Tsinghua Science and Technology
Doi: 10.26599/TST.2026.9010103

Research Article

MERGE: A Memory-Enhanced LLM Agent Framework for Explainable Bundle Recommendation

Yuxuan Li†, Qimeng Niu†, Bowen Hao*, Tingting Li, Yufan Yang, and Chaoran Wang

© The Author(s)

Abstract Bundle recommendation aims to accurately capture user preferences from the complex interactions among users, bundles, and items to generate explainable recommendations. However, existing methods based on large language models (LLMs) treat user interactions as static text, overlooking the dynamic evolution of user preferences. Although recent memory-augmented LLM approaches such as MR.Rec and MemoCRS maintain external memory, their heuristic update rules are non-adaptive and designed solely for individual recommendation, rendering them unsuitable for bundle recommendation scenarios. To address these issues, we propose MERGE, a Memory-Enhanced LLM Agent that autonomously manages memory in response to evolving user behaviors. Specifically, MERGE consists of a Memory Module and a Reasoning Module. The Memory Module employs a token-based hash structure to dynamically manage up-to-date user profiles through insert, update, forget, summary, and query operations, all of which are triggered by the current user interactions. The Reasoning Module then performs multi-stage reasoning based on retrieved information to generate explainable recommendations. We train MERGE using reinforcement fine-tuning (RFT), which first applies supervised fine-tuning (SFT) to establish a foundational understanding of the task, followed by group relative policy optimization (GRPO) to enhance its ability to invoke the Memory and Reasoning Modules. Experimental results on two benchmark datasets show that MERGE significantly outperforms existing state-of-the-art methods in both recommendation accuracy and explanation quality. We open-source our model and code at https://huggingface.co/no1forever/MERGE.

Keywords large language model agent; bundle recommendation; memory augmentation; explainable recommendation; reinforcement fine-tuning

1 Introduction

Recommender systems have become an effective tool for addressing information overload. As an important branch, bundle recommendation focuses on suggesting cohesive sets of items to users, thereby enhancing user experience while concurrently driving sales. Consequently, this approach has been widely applied in e-commerce retail, content subscription services, and digital product bundles[1]. The core challenge in bundle recommendation lies in accurately modeling user preferences, bundle characteristics, and item attributes from their complex interactions.

Traditional bundle recommendation methods employ Bayesian personalized ranking (BPR)[2], matrix factorization[3], recurrent neural networks (RNNs)[4,5], attention mechanisms[6–8], graph neural networks (GNNs)[9–12], and Transformers[13–15] to model the relationships among users, bundles, and items across user-bundle, user-item, and bundle-item interactions. However, these methods are hindered by a lack of interpretability, which prevents them from generating explainable recommendations.

To address this challenge, researchers have leveraged large language models (LLMs) to enhance explainability. Existing approaches fall into two paradigms: LLM-assisted models, which serve as feature enhancement tools for traditional recommendation models (e.g., DisCo[16], SAID[17], LLM-CBR[18]), and LLM-primary models, where the LLM acts as the main recommendation engine, directly consuming user-bundle interactions to generate recommendations and explanations (e.g., RLMRec[19], DUIP[20], CLLM4Rec[21], AICL[22]). However, these LLM-based models fail to capture the evolution of user profiles, leading to sub-optimal

• Yuxuan Li, Qimeng Niu, Bowen Hao, Tingting Li, Yufan Yang, and Chaoran Wang are with the School of Management, Capital Normal University, Beijing 100048, China. E-mail: 1232905018@cnu.edu.cn (Y. Li); 1232905021@cnu.edu.cn (Q. Niu); 6974@cnu.edu.cn (B. Hao); 1232905017@cnu.edu.cn (T. Li); 1232905012@cnu.edu.cn (Y. Yang); 1242905029@cnu.edu.cn (C. Wang).
† These authors contributed equally to this work.
* To whom correspondence should be addressed. E-mail: 6974@cnu.edu.cn (B. Hao).
Manuscript received: 2026-03-28; revised: 2026-08-11; accepted: 2026-11-11

Just Accepted

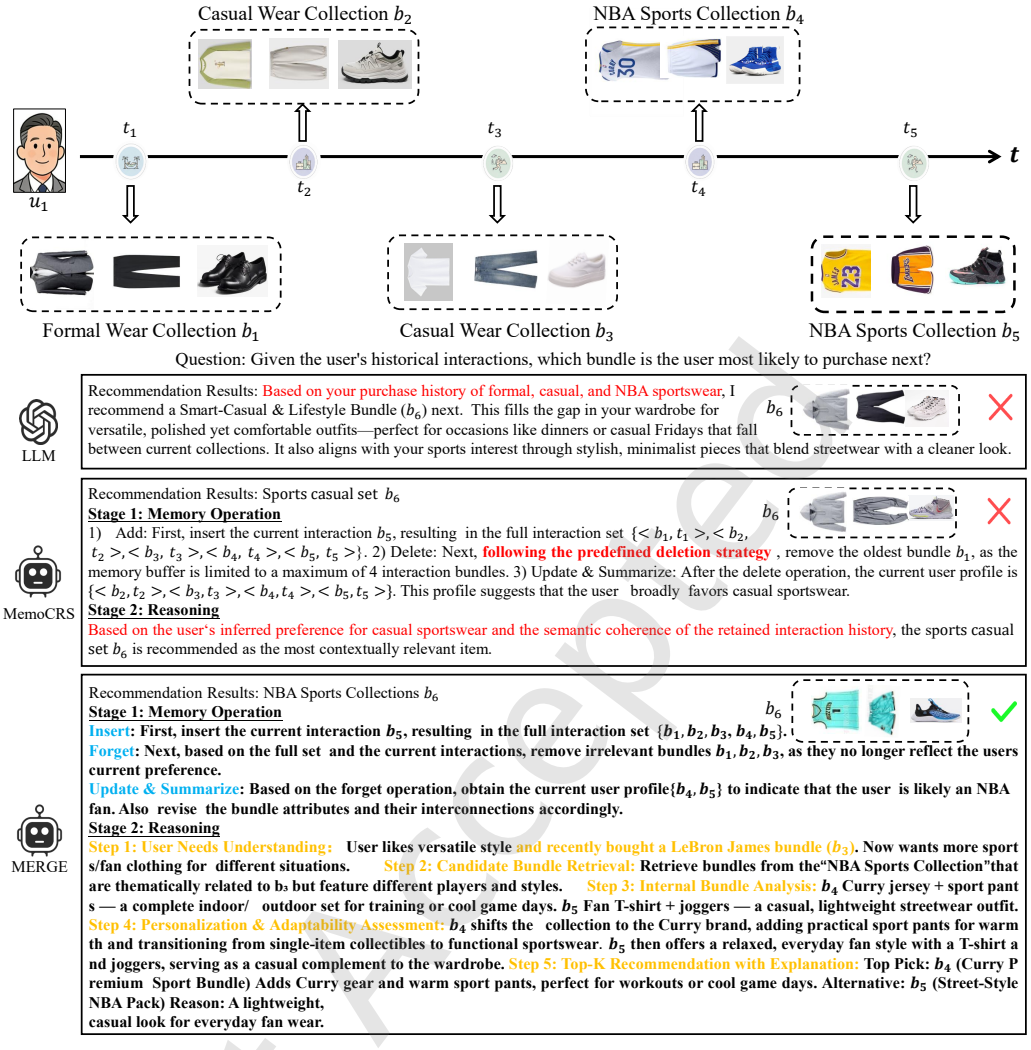


Fig. 1 A toy example of bundle recommendation.

recommendation performance. As shown in Fig. 1, when a user's interests shift rapidly from suits to sportswear to basketball gear, these models are unable to track such preference dynamics, resulting in degraded recommendations.

To capture the evolution of user preferences over time, researchers have proposed memory-based LLM methods that manage memory through insert, delete, update, query, and summarize operations in response to evolving user behaviors; representative methods include MemoCRS (memory-enhanced sequential conversational recommender systems)[23] and MR.Rec[24]. However, their heuristic update rules are non-adaptive and designed solely for individual recommendation, rendering them unsuitable for bundle recommendation scenarios. As shown in Fig. 1, the memory-based approach (MemoCRS) employs a predefined rule-based deletion operation to remove the earliest interacted suits when storage space is limited. Consequently, the resulting user profile is biased, which in turn leads to suboptimal recommendation performance. Besides, the reasoning process of the aforementioned LLM-based models (e.g., MemoCRS and MR.Rec) primarily relies on the internal mechanisms of LLMs, lacking explicit modeling of user profiles, bundle attributes, and item attributes. Therefore, there is an urgent need to design a new method for bundle recommendation that combines explicit reasoning capabilities with a dynamic profile updating mechanism.

To address the above issue, inspired by the autonomous memory management and reasoning capabilities of LLM agents[25], we propose MERGE, the first LLM agent for bundle recommendation, which formulates bundle recommendation as a sequential next-bundle recommendation task. Specifically, MERGE consists of a Memory Module and a Reasoning Module. The Memory Module employs a token-based hash structure to dynamically manage up-to-date user

Just Accepted

profiles through insert, update, forget, summary, and query operations, all of which are triggered by ongoing user interactions rather than predefined memory operation strategies, thereby enabling continuous tracking of dynamic user preferences. The Reasoning Module draws upon dynamic information retrieved from the Memory Module (including user profiles, interaction histories, and bundle descriptions) to execute an interpretable, multi-stage inference process. This process comprises five sequential stages: User Need Understanding, Candidate Bundle Filtering, Internal Bundle Coherence Analysis, Personalized Matching and Dynamic Adaptability Assessment, and Explainable Top-k Ranking. Through this structured pipeline, the model produces recommendations that are both accurate and transparent, thereby transcending the conventional "black-box" paradigm. In practice, we train MERGE under the reinforcement fine-tuning (RFT) paradigm[26], which first applies supervised fine-tuning (SFT) to establish a foundational understanding of the task, and then performs group relative policy optimization (GRPO) to enhance its ability to invoke the Memory and Reasoning Modules. Experimental results on the FashionBundleBench and Music163 datasets demonstrate that MERGE significantly outperforms existing state-of-the-art baseline methods in both top-k recommendation and the quality of generated explanations. The contributions of this paper are as follows:

- We propose MERGE, the first LLM agent equipped with memory-augmented reasoning capabilities that dynamically manages the profiles of the users to address the bundle recommendation task.
- We design a Memory Module to handle evolving user profiles and a Reasoning Module that explicitly models bundle composition logic and personalized matching to enhance transparency and interpretability.
- Experimental results on two benchmark datasets show that MERGE outperforms existing state-of-the-art methods in both recommendation accuracy and explainability.

