Action First: Leveraging Preference-Aware Actions for More Effective Decision-Making in Interactive Recommender Systems

Renting Rui
Computer Science and Technology
Shanghai Jiao Tong University
Shanghai, China
ruirenting@sjtu.edu.cn

Yunjia Xi
Computer Science
Shanghai Jiao Tong University
Shanghai, China
xiyunjia@sjtu.edu.cn

Weiwen Liu
Huawei Noah's Ark Lab
Shenzhen, China
liuweiwen8@huawei.com

Jianghao Lin
Computer Science and Technology
Shanghai Jiao Tong University
Shanghai, China
chiangel@sjtu.edu.cn

Bo Chen
Huawei Noah's Ark Lab
Shenzhen, China
chenbo116@huawei.com

Ruiming Tang
Huawei Noah's Ark Lab
Shenzhen, China
tangruiming@huawei.com

Weinan Zhang*
Shanghai Jiao Tong University
Shanghai, China
wnzhang@sjtu.edu.cn

Yong Yu*
Shanghai Jiao Tong University
Shanghai, China
yyu@sjtu.edu.cn

Abstract

Interactive recommender systems (IRSs) aim to meet user needs through natural language dialogues, optimizing recommendations with minimal interactions. Typically, IRSs are based on large language models (LLMs). Existing methods generally consist of two stages: decision-making (deciding whether to recommend or ask clarification questions) and action execution (generating recommendations or clarification questions). These methods usually follow a decision-first paradigm, where the model first decides on the action based on past conversations, and then executes the corresponding action. Since LLMs struggle to process a large number of candidate items, the recommendation process is often carried out in collaboration with external recommendation tools, which provide a small candidate set for LLMs to refine.

Existing methods face two key information gaps: (1) In the decision-making stage, the decision-first paradigm relies solely on past conversations, leading to incomplete decisions due to the uncertainty of subsequent actions' outcomes. (2) In the action execution stage, there is a unidirectional flow from external recommendation tools to LLMs, where these tools fail to interpret user preferences effectively, thus reducing the recommendation accuracy. To address these challenges, we introduce Action-First Interactive Recommender System(AF-IRS), a novel model that uses preference-aware actions to guide decision-making. Our Action-First paradigm informs decisions with future recommendations, ensuring more accurate decisions. Additionally, we establish a bidirectional interaction loop between LLMs and external recommendation tools, enabling LLMs to interpret and transmit session preferences for more precise recommendations. Experimental results on three benchmark datasets demonstrate that AF-IRS significantly improves both recommendation accuracy and efficiency, addressing the information gaps in both stages.

CCS Concepts

• **Information systems** → **Recommender systems**.

Keywords

Interactive Recommender Systems; Large Language Models (LLMs); Action-First

ACM Reference Format:
Renting Rui, Yunjia Xi, Weiwen Liu, Jianghao Lin, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. 2025. Action First: Leveraging Preference-Aware Actions for More Effective Decision-Making in Interactive Recommender Systems. In *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3726302.3729885

1 Introduction

Conversational Recommender Systems (CRSs) have made significant advancements in recent years, due to the integration of intelligent conversational interfaces and natural language processing (NLP) techniques [34, 61]. Unlike traditional recommendation systems, which typically rely on users' past behaviors and collaborative signals to generate recommendations [2, 15, 26], CRSs allow users to explicitly express their preferences, intentions, and feedback to recommendations in multi-turn natural language dialogues. Additionally, CRSs can proactively ask clarification questions to gather

*Weinan Zhang and Yong Yu are the co-corresponding authors.

