in Fig. 4, our model generated convincing rationales that not only support the correct recommendation but also make the reasoning behind it persuasive. On the full set of 2,400 samples across all domains, our model achieved the highest proportion of top-quality rationales (score 3) at 75.71%, outperforming the second-best Rec-SAVER by approximately 4%p. The specific prompt used for this test and actual examples are provided in the Appendix F, along with additional results from other domains in the Appendix E.

Discussion of Model Variants (RQ3)

In this section, we evaluate the efficacy of each training stage by testing five primary model variants:

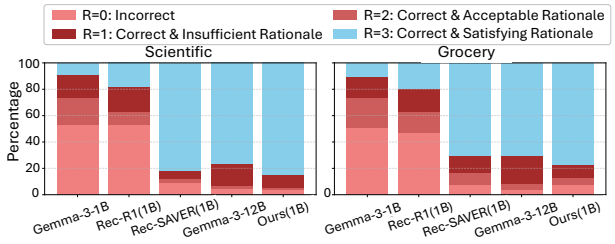

Figure 4: GPT-4o Evaluation of Rationale Quality

(A) Base: The base `Gemma-3-1B-it` model.
(B) +Stage2: The base model fine-tuned with only Stage 2.
(C) +Stage3: The base model fine-tuned with only Stage 3.
(D) +Stage2 + GRPO (Item Match + Formatting): The model was trained with GRPO, using only item matching and formatting rewards, **excluding the collaborative-guided reward**.
(E) +Stage2 + Stage3: Our full proposed model incorporating both collaborative guided and formatting rewards.

Our results demonstrate that the model incorporating all proposed stages (variant E) achieved the best recommendation performance across two domains, as shown in Fig. 5. The base model (A) showed the lowest performance among all variants, indicating the need for domain-specific training in recommendation tasks. Models enhanced with GRPO (variants D and E) outperformed the Stage 2-only model (variant B; only instruction-tuning), suggesting that while instruction-tuning enables the model to acquire some domain knowledge, it is insufficient for effective generalization. The variant (C), which applied only reinforcement learning to the base model, generally outperformed the base model but still underperformed compared to (B), (D), and (E), suggesting that the effect of Stage 3 alone—i.e., reinforcement learning without prior alignment—is limited. The difference between variants (D) and (E) was in the reward

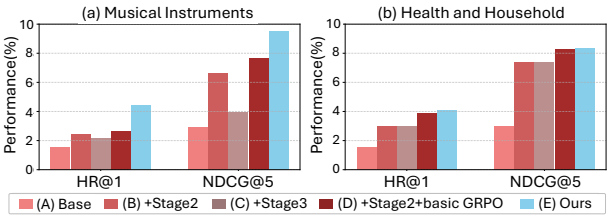

Figure 5: Comparisons between model variants

structure: variant (D) rewards only item matching, while variant (E) additionally incorporates collaborative signals. Our results demonstrate that variant (E) consistently outperforms variant (D), underscoring the importance of collaborative signals in recommendation systems. Results for all domains are provided in Appendix D.

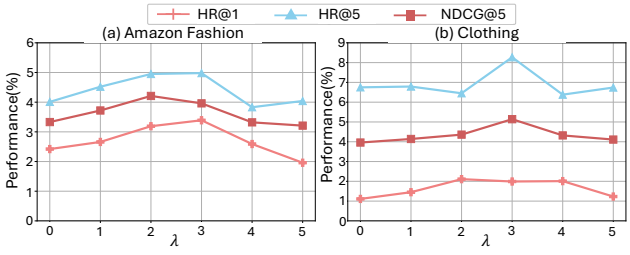

Figure 6: Performance variation with λ in the Collaborative Guided Reward

Effect of Collaborative Confidence (RQ4)

We performed a sensitivity analysis on the hyperparameter λ, which controlled the weight of CF-Rec's confidence in the collaborative-guided reward. According to the gradient analysis, λ guided the gradient updates to increase the likelihood of outputs that matched the target item and exhibited high CF confidence. Experiments on *Amazon Fashion* and *Clothing Shoes and Jewelry* showed that performance improved as λ increased up to 3, but slightly degraded beyond that. This suggested that moderate collaborative guidance in the collaborative-guided reward helped ground the LLM's predictions in user-item interactions. However, excessive reliance on collaborative signals forced the model to mimic CF-Rec's behavior, which hindered its ability to generate accurate outputs, as CF-Rec was not guaranteed to produce optimal recommendations.

Online A/B Test (RQ5)

We deployed our model in a real-world production environment and conducted an online A/B test. The test was performed over a two-week period in July 2025. For comparison, we employed two baseline models: a random control and a traditional machine learning model. We measured performance using the click-through rate metric, as shown in Table 2. This result shows that our model achieved a significant improvement in performance compared to all baselines, demonstrating its practical effectiveness in real-world deployment.