2 Related Work

2.1 Bundle Recommendation

Bundle recommendation methods fall into two broad categories: traditional methods and LLM-based techniques. Traditional methods leverage a range of models, such as BPR[2], matrix factorization[3], RNNs[4,5], attention mechanisms[6–8], GNNs[9–12], and Transformers[13–15], to learn user profiles and bundle attributes from user-bundle, user-item, and bundle-item interactions. However, these methods often lack interpretability. To solve this problem, LLM-based methods have been proposed, which can be classified into two paradigms. The first enhances traditional models with LLM-encoded representations (DisCo[16], SAID[17], LLM-CBR[18]) but lacks transparency. The second uses LLMs as the main recommender, outputting both bundles and explanations (RLMRec[19], DUIP[20], CLLM4Rec[21], AICL[22]), but it fails to model user profiles, bundle/item attributes, or preference changes over time. Although recent memory-based LLM approaches such as MemoCRS[23] and MR.Rec[24] manage memories via insert, delete, update, and summary operations, they rely on non-adaptive heuristic rules tailored solely for individual recommendation. Consequently, they cannot dynamically manage user profiles to address the bundle recommendation task.

2.2 Memory-Augmented Recommendation

Memory mechanisms play a crucial role in capturing dynamic user preferences. Early research primarily uses RNNs[4,5] to model user interactions. However, they suffer from vanishing gradient problems in long sequences. Subsequent works introduce explicit memory modules[27] to store user's historical interactions to capture long-term preferences. Nevertheless, these memory modules are coarsely designed and lack the support for deletion or update operations. Recently, memory-augmented LLM methods, including MARM[28], LMN[29], MemoCRS[23], and MR.Rec[24], have been developed to enhance recommendation performance by retrieving user historical interactions from memory. Despite these advances, the heuristic update rules employed by these methods remain non-adaptive and fail to effectively capture the evolution of user profiles over time, nor are they tailored for bundle recommendation.

2.3 Reasoning-Based Recommendation

Reasoning-based recommendation enhances interpretability via explicit decision-making and can be categorized into four technical approaches. Symbolic logic-based methods (e.g., KGAT[30], RuleRec[31]) use knowledge graphs or manual rules, offering strong interpretability but poor adaptability to dynamic user behaviors. Neural implicit methods (e.g., KGIN[32], SASRec[13]) infer reasoning pathways implicitly, which yields strong performance yet introduces a significant lack of transparency. LLM-based generative methods (e.g., CoT4Rec[33], RecExplainer[34]) leverage chain-of-thought for rationale generation, yet suffer from prompt and data quality issues. Hybrid frameworks (e.g., LOGIN[35], MoKGNN[36]) enhance GNNs with LLMs or memory modules for dynamic reasoning, yet their use in bundle recommendation, especially

Just Accepted

for explicit reasoning of composition and matching, remains underexplored.

3 Problem Definition

Let $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$ denote the set of users, $\mathcal{I} = \{i_1, i_2, \ldots, i_{|\mathcal{I}|}\}$ the set of items, and $\mathcal{B} = \{b_1, b_2, \ldots, b_{|\mathcal{B}|}\}$ the set of bundles. Each bundle $b_j \in \mathcal{B}$ is composed of any combination of items from $\mathcal{I}$ (e.g., a "spring hiking gear set"). We formulate bundle recommendation as a sequential next-bundle recommendation task. Formally, for a given user u with historical interaction bundle set $\mathcal{A}^u_{t-1} = \{b^u_1, \cdots, b^u_{t-1}\}$ prior to time step t, at time step t, the LLM agent f_θ is required to recommend top-k bundles from the candidate set $\mathcal{C} = \mathcal{B} - \mathcal{A}^u_{t-1}$ to the user, i.e.,

$$o_u = f_\theta\big(\mathcal{R}(\mathcal{M}(u, \mathcal{A}^u_{t-1}))\big), \tag{1}$$

where o_u denotes the top-k recommendation and their explanations, $f_\theta(\cdot)$ is the LLM agent parameterized by θ, $\mathcal{M}(\cdot)$ is the Memory Module responsible for managing user profiles, and $\mathcal{R}(\cdot)$ is the Reasoning Module, which leverages the retrieved profiles alongside bundle attributes from the Memory Module to perform a structured multi-stage reasoning process.

4 Methodology

We propose MERGE, which comprises a Memory Module and a Reasoning Module. The Memory Module employs a token-based hash structure to dynamically manage up-to-date user profiles through insert, update, forget, summary, and query operations, all of which are triggered by the current user interactions. The Reasoning Module then performs multi-stage reasoning based on retrieved information to generate explainable recommendations. We train MERGE under the RFT paradigm, i.e., we first apply SFT to equip the model with a basic understanding of the task; then, we use GRPO to enhance its ability to coordinate memory retrieval and multi-step reasoning. Fig. 2 illustrates the overall framework of MERGE, and the end-to-end inference pipeline is formalized in Eq. (1).

4.1 Memory Module

The Memory Module $\mathcal{M}$ functions as the principal mechanism for modeling dynamic user preferences. Built upon a token-based hash table, $\mathcal{M}$ comprises the following constituent units:

- $\mathcal{M}_U$: Stores the user's historical interaction bundles in chronological order, using the user ID as the key and a natural language description of each interaction as the corresponding value.
- $\mathcal{M}^*_U$: Stores dynamically generated user profile summaries, temporal evolution patterns of user interests, and user-specific sensitivities or preferences regarding key attributes, with each user ID mapped to a natural language description.
- $\mathcal{M}_B$: Stores bundle metadata (e.g., ID, name, category, scenario, and description) using the bundle ID as the key and a natural language description of the meta information as the corresponding value.
- $\mathcal{M}^*_B$: Stores relationships among bundles (e.g., complementary, substitutive, and scenario associations), among items within bundles, and among items across bundles, using the bundle ID or item ID as the key and a natural language description of the relational structure as the corresponding value.
- $\mathcal{M}_I$: Stores detailed item attributes keyed by item ID, along with value-assessment metrics (e.g., price in clothing and popularity in music) that are represented as natural language descriptions.

To enable efficient modeling and querying of dynamic user interests, and drawing inspiration from Toolformer[37], we equip $\mathcal{M}$ with five fundamental operations, implemented as functions that can be directly invoked by LLMs. The interfaces are illustrated in Fig. 3, with detailed descriptions provided below:

- **Insert**: Insert a new user-bundle interaction into $\mathcal{M}_U$, or a new bundle into $\mathcal{B}$, in the form of key-value pairs. Benefiting from the hash table's direct ID-based access, this process achieves $\mathcal{O}(1)$ time.
- **Update**: When new interactions occur from time step $t-1$ to t, append them to the end of the corresponding user's memory sequence. This triggers an updated summary of the user profile and also takes $\mathcal{O}(1)$ time.
- **Forget**: To prevent memory overload, once the number of historical interactions exceeds a predefined threshold φ, the proposed mechanism allows the agent to autonomously determine both *which* and *how many* bundles to evict based on the current memory state. Rather than naively discarding the oldest entries, it scans bundles chronologically from the earliest record and computes the cosine similarity between each bundle and the current user profile (obtained via the summary operation). Bundles with lower similarity scores are iteratively selected for removal until the memory size falls below the threshold, with the exact number of evictions decided by the agent according to the current memory pressure. This adaptive eviction strategy preserves long-term stable user preferences essential for personalized recommendation, with a time complexity of

Just Accepted

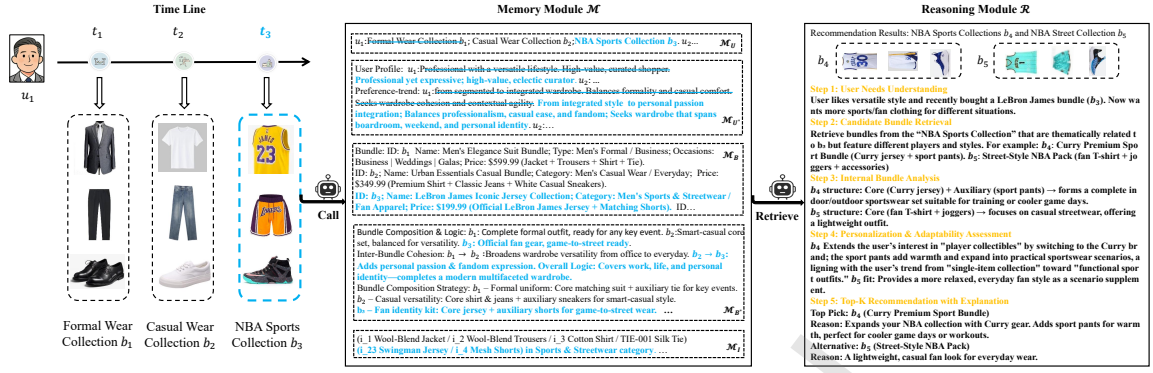


Fig. 2 Overall framework of MERGE.

$\mathcal{O}(\varphi)$.