SIGIR '25, Padua, Italy

ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3729885

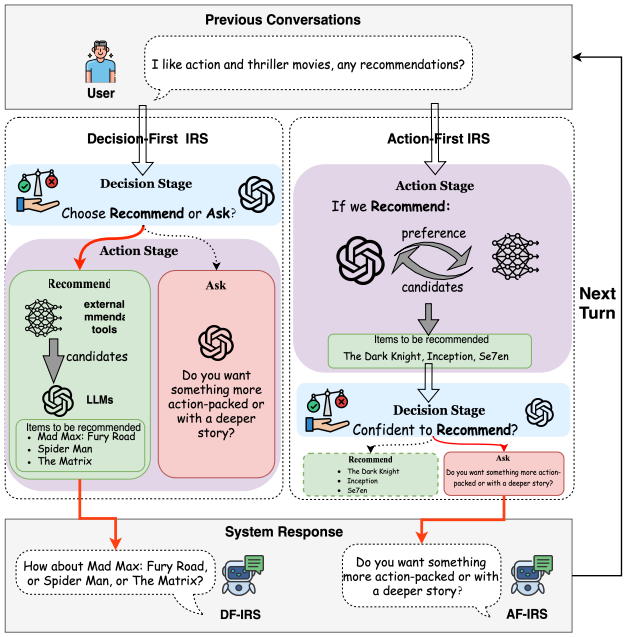

Figure 1: Comparison between traditional Decision-First paradigm and our Action-First paradigm in IRSs.

more specific information, creating a more interactive and personalized experience [3, 42]. This dynamic interaction allows the system to understand and interpret users' real needs better, ensuring more accurate and relevant recommendations.

One of the primary goals for multi-turn CRSs is to deliver accurate recommendations in the fewest possible turns. Early approaches often rely on filling predefined attributes into template-based questions to inquire about user preferences [6, 50], such as "Do you like action movies?" In these settings, users are typically confined to template-based responses like "yes" or "no". However, with the emergence of large language models (LLMs), conversational agents are now able to interact with users in a more natural, fluid manner. This enables users to express more explicit, detailed, and dynamic preferences that go beyond predefined templates [8, 33]. Furthermore, due to their powerful reasoning capabilities, LLMs can make more informed decisions and effectively utilize external tools at each conversation turn, enhancing recommendation accuracy [12, 29]. Such CRSs are often referred to as interactive recommendation systems (IRSs), where they leverage LLMs to engage users through natural language, aiming to recommend the correct products in the fewest turns possible [14, 49, 54].

In this interactive recommendation systems, LLMs are tasked with two key roles: decision-making and action execution. Existing models typically follow the Decision-First paradigm, where decision-making comes before action execution. As shown in the left part of Figure 1, LLMs receive the past conversation and must decide the next action, *e.g.*, *recommend items* or *ask clarification questions* based on dialogues. If the LLM has gathered enough information, it will choose to recommend; otherwise, it will choose to ask more questions to clarify the user's preferences. Once a decision is made, the action stage follows. The ask action can be led by LLMs for further preference gathering. Conversely, the recommendation requires collaboration between LLMs and external recommendation tools. This is because LLMs cannot handle large candidate item pools due to their limited context capacity. Thus, existing works utilize external tools to provide high-quality candidates that LLMs then refine [21, 28]. However, these approaches still suffer from information gaps in the above two stages.

*Firstly, the prevailing **decision-first** paradigm in IRSs often leads to suboptimal decisions due to insufficient information.* In this paradigm, LLMs make decisions solely on the user's conversation history. However, since the recommendation action involves a joint effort between the LLM and external recommendation tools, the LLM lacks insight into what candidate items the external tools will provide and whether they will meet the user's needs, leading to information gaps. For example, as shown in the left part of Figure 1, the recommendation action suggests movies like *Mad Max: Fury Road*, *Spider-Man*, and *The Matrix*, which fit the action genre but don't fully align with the user's thriller preference. However, since the decision stage is unaware of the upcoming recommendations, it assumes a sufficient understanding of the user's preferences and decides to recommend. This can waste conversation turns and harm the user experience. This demonstrates the limitation of decision-first paradigm, where decision-making is not fully informed by the potential outcomes of subsequent actions.