Model	# Impression	# Clicks	CTR	Latency
Random	30.23K	2.81K	9.30%	94ms/call
ML	309.04K	32.98K	10.67%	120ms/call
Ours	15.88K	1.83K	11.49%	138ms/call

Table 2: Online A/B Test Results: Performance Comparison

Effect of Incorporating Rationales (RQ6)

We investigate whether incorporating rationales improves recommendation performance, as illustrated in Fig. 7.

(A) Our Model + Item/Rationale Generation at Inference: Our model is instructed to generate both the recommended item and its rationale during inference.

(B) Our Model + Item-Only Generation at Inference: During inference, the model generates only the recommended item, without rationales.

(C) Our Model *w/o* Rationale Training + Item-Only Generation at Inference: The model is trained without any rationale supervision—Stage 1 excludes rationale generation, Stage 2 only involves auxiliary tasks (task 1 and 2), and Stage 3 excludes rationale-related instructions in the prompt. Inference is performed with item-only generation.

By comparing (A) and (B), we assess the impact of generating rationales during inference. The results show that model variant (A) consistently outperforms (B) across all domains, indicating that prompting the model to generate rationales encourages it to engage in a reasoning process, which in turn improves the recommendation performance. Meanwhile, comparing (B) and (C) reveals the importance of incorporating rationales during training.

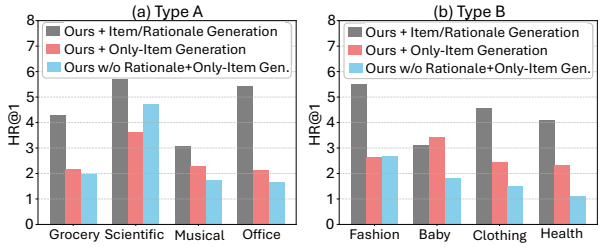

Figure 7: Domain-Wise Performance Comparison: Without vs. With Rationales (Training/Inference)

Conclusion

We propose an LLM-based recommendation framework that combines the strengths of LLM- and CF-based models to achieve accurate recommendations and high-quality rationales. By first building the reasoning foundation through instruction tuning and then aligning with collaborative signals, our approach significantly improves both recommendation performance and rationale quality, as validated by increased click-through rates in real-world deployment.

Acknowledgments

We would like to thank SK Telecom for providing access to the GPU cluster used in this research.

This work was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (RS-2019-II190075, Artificial Intelligence Graduate School Program (KAIST)). This work was also supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. RS-2025-00555621).

References

Gu, Z.; Zou, H. P.; Chen, Y.; Liu, A.; Zhang, W.; and Yu, P. S. 2025. Scaling Laws for Many-Shot In-Context Learning with Self-Generated Annotations. *arXiv preprint arXiv:2503.03062*.

Guo, D.; Yang, D.; Zhang, H.; Song, J.; Zhang, R.; Xu, R.; Zhu, Q.; Ma, S.; Wang, P.; Bi, X.; et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

He, R.; Kang, W.-C.; and McAuley, J. 2017. Translation-based recommendation. In *Proceedings of the eleventh ACM conference on recommender systems*, 161–169.

Hou, Y.; Li, J.; He, Z.; Yan, A.; Chen, X.; and McAuley, J. 2024. Bridging Language and Items for Retrieval and Recommendation. *arXiv preprint arXiv:2403.03952*.

Kang, W.-C.; and McAuley, J. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*, 197–206. IEEE.

Lei, Y.; Lian, J.; Yao, J.; Huang, X.; Lian, D.; and Xie, X. 2024. Recexplainer: Aligning large language models for explaining recommendation models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 1530–1541.

Lin, J.; Wang, T.; and Qian, K. 2025. Rec-r1: Bridging generative large language models and user-centric recommendation systems via reinforcement learning. *arXiv preprint arXiv:2503.24289*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shao, Z.; Wang, P.; Zhu, Q.; Xu, R.; Song, J.; Bi, X.; Zhang, H.; Zhang, M.; Li, Y.; Wu, Y.; et al. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Team, G.; Kamath, A.; Ferret, J.; Pathak, S.; Vieillard, N.; Merhej, R.; Perrin, S.; Matejovicova, T.; Ramé, A.; Rivière, M.; et al. 2025. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Wang, Y.; Kordi, Y.; Mishra, S.; Liu, A.; Smith, N. A.; Khashabi, D.; and Hajishirzi, H. 2022. Self-instruct: Aligning language models with self-generated instructions. *arXiv preprint arXiv:2212.10560*.

Yuan, F.; Karatzoglou, A.; Arapakis, I.; Jose, J. M.; and He, X. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the twelfth ACM international conference on web search and data mining*, 582–590.