- **Summary**: Following any initialization, update, or forget operation, we regenerate the user profile and bundle attribute summaries ($\mathcal{M}_U^*$ and $\mathcal{M}_B^*$) based on the current states of $\mathcal{M}_U$ and $\mathcal{M}_B$. This regeneration incurs a time complexity of $\mathcal{O}(n) + \text{LLM_cost}(n)$, where n denotes the number of historical interaction bundles associated with a given user (or $n = 1$ for a single bundle), and $\text{LLM_cost}(n)$ represents the generation overhead of the underlying LLM. Notably, any modification to the historical records such as insertion, update, or forget triggers a complete regeneration of the summaries.
- **Query**: The retrieval phase comprises two primary components: Historical Interaction Lookup and Ranking Retrieval. Historical Interaction Lookup employs a hash-based indexing mechanism, achieving a time complexity of $\mathcal{O}(1)$ for any users, items, or bundles. Ranking Retrieval is further decomposed into two stages. The first stage, Coarse Semantic Filtering, embeds users and bundles using Qwen3 [38] and applies hierarchical navigable small world (HNSW) indexing implemented in FAISS [39], which reduces the candidate set to the top-K bundles (where $K \ll |\mathcal{B}|$) with a complexity of $\mathcal{O}(\log |\mathcal{B}|)$. The second stage, Fine-grained LLM Filtering, feeds user profile summaries to an LLM (e.g., DeepSeek-R1) with the following prompt: *"Based on user profiles, select bundles highly relevant to the user's interests from the Candidate Bundle List $\mathcal{C} = [\ldots]$"*. This stage evaluates the K candidates in $\mathcal{O}(K)$ time. Accordingly, the total time complexity of the retrieval phase is $\mathcal{O}(1) + \mathcal{O}(\log |\mathcal{B}|) + \mathcal{O}(K)$.

4.2 Memory-Augmented Reasoning Module

Although LLMs demonstrate strong general reasoning abilities, they are not inherently tailored to bundle recommendation, making it difficult to effectively model the composition logic within bundles and the matching process with dynamic user preferences. To address this, we introduce a Memory-Augmented Reasoning Module, which transparently reconstructs the evaluation and selection process for bundles through five stages: 1) User Need Understanding, 2) Candidate Bundle Filtering, 3) Internal Bundle Coherence Analysis, 4) Personalized Matching and Dynamic Adaptability Assessment, and 5) Explainable Top-k Ranking, thereby moving beyond "black-box" inference and delivering fully interpretable recommendations. Similar to the Memory Module, the Reasoning Module provides encapsulated tool functions

```
<memory>
operation: "Insert" | "Update" | "Forget" | "Retrieval" | "Summarize"
user_id: str
[interaction_data]: List[Tuple[timestamp, bundle_id, interaction_type]] # Only for insert/update
[memory_threshold]: int # Only for forget
[query_type]: "profile" | "history" | "trend" | "attribute" # Only for retrieval
[bundle_pool]: Dict[bundle_id, description] # Required during initialization
</memory>

<reasoning>
<stage1> Stage1: User Need Understanding and Scenario Identification ... </stage1>
<stage2> Stage2: Candidate Bundle Retrieval and Filtering ... </stage2>
<stage3> Stage3: Bundle Internal Coherence Analysis ... </stage3>
<stage4> Stage4: Personalized Matching and Dynamic Adaptability Assessment ... </stage4>
<stage5> Stage5: Explainable Top-k Generation and Ranking ... </stage5>
</reasoning>
```

Fig. 3 Interface of the Memory and Reasoning modules.

Stage 1: User Need Understanding
As a preference analyst, given the user's recent bundle interactions and profile retrieved from the memory module, infer the user's current core needs (e.g., "commuting outfit", "focus work playlist") and key attribute preferences (e.g., price range, preferred genres).
Output format: A structured user need summary D_u.

Stage 2: Candidate Bundle Filtering
Based on the user need summary D_u, retrieve candidate bundles from the memory module. First, perform semantic matching between D_u and bundle descriptions. Then, expand the results using bundle relationships (e.g., complementary, similar) to enhance diversity. Finally, filter based on user attribute preferences (e.g., price range, genre) to obtain a manageable candidate set $\mathcal{C}'$.

Stage 3: Bundle Internal Coherence Analysis
Evaluate each candidate bundle $b \in \mathcal{C}'$ for internal coherence. Analyze its internal structure (e.g., item roles, relationships) using bundle memory, combined with item attributes from item memory. Assess the following: **completeness** (whether core items are present), **logical consistency** (whether items are functionally complementary and thematically coherent), and **overall value**. Output an internal coherence score $s_{\text{int}}(b)$ with a key rationale.

Stage 4: Personalized Matching and Dynamic Adaptability Assessment
Assess how well candidate bundle b matches the specific user u. Integrate the user need summary D_u and user profile to evaluate: **profile coverage** (overlap between bundle attributes and user's long-term profile), **trend alignment** (whether the bundle aligns with recent preference shifts), and **novelty** (avoiding items already interacted with). Output a personalized matching score $s_{\text{per}}(b)$.

Stage 5: Explainable Top-k Ranking
Compute a composite score for each candidate bundle: $S(b) = \alpha \cdot s_{\text{int}}(b) + \beta \cdot s_{\text{per}}(b) + \gamma \cdot s_{\text{div}}(b)$,where $s_{\text{div}}(b)$ is a diversity score. Select the top-k bundles $\{b_1, b_2, \ldots, b_k\}$ as the final recommendation list and generate natural language explanations for each, incorporating insights from previous stages.

Fig. 4 Prompt templates for the Reasoning Module.

Just Accepted

for the LLM to call, with its interfaces shown in Fig. 3. The details of each reasoning stage are as follows, with the complete prompt illustrated in Fig. 4.

4.2.1 Stage 1: User Need Understanding

Given a target user u, the Reasoning Module first retrieves the corresponding dynamic profile $\mathcal{M}_u^* \in \mathcal{M}_U^*$ from the Memory Module. It then infers the user's current core needs, such as "commute outfit" or "focus work playlist", and key attribute preferences, such as "budget range for clothing" or "preferred music genres". The module outputs a structured user need summary D_u.

4.2.2 Stage 2: Candidate Bundle Filtering

Based on the summary D_u, this stage recalls candidate bundles from the Memory Module. First, we semantically match D_u against the bundle descriptions in $\mathcal{M}_B^*$. Next, we leverage inter-bundle relationships recorded in $\mathcal{M}_B$ (e.g., complementarity or similarity) to expand the results, thereby increasing candidate diversity. Finally, we filter the candidates based on user attribute preferences stored in $\mathcal{M}_U^*$, including price range and genre inclination, to obtain a manageable preliminary candidate set $\mathcal{C}'$.

4.2.3 Stage 3: Internal Bundle Coherence Analysis

This stage evaluates the internal composition coherence of each candidate bundle $b \in \mathcal{C}'$. We invoke $\mathcal{M}_B^*$ to analyze the internal structure of each bundle, including item roles and inter-item relationships. Specifically, we adopt in-context learning (ICL)[40] combined with an LLM-as-a-judge paradigm[41]. To establish ground-truth labels, we first recruit academic volunteers with backgrounds in recommender systems to annotate a randomly selected set of 100 bundles and their constituent items. Each annotator evaluates three key aspects: composition completeness s_1, which assesses whether core items or tracks are present; logical consistency s_2, which examines the complementarity and harmony of items in terms of function, style, or theme; and overall value s_3, which aggregates attribute-level $\mathcal{M}_I$ information into a holistic assessment. Each aspect is scored on a four-level scale: satisfied (5), partially satisfied (3–4), slightly satisfied (1–2), and not satisfied (0). From this annotated pool, we randomly sample 20 bundles out of these 100 bundles as few-shot ground-truth labels to guide the LLM in scoring s_1, s_2, and s_3. The final internal coherence score for each bundle b is then computed as the average of the three aspect scores $s_{\text{int}}(b) = \text{avg}[s_1(b), s_2(b), s_3(b)]$, where $\text{avg}(\cdot)$ denotes the average operation.

4.2.4 Stage 4: Personalized Matching and Dynamic Adaptability Assessment

This stage assesses the degree of matching between candidate bundle b and specific user u. We synthesize the need summary D_u and the user profile $\mathcal{M}_U^*$ to compute three evaluation dimensions: profile coverage s_4, which measures the overlap between the bundle's attributes and the user's long-term profile $\mathcal{M}_U^*$; trend alignment s_5, which evaluates whether the bundle aligns with preference shifts revealed by $\mathcal{M}_U$; and novelty s_6, which avoids recommending content the user has already interacted with in $\mathcal{M}_U$. Following the same paradigm employed in Stage 3, we utilize ICL combined with an LLM-as-a-judge framework to guide the LLM in scoring the three dimensions s_4, s_5, and s_6. These scores are then aggregated to yield a personalized matching score for each bundle b, denoted as $s_{\text{per}}(b) = \text{avg}[s_4(b), s_5(b), s_6(b)]$.

4.2.5 Stage 5: Explainable Top-k Ranking

Finally, we compute a comprehensive score for each candidate bundle by aggregating its internal coherence, personalized matching, and diversity assessments. Similar to the previous stages, we adopt ICL combined with an LLM-as-a-judge paradigm to guide the LLM in evaluating three diversity dimensions: category coverage s_7, which measures the variety of item categories or genres within the bundle; style dispersion s_8, which assesses the stylistic range across the constituent items; and novelty s_9, which captures the proportion of items that are uncommon or exploratory relative to the user's history. The diversity score is then computed as the average of these three dimensions: $s_{\text{div}}(b) = \text{avg}[s_7(b), s_8(b), s_9(b)]$. The overall score for each bundle is then defined as

$$S(b) = \alpha \cdot s_{\text{int}}(b) + \beta \cdot s_{\text{per}}(b) + \gamma \cdot s_{\text{div}}(b), \qquad (2)$$

where α, β, and γ are tunable weighting hyperparameters. Finally, based on the overall score $S(b)$, we select the top-k bundles $\{b_1, \ldots, b_k\}$ as the final recommendation list.