Secondly, in the recommendation action, the information flow between LLMs and external recommenders is unidirectional. Due to the limited context capacity, LLMs cannot process large candidate pools, so they rely on external recommendation tools to generate a smaller, high-quality candidate set. However, these conventional recommenders, typically based on ID features, struggle to understand the textual nuances within the dialogue. Thus, the flow of information is unidirectional, with external tools passing candidates to LLMs, while LLMs are unable to transfer the extracted preferences to external tools. Some methods leverage SQL-based filters, but they can only work with predefined attributes, not the nuanced preferences expressed during the conversation [14, 35]. This unidirectional information flow highlights that even when a user's needs are clear, external recommenders fail to deliver an optimal set of candidates, often leading to suboptimal recommendations.

To address these two challenges, we propose a novel **A**ction **F**irst **I**neteractive **R**ecommendation **S**ystem (AF-IRS), which leverages dynamic preference-aware actions to guide effective decision-making. Firstly, to fill information gaps, we introduce a new standard operating procedure **Action-First** as illustrated in the right part of Figure 1. This paradigm shifts actions, especially recommendations, to occur before decision-making. Thus, LLMs can access future recommendations in advance, providing them with more comprehensive information to make well-informed decisions. Secondly, to solve the issue of unidirectional information flow, we establish a **bidirectional interaction loop** between LLMs and recommendation tools. We first leverage LLMs to construct a preference hierarchical tree that maps user's dynamic preferences to the candidate items. In each turn, the dynamic preferences extracted by LLMs are passed to external recommenders, which filter and refine items through collaborative signals. The refined candidates are then returned to LLMs for further refinement. This facilitates

recommendations that better align with user's needs. Our main contributions can be summarized as follows:

- We identify two key information gaps in IRSs: insufficient information in decision-making and unidirectional information flow during recommendation. To the best of our knowledge, this is the first work to address the information gaps in IRSs.
- We propose AF-IRS, which introduces a new paradigm **Action-First** that prioritizes actions to guide effective decision-making, with a well-designed **bidirectional interaction loop** module between external recommenders and LLMs, generating high-quality items that assist decision-making.
- Extensive experiments on three benchmarks demonstrate that our model achieves a higher recommendation success rate and fewer dialogue turns, ensuring both accuracy and efficiency.

2 Related Works

2.1 Conversational Recommender Systems

Conversational recommender systems (CRS) dynamically capture user preferences through dialogues and are categorized into open-ended generation and attribute-based approaches [11]. Open-ended methods aim to generate fluent conversations and accurate recommendations by combining external resources, such as item metadata, knowledge graphs, and user histories, with fine-tuned language models [3, 22, 23, 30, 34, 39, 42, 52, 57, 58], but often overlook multi-turn decision-making strategies. Attribute-based approaches model the dialogue as a Markov Decision Process (MDP), using reinforcement learning to optimize policies for querying or recommending [4, 7, 13, 17, 19, 20, 55], yet constrain users to template-based responses [53, 56], limiting interaction naturalness. Recent advances integrate Large Language Models (LLMs) into CRS to enable free-form, strategic conversations, leading to the development of interactive recommender systems (IRSs) [14, 49, 54], which we focus on to enhance recommendation accuracy while minimizing dialogue turns.

2.2 Large Language Models for Interactive Recommender Systems

Large Language Models (LLMs) have shown strong knowledge and reasoning abilities in natural language processing and recommendation tasks [24, 25, 38, 40, 41, 41]. In interactive recommender systems (IRSs), LLMs are mainly applied to user simulators and IRS agents. As user simulators, LLMs mimic real user behavior through prompt learning and agent modeling [32, 43, 47, 48, 50, 59, 60]. As IRS assistants, LLMs interact naturally to provide recommendations [5, 9, 12, 51], though they suffer from popularity bias and lack domain expertise. To address this, studies integrate external tools like recommender systems and knowledge graphs [14, 16, 21], leveraging LLMs' planning abilities [10, 31, 44] with methods such as CoT [37], ReAct [46], Reflexion [27], ToT [45], and Self-Inspiring [35]. Plan-first strategies and multi-agent coordination are further used to enhance decision-making [14, 36]. However, existing methods often rely solely on conversation history, limiting performance in domain-scarce settings. Our work proposes generating recommendation items first, allowing the LLM to focus on planning and decision-making, thereby improving domain adaptation and recommendation quality.

3 Preliminaries

The goal of Interactive Recommender Systems (IRSs) is to provide accurate recommendations through a multi-turn dialogue with the user. Specifically, the system aims to minimize the number of interaction turns while maximizing recommendation relevance by querying user preferences in natural language.

Let U and I represent the sets of users and items, respectively. Each user $u \in U$ has an interaction history H_u and a set of target items $T_u \subset I$ that the user will like. Each item $i \in I$ is characterized by a set of attributes A_i , a title, and a description, all maintained by the system. The goal of IRSs is to recommend any item in T_u as soon as possible.

As described in iEvalLM [32], a conversation session begins with the user providing textual-based intentions related to their target items, reflecting their preferences. Based on this input, the system determines its next action: either asking the user contextually relevant questions about their preferences or recommending a set of candidate items I_{cand}. The user then responds by either answering the questions or accepting/rejecting the recommendations. This process continues until the user either accepts the recommendation or terminates the conversation.

The primary challenge is to decide, at each turn, which items to recommend or what questions to ask, with the objective of identifying the user's target items in the fewest possible turns.

The Decision-First Paradigm consists of three sequential stages: decision-making, action execution, and response generation. Firstly, in the decision-making stage, the model uses LLMs to determine whether to recommend items or ask clarification questions based on the conversation history. If sufficient information is available, the model decides to recommend; otherwise, it chooses to ask for more information. Secondly, in the action execution stage, if the decision is to recommend, the system generates a list of candidate items I_{cand} using either LLMs or external recommendation tools. If the decision is to ask, the system generates a clarification question Q using LLMs. Lastly, in the response generation stage, LLMs are used to generate a natural language response to be delivered to the user, which is consistent across all methods, including our own.

4 Methodology

In this section, we detail the methodology of our proposed AF-IRS, as illustrated in Figure 2. Unlike Decision-First paradigm, which places the decision stage before the action stage, our novel approach prioritizes the action stage to guide effective decision-making through preference-aware actions. First, we introduce the action stage, where we establish a bidirectional interaction loop between LLMs and external recommendation tools to generate high-quality candidate items. Next, we describe the decision stage, where the generated recommendations inform the decision process, enabling the system to determine whether to recommend items or ask clarification questions.