4.3 Model Training

We train MERGE using the RFT paradigm[26]. Specifically, we first apply SFT to establish a foundational understanding of the task, followed by GRPO to enhance the agent's capability in invoking the Memory and Reasoning Modules.

4.3.1 SFT Stage

We construct a high-quality instruction dataset using a combination of self-refinement[42] and ICL. The process proceeds as follows: First, we employ DeepSeek-R1[43] to generate initial recommendations and corresponding explanations for a given user u and their historical interaction sequence $\mathcal{A}_{t-1}^u$. The model then self-evaluates its output and assesses the

Just Accepted

The predefined criteria consist of four dimensions, each scored on a 0–5 scale.
Bundle Internal Coherence (0–5): Evaluates the logical combination of items within a recommended bundle, considering functional complementarity, thematic consistency, and role completeness (e.g., presence of core items). Higher scores indicate stronger internal coherence.
User Preference Matching (0–5): Assesses the semantic alignment between the recommended bundle and the user's core preferences (e.g., style, category, price range), derived from user profile and historical interactions.
Explanation Quality (0–5): Measures the fluency, informativeness, and clarity of the generated explanation. A high-quality explanation clearly articulates the recommendation rationale and links it to user needs and bundle characteristics.
Diversity and Novelty (0–5): Evaluates the overall diversity of the top-K recommendation list and the novelty of each bundle relative to the user's historical interactions, avoiding repetitive or overly similar suggestions.
The maximum total score is 20. Only instances achieving a total score above a predefined threshold (e.g., 15) are retained for the final training corpus.

Fig. 5 Predefined criteria.

difficulty of the recommendation task, guided by 20 manually annotated few-shot examples provided by domain experts in recommender systems. The evaluation criteria, shown in Fig. 5, encompass four dimensions: Bundle Internal Coherence, User Preference Matching, Explanation Quality, and Diversity and Novelty. Based on the critique, the model regenerates refined outputs for two iterative rounds. From all generated candidates, we rank samples by difficulty score in descending order and select the top 500 hard samples for further manual annotation, thereby constructing high-quality ground-truth labels. Three domain experts independently annotate each sample, with the final label determined by majority voting to ensure reliability and consistency. The annotation covers two dimensions: recommendation list accuracy (hit rate at $k = 5$, HR@5) and explanation quality (1–5 score). Finally, we randomly sample 10 examples from the 500 expert-calibrated instances to serve as in-context examples, and apply the ICL approach to batch-produce the high-quality training corpus. Each constructed data instance is represented as $(q_u, \mathcal{A}^u_{t-1}, o_u)$, where q_u denotes the user's basic profile information, $\mathcal{A}^u_{t-1}$ the interaction history up to time step $t-1$, and o_u the expert-calibrated ground-truth recommendation. The SFT objective is formulated as

$$\mathcal{L}_{\mathrm{SFT}} = -\mathbb{E}_{(q_u, \mathcal{A}^u_{t-1}, o_u)\sim\mathcal{D}} \left[\sum_{t=1}^{T} \log P_\theta(y_t \mid x) \right], \quad (3)$$

where $\mathcal{D}$ is the training dataset, $x = \{q_u, \mathcal{A}^u_{t-1}\}$ denotes the model input, $y = \{o_u\}$ the supervision signal, T the length of the target sequence, and $P_\theta(y_t \mid x)$ the probability of generating token y_t given input x, parameterized by θ.

4.3.2 RL Stage

We further employ GRPO, starting from the SFT-initialized policy in Eq. (3), to enable the model to autonomously invoke the Memory and Reasoning Modules. For a given input state x, GRPO samples a group of actions from the current policy π_θ, producing G candidate answers $\{o_1, o_2, \ldots, o_G\}$, where each $o_i \sim \pi_\theta(o \mid x)$, $i = 1, 2, \ldots, G$. This group-based sampling strategy encourages the model to explore diverse responses, thereby mitigating premature homogenization of responses.

Prompt=f """Please evaluate the quality of the reasoning process for a bundle recommendation system.
Model Output:
Reflection: [...]
Reasoning: {...}
Evaluation Dimensions (Total Score: 1.0 point):
- Text Quality (0.0–0.5 points): Language fluency, logical clarity, depth of analysis.
- User Preference Understanding (0.0–0.25 points): Accuracy of capturing the user's core needs and preferences.
- Bundle Internal Coherence (0.0–0.25 points): Logical consistency and complementarity among items in the recommended bundles.
Important: Ensure that total_score is the sum of the first three scores, and that all scores are within the specified ranges. """

Fig. 6 Prompt template for reasoning reward evaluation.

Subsequently, a reward model evaluates each generated response, producing a reward sequence $\{r_1, r_2, \ldots, r_G\}$. To guide the model in effectively utilizing the Memory and Reasoning modules, we design the following reward functions:

- **Format Reward** R_f: To enforce structured reasoning and output consistency, we require the model to use four tags: <think>, <memory>, <reason>, and <rec>. Each correctly applied tag yields 0.25 points, for a maximum total of 1.0.
- **Recommendation Reward** R_{rec}: For each generated answer o_i ($i = 1, 2, \cdots, G$), we evaluate its alignment with the ground-truth recommendation list using two metrics: normalized discounted cumulative gain at k (NDCG@k) and hit ratio at k (HR@k). The reward is then defined as the arithmetic mean of the two scores, as given in Eq. (4):

$$R^i_{\mathrm{rec}} = 0.5 \cdot \mathrm{NDCG@}k + 0.5 \cdot \mathrm{HR@}k \quad (4)$$

- **Reasoning Reward** R_{rea}: To mitigate bias inherent in vanilla LLM evaluation, we adopt the same ICL combined with an LLM-as-a-judge approach described in Section 4.2 to guide the LLM in scoring the reasoning reward. The evaluation considers three criteria: Text Generation Quality (weight 0.5), Internal Logical Coherence (weight 0.25), and Degree of Personalization (weight 0.25). Fig. 6 shows the evaluation prompt.

The final reward is calculated as a weighted sum: $r = w_1 R_f + w_2 R_{\mathrm{rec}} + w_3 R_{\mathrm{rea}}$, where w_1, w_2, and w_3 are tunable hyperparameters that balance the contributions of each type of reward. Finally, we maximize the GRPO objective function in Eq. (5),

$$\mathcal{F}(\theta) = \mathbb{E}\Big[\sum_{i=1}^{G} \min\big(\rho_i A_i,\ \mathrm{clip}(\rho_i,\ 1-\epsilon,\ 1+\epsilon) A_i\big) \Big], \quad (5)$$

where the expectation is taken over prompts $x \sim \mathcal{D}$ and a group of G sampled outputs $\{o_i\}_{i=1}^G$ drawn from the old policy $\pi_{\mathrm{old}}(\cdot \mid x)$, $\rho_i = \pi_\theta(o_i \mid x) \,/\, \pi_{\mathrm{old}}(o_i \mid x)$ is the importance sampling ratio, $A_i = (r_i - \mathrm{mean}(r)) \,/\, \mathrm{std}(r)$ is the advantage function, and ϵ is the clipping hyperparameter.

Just Accepted

5 Experiments

We design experiments to answer the following research questions:

- RQ1: Does MERGE outperform existing baseline methods in terms of recommendation accuracy and explainability?
- RQ2: Does MERGE effectively capture the temporal evolution of user preferences across multiple prediction steps, and how does it compare to baselines?
- RQ3: How do the Memory Module's modeling of evolving profiles and the Reasoning Module's multi-stage process contribute to a more accurate and interpretable recommendation?
- RQ4: What necessitates the inclusion of both the SFT and RL phases within the RFT training pipeline?
- RQ5: How do the designs of the format, reasoning, and recommendation rewards in the reward function affect model performance?
- RQ6: What are the time efficiencies of the Memory Module's core operations (e.g., insert, update, forget, summary, and query), and what are the computational costs associated with each stage of the Reasoning Module?
- RQ7: How do hyperparameters affect the overall recommendation performance?

5.1 Experimental Setup

5.1.1 Datasets

We construct experimental data based on two public datasets, FashionBundleBench and Music163. The FashionBundleBench dataset provides user interaction records with clothing items and predefined clothing bundle combinations (e.g., "commute outfit" and "outdoor sports set"), where each bundle comprises several clothing items with complementary or matching relationships. The Music163 dataset provides user listening and collection histories for playlists, where we consider each playlist as a bundle composed of multiple songs with similar styles, moods, or themes. We represent user interaction history as the set of playlists the user has listened to or collected. Notably, in both datasets, the bundles for user interaction are chronologically ordered. We summarize dataset statistics in Table 1. Notably, we perform the leave-one-out evaluation [13] at each time step.

Table 1 Dataset statistics, where U, B, and I denote the set of users, bundles and items, respectively; U-B and B-I represent user-bundle and bundle-item interactions, respectively. Fashion is short for FashionBundleBench due to space constraints.

Dataset	U	B	I	U-B	B-I	Tags
Fashion	6,000	14,939	31,869	58,752	82,422	11,604
Music163	4,500	45,638	4,284	69,666	136,833	12,970

5.1.2 Baselines

We compare MERGE with the following seven types of baseline methods.

- Traditional methods: including the deep learning-based approach SASRec [13], which employs a Transformer for sequential bundle recommendation.
- Vanilla LLMs: including vanilla LLM-based approaches DeepSeek-R1-8B [43] and LLaMA3-8B [44], which are directly prompted with historical user interactions to produce both recommendations and explanations.
- LLM-based methods: LLMCBR [18], which integrates LLMs with semantic refinement and collaborative modeling.
- In-context learning methods: CoT-ICL [45] combines chain-of-thought with in-context learning for step-by-step reasoning and top-k recommendation generation, and AICL [22] employs adaptive in-context learning with retrieval and self-correction.
- MERGE-API methods: call the application programming interface (API) of a commercial LLM (DeepSeek-R1 [43]) with user histories and bundle descriptions as text prompts, employing structured templates to trigger external tools (e.g., memory retrieval) via function calling. While it preserves LLM interpretability and basic tool invocation capabilities, it lacks the dynamic memory and multi-stage reasoning that characterize the full MERGE framework.
- Memory-based LLM methods: including MemoCRS [23] and MR.Rec [24], which employ predefined strategies to manage memory through insert, delete, update, query, forget, and summarize operations in response to evolving user behaviors.
- MERGE and its variants: MERGE-DPO (trained with SFT followed by direct preference optimization (DPO), replacing GRPO), MERGE-M (Memory Module only), MERGE-R (Reasoning Module only), MERGE-SFT (trained only with SFT), and MERGE-GRPO (trained only with GRPO). We use these variants to analyze the contribution of each component.