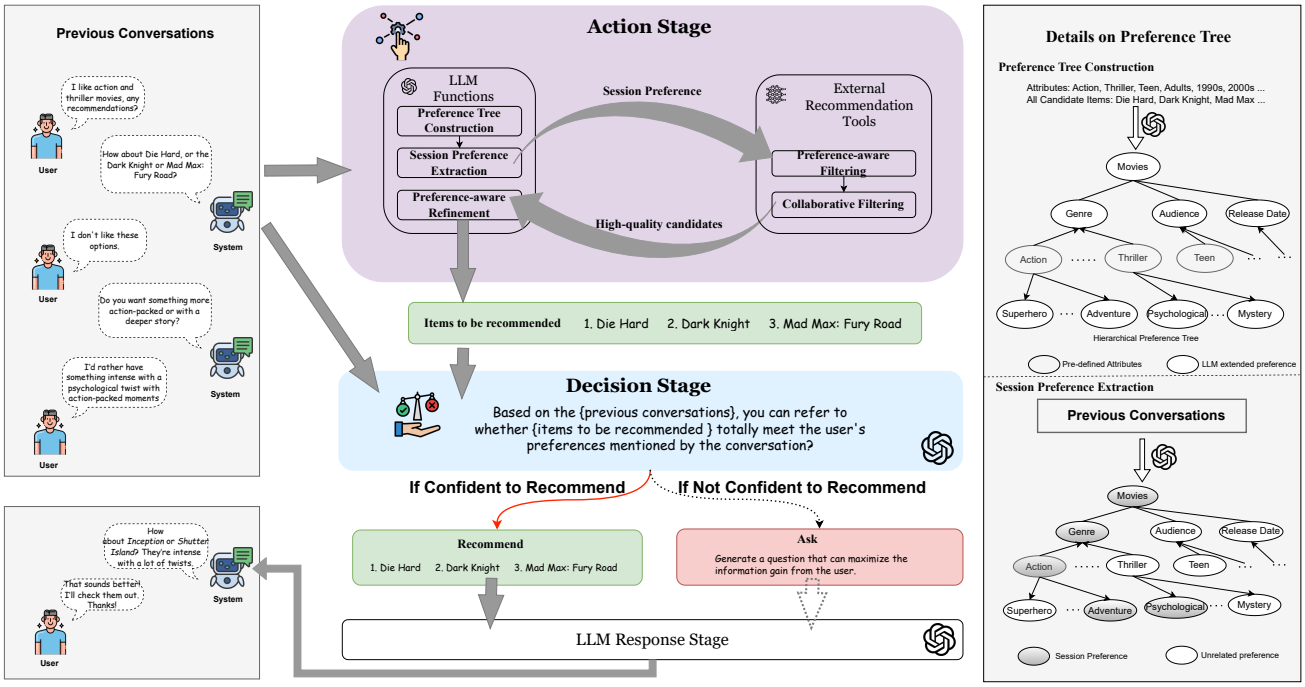

Figure 2: The overall framework of Action First Ineteractive Recommendation System (AF-IRS). The left part shows the conversations between the user and our system. The middle part shows our pipeline. The pipeline begins with the action stage, where items to be recommended are generated, guiding effective decision-making. The right part provides details on preference tree construction and session preference extraction in action stage.

4.1 Action Stage

The action stage aims to provide high-quality recommendations that align with user preferences, which will guide the decision stage in determining the next appropriate action.

This stage faces significant challenges due to the inherent limitations of both external recommendation tools and LLMs. While external recommenders can process large item corpora, they lack the flexibility to adapt to dynamic, text-based user responses, often resulting in recommendations that do not reflect the evolving needs of the user. On the other hand, LLMs excel in understanding text but are constrained by context limitations, making it difficult to handle large item pools effectively. Existing methods only address LLM limitations by relying on a unidirectional flow, where external recommenders provide candidates to the LLM for refinement. However, since external recommenders cannot interpret dynamic textual responses, the candidates they provide tend to be of lower quality, ultimately degrading the overall recommendation quality.

Hence, to generate the most accurate recommendations, we must first interpret the user's needs and select relevant items from a large corpus. Thus, we construct a bidirectional interaction loop between LLMs and external recommendation tools. This allows the LLM's natural language understanding to help interpret the user's session preferences, while the recommendation tools process large item pools to provide a narrowed set of high-quality items that meet the LLM's context limitations.

In the action stage, LLMs pre-construct a hierarchical preference tree to match various granular expressions of user responses. Using this preference tree, LLMs match the user's input from top to bottom, identifying the most fine-grained and information-rich session preferences. These session-specific preferences are then passed to the external recommendation tools, enabling them to better understand the semantic nuances of the user's text-based responses. The recommendation tools perform two levels of filtering: the first level filters based on the session preferences provided by the LLM, applying session-aware filtering. The second level uses the user's historical behavioral data to perform collaborative filtering. This results in a narrowed-down set of high-quality items, which is then returned to the LLMs while staying within the context limits. The LLMs, informed by their understanding of the user's preferences from the conversation, further refine this candidate set. Finally, the LLMs generate the final set of recommended items, which are sent to the decision stage for effective decision-making.