5.1.3 Evaluation Metrics

We use three types of evaluation metrics.

- Recommendation metrics: hit ratio (HR@k), normalized discounted cumulative gain (NDCG@k), and mean reciprocal rank (MRR), with $k = 5$ by default.
- Explainability metrics: bilingual evaluation understudy (BLEU), recall-oriented understudy for gisting evaluation (ROUGE), where ROUGE* denotes the average score of ROUGE-1, ROUGE-2, and ROUGE-L, and BERTScore.

Just Accepted

- Human evaluation metrics: comprises four carefully designed dimensions. UPU (User Preference Understanding) measures whether the model correctly identifies the user's core needs, preference categories, interest trends, and contextual intentions, calculated as the average of per-annotator scores $\mathrm{UPU} = \frac{1}{N}\sum_{i=1}^{N} S^i_{\mathrm{upu}}$. BCA (Bundle Coherence Assessment) evaluates the logic of item combinations within a recommended bundle, focusing on complementarity, thematic consistency, and role completeness, formulated as $\mathrm{BCA} = \frac{1}{N}\sum_{i=1}^{N} S^i_{\mathrm{bca}}$. RQS (Recommendation Quality Score) provides a comprehensive assessment of the overall recommendation list across four dimensions: user need matching, item combination rationality, diversity, and novelty, with the final score computed as the average of per-annotator holistic ratings $\mathrm{RQS} = \frac{1}{N}\sum_{i=1}^{N} S^i_{\mathrm{rqs}}$. Notably, S^i_{upu}, S^i_{bca}, and S^i_{rqs} are the scores annotated by each annotator. Among these, UPU, BCA, and RQS primarily evaluate recommendation performance, while EQ (Explanation Quality) is specifically designed to assess the generated explanations from three weighted aspects: transparency (S_{T}), comprehensibility (S_{C}), and persuasiveness (S_{P}), yielding the composite score $\mathrm{EQ} = 0.4 \times S_{\mathrm{T}} + 0.3 \times S_{\mathrm{C}} + 0.3 \times S_{\mathrm{P}}$, where $S_{\mathrm{T}} = \frac{1}{N}\sum_{i=1}^{N} S^i_{\mathrm{T}}$, $S_{\mathrm{C}} = \frac{1}{N}\sum_{i=1}^{N} S^i_{\mathrm{C}}$, and $S_{\mathrm{P}} = \frac{1}{N}\sum_{i=1}^{N} S^i_{\mathrm{P}}$, in which S^i_{T}, S^i_{C}, and S^i_{P} are the scores annotated by each annotator. Notably, S^i_{upu}, S^i_{bca}, S^i_{rqs}, S^i_{T}, S^i_{C}, and S^i_{P} are human evaluation scores on a 10-point scale, further divided into four levels: Poor (0–2), Fair (3–5), Good (6–8), and Excellent (9–10). Human evaluations are conducted by three professional domain experts in a single-blind manner. Each sample was independently evaluated twice by each expert. Inter-rater agreement among the three experts was measured using Fleiss' kappa[46] ($\kappa = 0.7347$), which is designed for agreement assessment with multiple raters, confirming the reliability of the evaluation.
- Hallucination Rate (Halluc.): This metric measures the proportion of atomic claims in generated explanations that contradict the user's history, bundle content, or common sense. The evaluation pipeline is: (1) Use DeepSeek-API to split each explanation into atomic claims; (2) Mark claims that contradict or are unsupported by the above sources as hallucinations. The hallucination rate is:

$$\mathrm{Halluc.} = \frac{1}{M}\sum_{j=1}^{M}\frac{H_j}{N_j}, \tag{6}$$

where M is the sample count, N_j is the number of atomic claims in the j-th sample, and H_j is the hallucination count. Lower scores indicate better factual consistency. Using atomic claims as the minimal unit avoids coarse-grained sentence-level biases. Following the ICL + LLM-as-a-judge paradigm (mentioned in Section 4.2), three experts annotate 100 samples as few-shot examples, which then guide the LLM to evaluate the remaining samples in batch.

5.1.4 *Experimental Details*

We use LLaMA3-8B[44] as the backbone LLM and employ DeepSeek-R1-8B[43] to verify whether MERGE is sensitive to different backbone models. We perform all experiments on a single NVIDIA RTX PRO 6000 GPU using low-rank adaptation (LoRA). We configure the learning rate as 5×10^{-6}, temperature coefficient as 0.7, training batch size as 1, inference batch size as 4, the KL coefficient β_{KL} of GRPO as 0.001, rollout number as 4, and maximum sequence length as 3,000 tokens. We set all reward function weights w_1, w_2, and w_3 to 1, and the composition weights α, β, and γ (defined in Eq. (2)) to 1. To improve candidate bundle retrieval efficiency, we use Qwen3[38] to pre-embed bundle descriptions, and compute semantic similarity based on the user need summary, retrieving the $K = 200$ most relevant bundles for each user as the candidate set $\mathcal{C}$. We average all experimental results over three independent runs, and report the corresponding statistically significant p-values.

5.2 Overall Performance (RQ1 and RQ2)

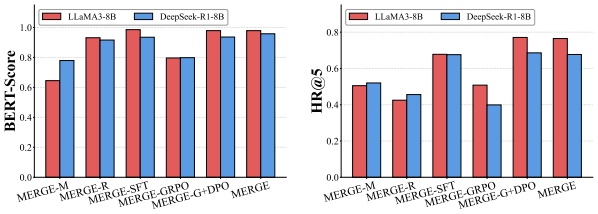
The overall performance on the FashionBundleBench and Music163 datasets across all methods is shown in Tables 2 and 3. All performance improvements are statistically significant with $p < 0.001$ using the Mann-Whitney U test. Besides recommendation accuracy and explainability, we report the hallucination rate defined in Eq. (6), where lower values indicate better factual consistency. Our findings are as follows:

- MERGE and its variant MERGE-DPO achieve the best performance on the vast majority of metrics, demonstrating their ability to model dynamic user preferences and explain bundle logic and item relationships.
- The deep learning-based method SASRec performs the worst and suffers from limited explainability and transparency.
- Although LLMCBR outperforms deep learning-based methods, it still relies on traditional collaborative signals and lacks explicit modeling of dynamic user preferences as well as interpretable reasoning. Consequently, it achieves lower scores on both ranking and explainability metrics.
- Vanilla LLM methods (LLaMA3-8B and DeepSeek-R1-8B) achieve competitive recommendation scores, yet they

Just Accepted

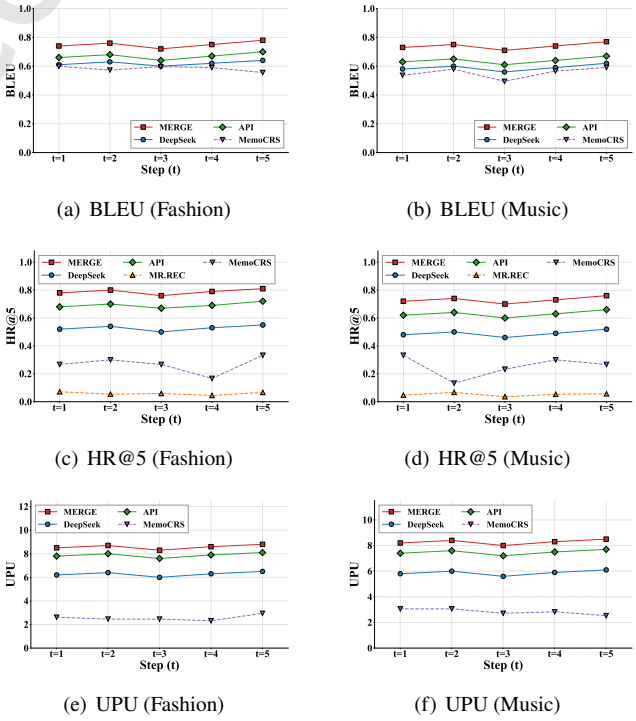
still fall short of MERGE because they are not fine-tuned for bundle recommendation.

- Compared with vanilla LLM methods, ICL methods (CoT-ICL and AICL) further improve performance through the incorporation of chain-of-thought reasoning. However, they still underperform relative to MERGE, indicating that RFT effectively enhances recommendation performance.
- MERGE greatly outperforms MERGE-API on all metrics, likely because the underlying LLM is a general-purpose model without specialized knowledge of bundle recommendation.
- Memory-based methods exhibit inconsistent performance. MR.Rec, which relies solely on memory retrieval for user profiling without fine-grained memory maintenance (e.g., removing irrelevant item interactions or updating user profiles), performs poorly. MemoCRS, while supporting user profile evolution through Create, Read, Update, and Delete operations, relies on heuristic algorithms for memory maintenance and lacks autonomous updating capabilities, resulting in inferior performance compared to MERGE.
- We further compare MERGE with its variant MERGE-DPO. As shown in Tables 2 and 3, the two variants achieve broadly comparable performance at the 7B scale, with no consistent winner across all metrics. This aligns with our controlled experiments in Section 5.5, which show that GRPO's group-relative advantage becomes increasingly pronounced only with larger backbones ($\geqslant$14B). Given this scale-dependent trend and the potential for future extension to larger models, we adopt GRPO as the RL component in MERGE. Notably, both GRPO and DPO yield substantial gains over MERGE-SFT, confirming that the RL phase within RFT is essential regardless of the specific algorithm. Section 5.5 presents the comparison between SFT+DPO and SFT+GRPO, along with GRPO setup details and the impact of model scale.
- We evaluate the framework's sensitivity to different base models (LLaMA3-8B and DeepSeek-R1-8B) on the FashionBundleBench dataset. Results are shown in Fig. 7, which demonstrate that both models achieve comparable performance. This indicates that MERGE is robust to the choice of backbone model.
- Furthermore, to assess whether MERGE captures the temporal evolution of user preferences, we construct five prediction tasks for each user using a *leave-last-t-out* strategy. Specifically, for $t \in \{1, 2, 3, 4, 5\}$, we remove the most recent t interactions of a user and require the model to predict the **earliest among the removed bundles** (i.e., the $t-1$ most recent interaction), using only interactions before the earliest held-out bundle as input. We compare MERGE, MERGE-API, MR.Rec (where applicable), MemoCRS, and vanilla DeepSeek-R1-8B on FashionBundleBench and Music163, evaluated with BLEU, UPU, HR@5, and NDCG@5. As shown in Fig. 8, MERGE outperforms both baselines across all five time steps, demonstrating its ability to capture evolving user preferences and maintain superior multi-step prediction performance.



(a) BERTScore (b) HR@5

Fig. 7 Comparison of different backbone LLMs.



(a) BLEU (Fashion) (b) BLEU (Music)

(c) HR@5 (Fashion) (d) HR@5 (Music)

(e) UPU (Fashion) (f) UPU (Music)

Fig. 8 Temporal evaluation on FashionBundleBench and Music163. The x-axis t denotes the number of held-out future interactions (leave-last-t-out).

5.3 Ablation Study (RQ3–RQ5)

We conduct ablation experiments on three key components to evaluate their contributions: the Memory and Reasoning

Just Accepted

Table 2 Overall performance on FashionBundleBench.

Model	HR@5	NDCG@5	MRR	BLEU	ROUGE*	BERTScore	UPU	BCA	RQS	EQ	Halluc.
SASRec	0.0594	0.0360	0.0285	—	—	—	—	—	—	—	—
LLaMA3-8B	0.4020	0.1990	0.1396	0.0165	0.0281	0.6467	4.66	6.60	4.66	5.82	0.7130
DeepSeek-R1-8B	0.3920	0.2079	0.1481	0.0971	0.0499	0.7739	4.96	6.47	5.52	6.22	0.5038
LLMCBR	0.1682	0.1068	0.0868	—	—	—	—	—	—	—	—
CoT-ICL	0.4220	0.2392	0.1802	0.5232	0.5144	0.9446	3.68	7.22	4.42	6.17	0.4799
AICL	0.4600	0.2717	0.2083	—	—	—	—	—	—	—	—
MERGE-API	0.1600	0.0437	0.0717	0.2263	0.2854	0.7455	5.12	6.66	5.59	6.44	0.2480
MR.Rec	0.0667	0.0376	0.0275	—	—	—	—	—	—	—	—
MemoCRS	0.3330	0.1440	0.0830	0.0031	0.1559	0.2050	2.55	2.65	2.54	1.92	0.9977
MERGE-DPO	**0.7710**	**0.4488**	**0.3690**	0.5955	0.7494	0.9788	**8.03**	7.32	**7.20**	**7.68**	0.1037
MERGE-M	0.5050	0.2981	0.2314	0.0161	0.0282	0.6450	6.14	7.13	5.72	6.67	0.7063
MERGE-R	0.4250	0.2653	0.1806	0.5136	0.5204	0.9314	5.70	6.51	5.92	6.38	0.3506
MERGE-SFT	0.6780	0.4369	0.3581	0.6527	**0.7605**	0.9854	6.25	**7.33**	5.96	6.71	0.2312
MERGE-GRPO	0.5080	0.3093	0.2453	0.0807	0.0414	0.7963	5.63	6.04	5.43	5.36	0.6691
MERGE	0.7650	0.4396	0.3587	**0.6955**	0.7484	**0.9887**	7.73	7.23	7.17	7.59	**0.1026**

Table 3 Overall performance on Music163.

Model	HR@5	NDCG@5	MRR	BLEU	ROUGE*	BERTScore	UPU	BCA	RQS	EQ	Halluc.
SASRec	0.0498	0.0295	0.0229	—	—	—	—	—	—	—	—
LLaMA3-8B	0.3950	0.2819	0.2442	0.0526	0.0440	0.6889	3.08	5.26	4.17	5.46	0.8267
DeepSeek-R1-8B	0.2840	0.1280	0.1976	0.1022	0.0614	0.7463	6.46	5.74	6.11	6.40	0.6868
LLMCBR	0.0423	0.0245	0.0187	—	—	—	—	—	—	—	—
CoT-ICL	0.4160	0.1988	0.1297	0.5160	0.3980	0.9210	5.15	5.27	5.17	5.44	0.6842
AICL	0.3200	0.1512	0.0980	—	—	—	—	—	—	—	—
MERGE-API	0.2400	0.0883	0.1255	0.1551	0.2384	0.7164	3.17	4.99	4.01	5.04	0.4790
MR.Rec	0.0566	0.0331	0.0244	—	—	—	—	—	—	—	—
MemoCRS	0.2670	0.1407	0.1117	0.0017	0.1573	0.2452	1.92	0.88	1.37	1.30	0.7188
MERGE-DPO	0.6020	0.5773	0.5627	0.6834	**0.5685**	0.9341	8.06	7.00	7.20	7.74	**0.0460**
MERGE-M	0.3965	0.2830	0.2447	0.0536	0.0441	0.6877	5.71	5.42	5.64	5.93	0.8236
MERGE-R	0.4180	0.1939	0.1606	0.5240	0.4019	0.9282	5.77	5.98	5.32	5.72	0.6742
MERGE-SFT	0.5964	0.5759	0.5689	0.6808	0.5530	0.9242	8.12	**7.11**	**7.28**	**7.85**	0.2520
MERGE-GRPO	0.4080	0.2838	0.2925	0.0458	0.0606	0.7849	5.74	5.33	5.94	6.03	0.7803
MERGE	**0.6240**	**0.5899**	**0.5723**	**0.6835**	0.5668	**0.9344**	**8.07**	6.96	7.13	7.59	0.0501

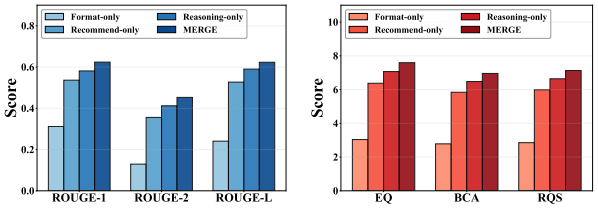
Table 4 Prompt ablation study: performance under different prompt variants.

Prompt variants	**Description**	**BLEU**	**ROUGE***	**BERTScore**
Original	Full reasoning prompt (complete model)	**0.6955**	**0.7484**	**0.9887**
Synonym replacement	Replaced key terms with synonyms	0.6772	0.7359	0.9610
Dimension removal	Removed explicit scoring criteria for diversity	0.6451	0.7321	0.9819
Stage-wise ablation				
Remove Stage 1	Remove "User Need Understanding" stage	0.2439	0.2663	0.8194
Remove Stage 2	Remove "Candidate Bundle Filtering" stage	0.6079	0.7525	0.9622
Remove Stage 3	Remove "Internal Bundle Coherence Analysis" stage	0.5412	0.5939	0.9348
Remove Stage 4	Remove "Personalized Matching and Adaptability Assessment" stage	0.2792	0.2976	0.8313
Remove Stage 5	Remove "Explainable Top-k Ranking" stage	0.5481	0.6159	0.9698

Just Accepted

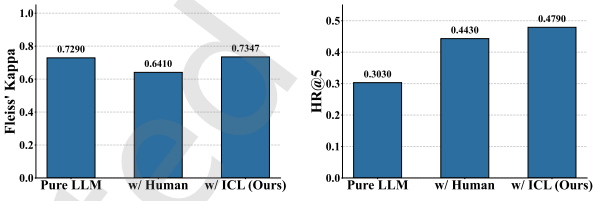
modules (RQ3), the SFT and GRPO training stages (RQ4), and the reward functions (RQ5). Tables 2–3 and Fig. 9 present the results, showing that:

- Removing the Memory Module (MERGE-R) reduces recommendation performance, confirming its crucial role in tracking dynamic user preferences for robust recommendations.
- Removing the Reasoning Module (MERGE-M) lowers explainability scores, confirming that explicit modeling of bundle consistency and item relationships is essential.
- SFT and GRPO are both essential for RFT. SFT must come before GRPO, and using both yields optimal performance, as MERGE-SFT and MERGE-GRPO each perform worse than MERGE.
- All reward components are essential. Removing the format reward harms performance; removing the recommendation reward reduces accuracy, and removing the reasoning reward reduces explainability.
- We further examine the judgement bias in LLM evaluation within the Reasoning Module. Specifically, we compare our proposed *ICL + LLM-as-a-judge* approach with two variants: a) *Pure LLM-as-a-judge*: directly scoring by LLM without any human annotation; and b) *LLM-as-a-judge + Human Annotation*: human scoring based on LLM-generated outputs. We report Fleiss' kappa [46] between human consensus and LLM scores, and HR@5 for ranking quality in Fig. 10. Results show that *ICL + LLM-as-a-judge* consistently outperforms both variants on both metrics, demonstrating its effectiveness in reducing LLM scoring bias.
- To explore robustness against prompt variations, we conduct a prompt ablation study on 200 random samples from FashionBundleBench using two variants: a) Prompt variants: we explore two prompt variants. The first substitutes key phrases in Fig. 4 with synonyms, for example, "User Preference Matching" $\rightarrow$ "User Preference Alignment", "Explanation top-k Ranking" $\rightarrow$ "Interpretability top-k Ranking". The second replaces the diversity scoring criteria $[s_1(b), s_2(b), s_3(b)]$, $[s_4(b), s_5(b), s_6(b)]$, $[s_7(b), s_8(b), s_9(b)]$ with a single unified score per stage $s_{\text{int}}(b)$, $s_{\text{per}}(b)$, $s_{\text{div}}(b)$ for stages 3, 4, and 5, respectively. b) Stage-wise ablation: we investigate the contribution of each stage by systematically removing individual stages to assess their respective utility. Table 4 summarizes the ablation results for the prompt variants. Regarding prompt variants, our findings indicate that MERGE is largely insensitive to synonym substitution, with negligible performance variation. In contrast, MERGE exhibits slight sensitivity to the diversity scoring criteria; performance decreases slightly when diversity scoring is restricted to a single unified score. Regarding stage-wise ablation, MERGE consistently outperforms all its ablated variants, with performance degrading upon the removal of any stage, confirming that every stage is beneficial.



(a) ROUGE* (b) Human evaluation scores

Fig. 9 Ablation study of reward function combinations.



(a) Fleiss' kappa (b) HR@5

Fig. 10 Ablation study of judgement bias analysis, $p < 0.01$.

5.4 Model Efficiency Analysis (RQ6)

To assess the efficiency of the Memory Module's core operations and each stage within the Reasoning Module (RQ6), we decompose MERGE's inference pipeline into multiple stages for cost analysis. As shown in Table 5, we measure the processing times of the Memory Module's core operations (insert, update, forget, summary, and query) and the processing times of each stage in the Reasoning Module. Additionally, we also report input/output token counts, time complexity, and relative time proportion for each stage on FashionBundleBench. The results show that the Memory Module and Reasoning Module add only negligible overhead to the execution time due to their low time complexity.

5.5 Hyperparameter Analysis (RQ7)

We analyze the impact of key hyperparameters on recommendation performance.

1) Specifically, we investigate whether the GRPO rollout number affects performance. To this end, we evaluate LLaMA3-8B on the FashionBundleBench dataset, with results shown in Fig. 11. We find that GRPO performance generally improves as the rollout number increases, though the trends differ across metrics. For BLEU and ROUGE*,

Just Accepted

Table 5 Model retrieval and generation time analysis. Generation time (s) is the total time taken by each model to generate output; token count is the total length of the generated output.

Stage	Time (s)	Tokens	Time proportion
Memory Module			
Insertion	0.0020	–	0.15%
Forgetting	0.0014	–	0.11%
Updating	0.0085	–	0.64%
Retrieval	0.0382	–	2.88%
Summarization	0.3142	200	23.69%
Reasoning Module			
Need understanding	0.0518	133	3.91%
Candidate filtering	0.0912	108	6.88%
Internal coherence	0.3487	127	26.29%
Personalized matching	0.3375	98	25.45%
Top-k generation	0.4395	977.4	33.14%
Comparison models			
LLaMA3-8B (vanilla)	0.7630	1023.9	–
LLaMA3-8B + CoT + ICL	0.9860	1019.7	–
MERGE (overall)	1.3260	977.4	–

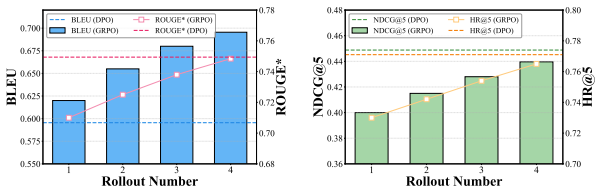
GRPO surpasses DPO when the rollout reaches 4. For HR@5 and NDCG@5, GRPO improves steadily but remains slightly behind DPO even at rollout 4.

2) We also assess each of the composition weights α, β, and γ introduced in Eq. (2). To this end, we evaluate LLaMA3-8B on the FashionBundleBench dataset, with results reported in Fig. 12. The results indicate that each composition weight is necessary for optimal performance.

3) We further investigate whether the choice of base LLM backbone size affects the performance of DPO and GRPO. To this end, we evaluate Qwen2.5-7B, Qwen2.5-14B, and Qwen2.5-72B on the FashionBundleBench dataset, with results reported in Fig. 13. The results demonstrate that DPO and GRPO achieve comparable overall performance at the 7B scale, with each method showing distinct strengths. However, as model size increases, GRPO consistently surpasses DPO, and this advantage grows increasingly pronounced with larger models.

4) We further analyze the effect of memory capacity φ on performance. To this end, we evaluate LLaMA3-8B on the FashionBundleBench dataset, with the results shown in Table 6. As the results indicate, the best performance occurs at $\varphi = 8$ (HR@5 = 0.67). A small capacity ($\varphi = 2$) leads to early forgetting of key information, while a large capacity ($\varphi = 12$) adds noise and computational cost. Thus, we set $\varphi = 8$ for all experiments.

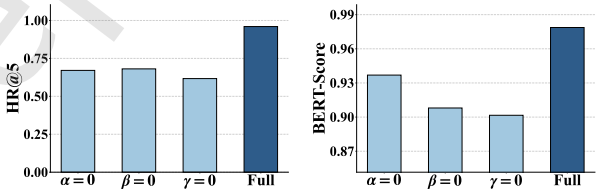
5) To evaluate whether the proposed irrelevant forgetting mechanism can effectively capture evolving user profiles, we evaluate LLaMA3-8B on the Music163 dataset. We adopt the *leave-last-t-out* strategy and compare two forgetting variants:



(a) BLEU and ROUGE* (b) NDCG@5 and HR@5

Fig. 11 Comparison of different rollout numbers.

a) Retain All (No Forget); and b) Chronologic Forgetting (FIFO), which discards the oldest entries. For these three models, the memory capacity is set as $\varphi = 8$. We randomly select 500 Music163 samples, each with an average of 10.8 interactions. The results are shown in Fig. 14, which implies: i) MERGE consistently achieves the best performance, demonstrating its ability to selectively remove irrelevant items for long-term memory. ii) The No Forget strategy performs the worst, as it cannot capture the user profile accurately.



(a) HR@5 (b) BERTScore

Fig. 12 Ablation study of weight parameters α, β, and γ in Eq. (2).

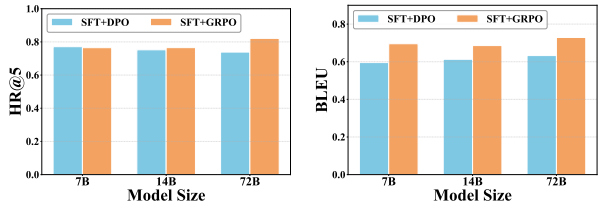
Table 6 Ablation study of memory capacity φ on FashionBundleBench.

φ	HR@5	NDCG@5	UPU	BCA	RQS	EQ
2	0.61	0.47	7.18	7.01	6.28	7.15
4	0.65	0.52	7.72	7.53	6.79	7.70
6	0.64	0.50	7.58	7.39	6.64	7.55
8	**0.67**	**0.54**	**8.02**	**7.82**	**7.08**	**8.00**
10	0.63	0.49	7.55	7.36	6.62	7.52
12	0.60	0.46	7.10	6.92	6.20	7.08

6 Conclusion

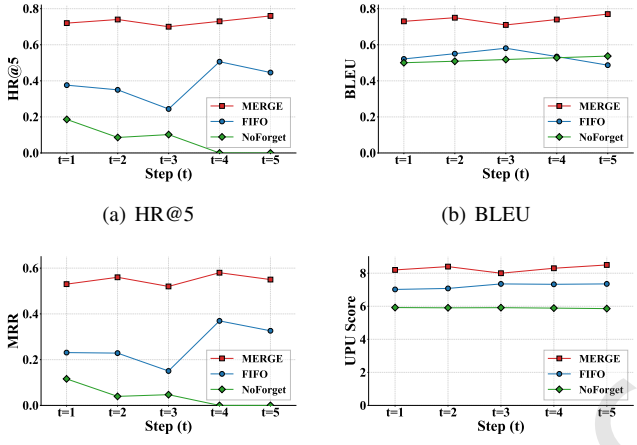
We introduce MERGE, a memory-enhanced LLM agent for bundle recommendation, which employs a Memory Module to dynamically update user profiles from ongoing interactions and a Reasoning Module to produce explainable recommendations. The model is trained via RFT, a paradigm that integrates SFT with GRPO to improve its coordination of both modules. Experiments on two benchmark datasets show that MERGE surpasses existing methods in both recommendation accuracy and explanation quality.

Just Accepted



(a) HR@5 (b) BLEU

Fig. 13 Comparison of different model scales with SFT+DPO and SFT+GRPO.



(a) HR@5 (b) BLEU

(c) MRR (d) UPU

Fig. 14 Forgetting comparison of irrelevant forgetting mechanism and its variant on Music163 across five time steps.

Acknowledgment

This research was supported by the National Natural Science Foundation of China (No. 62402328), the Undergraduate "Qiyan" Program of Beijing Natural Science Foundation (No. QY25392), and the Capital Normal University Talent Support Program (No. 26563005079).

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

Reference

[1] M. Sun, L. Li, M. Li, X. Tao, D. Zhang, Q. Xie, et al., A survey on bundle recommendation: Methods, applications, and challenges, arXiv preprint arXiv:2411.00341, 2024.

[2] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, BPR: Bayesian personalized ranking from implicit feedback, in *Proc. 25th Conf. Uncertainty in Artificial Intelligence (UAI)*, 2009, pp. 452–461.

[3] L. Chen, Y. Liu, X. He, L. Gao, and Z. Zheng, Matching user with item set: Collaborative bundle recommendation with deep attention network, in *Proc. 28th Int. Joint Conf. Artificial Intelligence (IJCAI)*, 2019, pp. 2095–2101.

[4] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, Session-based recommendations with recurrent neural networks, in *Proc. 4th Int. Conf. Learning Representations (ICLR)*, 2016.

[5] D. Jannach and M. Ludewig, When recurrent neural networks meet the neighborhood for session-based recommendation, in *Proc. 11th ACM Conf. Recommender Systems (RecSys)*, 2017, pp. 306–310.

[6] J. Li, P. Ren, Z. Chen, C. Ren, T. Lian, and J. Nie, Neural attentive session-based recommendation, in *Proc. 26th ACM Int. Conf. Information and Knowledge Management (CIKM)*, 2017, pp. 1419–1428.

[7] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, STAMP: Short-term attention/memory priority model for session-based recommendation, in *Proc. 24th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, 2018, pp. 1831–1839.

[8] Y. Cao, J. Liu, M. Shi, et al., Service recommendation based on attentional factorization machine, in *Proc. IEEE Int. Conf. Services Computing (SCC)*, 2019, pp. 189–196.

[9] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, Session-based recommendation with graph neural networks, in *Proc. 33rd AAAI Conf. Artificial Intelligence (AAAI)*, 2019, pp. 346–353.

[10] Z. Wang, W. Wei, G. Cong, X.-L. Li, X.-L. Mao, and M. Qiu, Global context enhanced graph neural networks for session-based recommendation, in *Proc. 43rd Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, 2020, pp. 169–178.

[11] Y. Li, C. Gao, H. Luo, D. Jin, and Y. Li, Enhancing hypergraph neural networks with intent disentanglement for session-based recommendation, in *Proc. 45th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, 2022, pp. 1997–2002.

[12] R. Jia, X. Zhou, L. Dong, and S. Pan, Hypergraph convolutional network for group recommendation, in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, 2021, pp. 260–269.

[13] W.-C. Kang and J. McAuley, Self-attentive sequential recommendation, in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, 2018, pp. 197–206.

[14] F. Sun, J. Liu, J. Wu, C. Pei, X. Lin, W. Ou, and P. Jiang, BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer, in *Proc. 28th ACM Int. Conf. Information and Knowledge Management (CIKM)*, 2019, pp. 1441–1450.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., Attention is all you need, in *Proc. Adv. Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[16] K. Du, J. Chen, J. Lin, Y. Xi, H. Wang, X. Dai, et al., DisCo: Towards harmonious disentanglement and collaboration between tabular and semantic space for recommendation, in *Proc.*

Just Accepted

30th ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD), 2024, pp. 666–676.

[17] J. Hu, W. Xia, X. Zhang, C. Fu, W. Wu, Z. Huan, et al., Enhancing sequential recommendation via LLM-based semantic embedding learning, in *Companion Proc. ACM Web Conf. (WWW Companion)*, 2024, pp. 103–111.

[18] S. Liu, C. Li, M. Zhao, L. Zhang, and J. Bu, LLMCBR: Large language model-based multi-view and multi-grained learning for bundle recommendation, in *Proc. 34th ACM Int. Conf. Information and Knowledge Management (CIKM)*, 2025, pp. 1892–1902.

[19] X. Ren, W. Wei, L. Xia, L. Su, S. Cheng, J. Wang, et al., Representation learning with large language models for recommendation, in *Proc. ACM Web Conf. (WWW)*, 2024, pp. 3464–3475.

[20] X. Xu, Z. Xu, P. Yu, and J. Wang, Enhancing user intent for recommendation systems via large language models, in *Proc. Int. Conf. Artificial Intelligence and Machine Learning Research (CAIMLR)*, vol. 13635, 2024, pp. 46–54.

[21] Y. Zhu, L. Wu, Q. Guo, L. Hong, and J. Li, Collaborative large language model for recommender systems, in *Proc. ACM Web Conf. (WWW)*, 2024, pp. 3162–3172.

[22] Z. Sun, K. Feng, J. Yang, X. Qu, H. Fang, Y.-S. Ong, and W. Liu, Adaptive in-context learning with large language models for bundle generation, in *Proc. 47th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, 2024, pp. 966–976.

[23] Y. Xi, W. Liu, J. Lin, B. Chen, R. Tang, W. Zhang, and Y. Yu, MemoCRS: Memory-enhanced sequential conversational recommender systems with large language models, in *Proc. 33rd ACM Int. Conf. Information and Knowledge Management (CIKM)*, 2024, pp. 2585–2595.

[24] J. Huang, X. Zou, L. Xia, et al., MR. Rec: Synergizing memory and reasoning for personalized recommendation assistant with LLMs, arXiv preprint arXiv:2510.14629, 2025.

[25] G. Zhang, H. Geng, et al., The landscape of agentic reinforcement learning for LLMs: A survey, *Transactions on Machine Learning Research*, 2026.

[26] Z. Zhang, Q. Dong, Q. Zhang, et al., Why reinforcement fine-tuning enables MLLMs to preserve prior knowledge better: A data perspective, in *Proc. Int. Conf. Learning Representations (ICLR)*, 2026.

[27] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, Improving sequential recommendation with knowledge-enhanced memory networks, in *Proc. 41st Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, 2018, pp. 505–514.

[28] X. Lv, J. Cao, S. Guan, X. Zhou, Z. Qi, Y. Zang, et al., MARM: Unlocking the recommendation cache scaling-law through memory augmentation and scalable complexity, in *Proc. 34th ACM Int. Conf. Information and Knowledge Management (CIKM)*, 2025, pp. 2022–2031.

[29] H. Lu, Z. Chai, Y. Zheng, Z. Chen, D. Xie, P. Xu, et al., Large memory network for recommendation, in *Companion Proc. ACM Web Conf. (WWW Companion)*, 2025, pp. 1162–1166.

[30] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, KGAT: Knowledge graph attention network for recommendation, in *Proc. 25th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD)*, 2019, pp. 950–958.

[31] W. Ma, M. Zhang, Y. Cao, W. Jin, C. Wang, Y. Liu, et al., Jointly learning explainable rules for recommendation with knowledge graph, in *Proc. ACM Web Conf. (WWW)*, 2019, pp. 1210–1221.

[32] X. Wang, T. Huang, D. Wang, Y. Yuan, Z. Liu, X. He, et al., Learning intents behind interactions with knowledge graph for recommendation, in *Proc. ACM Web Conf. (WWW)*, 2021, pp. 878–887.

[33] W. Yue, Y. Yin, X. Zhang, B. Shi, T. Liang, and J. Wan, CoT4Rec: Revealing user preferences through chain of thought for recommender systems, in *Proc. 39th AAAI Conf. Artificial Intelligence (AAAI)*, vol. 39, 2025, pp. 13142–13151.

[34] Y. Lei, J. Lian, J. Yao, X. Huang, D. Lian, and X. Xie, RecExplainer: Aligning large language models for explaining recommendation models, in *Proc. 30th ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD)*, 2024, pp. 1530–1541.

[35] Y. Qiao, X. Ao, Y. Liu, et al., LOGIN: A large language model consulted graph neural network training framework, in *Proc. 18th ACM Int. Conf. Web Search and Data Mining (WSDM)*, 2025, pp. 232–241.

[36] H. Yan, C. Li, J. Yin, et al., MoKGNN: Boosting graph neural networks via mixture of generic and task-specific language models, in *Proc. 18th ACM Int. Conf. Web Search and Data Mining (WSDM)*, 2025, pp. 242–250.

[37] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, E. Hambro, et al., Toolformer: Language models can teach themselves to use tools, in *Proc. Adv. Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023, pp. 68539–68551.

[38] A. Yang, A. Li, B. Yang, et al., Qwen3 technical report, arXiv preprint arXiv:2505.09388, 2025.

[39] Y. A. Malkov and D. A. Yashunin, Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 4, pp. 824–836, 2020.

[40] T. B. Brown et al., Language models are few-shot learners, in *Proc. Adv. Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 1877–1901.

[41] J. Gu, X. Jiang, Z. Shi, H. Tan, X. Zhai, C. Xu, et al., A survey on LLM-as-a-judge, *The Innovation*, vol. 7, p. 101253, 2026.

[42] A. Madaan, N. Tandon, P. Gupta, et al., Self-Refine: Iterative refinement with self-feedback, in *Proc. Adv. Neural Information Processing Systems (NeurIPS)*, vol. 36, 2023, pp. 46534–46594.

[43] DeepSeek-AI, DeepSeek-R1: Incentivizing reasoning capa-

Just Accepted

bility in LLMs via reinforcement learning, arXiv preprint arXiv:2501.12948, 2025.

[44] A. Grattafiori et al., The Llama 3 herd of models, arXiv preprint arXiv:2407.21783, 2024.

[45] V. Kothapalli, H. Firooz, and M. Sanjabi, CoT-ICL lab: A synthetic framework for studying chain-of-thought learning from in-context demonstrations, in *Proc. 63rd Annu. Meeting of the Association for Computational Linguistics (ACL)*, 2025, pp. 14620–14642.

[46] J. L. Fleiss, Measuring nominal scale agreement among many raters, *Psychological Bulletin*, vol. 76, no. 5, pp. 378–382, 1971.

Author biography



Yuxuan Li is currently an undergraduate student with the School of Management, Capital Normal University, Beijing, China. Her research interests include large language model-based recommendation systems and data mining.



Qimeng Niu is currently an undergraduate student with the School of Management, Capital Normal University, Beijing, China. Her main research interests include explainable large language models for recommendation.



Bowen Hao received the Ph.D. degree from the School of Information, Renmin University of China, Beijing, China, in 2022. He is currently an associate professor with Capital Normal University, Beijing, China. His research interests include data mining, and large language models for recommendation and medicine.



Tingting Li is currently an undergraduate student with the School of Management, Capital Normal University, Beijing, China. Her research interests include large language model-based recommendation systems, and data analysis and visualization.



Yufan Yang is currently an undergraduate student with the School of Management, Capital Normal University, Beijing, China. His research interests include programming for large language model-based recommendation systems.



Chaoran Wang is currently an undergraduate student with the School of Management, Capital Normal University, Beijing, China. Her research interests include personalized recommendation algorithms based on large language models.

Just Accepted