4.1.1 Preference Tree Construction. Many existing methods rely on mapping user responses to predefined attributes to understand preferences. However, this approach has two main drawbacks. First, predefined attribute sets, such as those found in datasets like ML-1M, are often limited in number. Even with accurate mappings, the candidate pool can remain large and imprecise, making it harder to provide highly relevant recommendations. Second, predefined attributes are inherently rigid and may not fully capture the complexity of user preferences. For instance, a user might express a

desire for "something exciting" or "a movie that feels fresh," which is not easily mapped to predefined categories like "action" or "comedy." These types of vague or subjective responses pose challenges for systems that rely on fixed attributes. To address this, we propose using LLMs to construct a **preference tree** based on these pre-defined attributes. This method allows us to capture user preferences at multiple granularities and interpret them more flexibly. By organizing attributes into hierarchical categories, we enable a more nuanced understanding of user responses. LLMs can group attributes into broad categories, then further refine them into more specific subcategories, enhancing the overall recommendation accuracy and making the system more adaptable to diverse user needs.

In detail, let $A = \{a_1, a_2, ..., a_n\}$ represent the set of predefined attributes. We employ LLMs for hierarchical clustering and the prompt template is equipped with these attributes like *"Organize the provided attributes into a hierarchical attribute tree with three levels: broad categories at the first level, meaningful subcategories at the second level, and the attributes as leaf nodes at the third level."*. For instance, in the movie domain as illustrated in the right of Figure 2, attributes such as *action* and *thriller* are grouped into a broader category labeled *genre*. This broader category forms new higher-level nodes within the preference tree.

The LLM then analyzes the descriptions and titles of items under these attributes and clusters them into more specific categories and the prompt template is equipped with items under these attributes like *"Group items under broad attributes and further categorize them into specific subcategories based on shared characteristics."*. For example, movies under the *action* category might be subdivided into *Superhero* (e.g., *The Dark Knight*) and *Adventure* (e.g., *Mad Max: Fury Road*), based on their thematic elements. These broader categories and their resulting subcategories form the hierarchical preference tree, denoted as $\mathcal{T} = \{\mathcal{T}_1, \mathcal{T}_2, ... \ \mathcal{T}_k\}$, where each T_i represents a specific node in the tree. This hierarchical structure helps distinguish items and map them to the correct nodes in the tree. It also provides flexibility in handling varying levels of user specificity and can adapt to new data as it becomes available.

4.1.2 Session Preference Extraction. Once the hierarchical preference tree is constructed, the next step is to map user responses to the tree in order to extract session-specific preferences, which will guide subsequent filtering by the external recommendation tools.

Let U_{response} represent the user's textual response from all previous turns in the conversation. We use LLMs to iteratively map U_{response} to nodes in the preference tree $\mathcal{T}$, which has L levels of nodes, and the LLM prompt template $P_{extract}$ is like *"Given user responses and a set of preference nodes, identify and select the most relevant preference nodes based on the user's expressed preferences. Output the selected nodes in order of relevance"*. At each level l of the tree, the model selects one or more nodes from the available child nodes at that level, and the child nodes become candidates for the next selection. This process continues recursively, progressing from the root node $\mathcal{T}_0$ down to the leaf nodes $\mathcal{T}_L$, which represent the final session preferences.

We define this process as:

$$\mathcal{T}_l = f^l_{\text{LLM}}(U_{response}, \mathcal{T}_{l-1}, P_{extract}) \quad \text{for each } l = 1, 2, \dots, L \tag{1}$$

Where $\mathcal{T}_l$ represents the selected nodes at level l for $U_{response}$

At the final level L, we extract the session-specific preferences:

$$S_{\text{pref}} = \bigcup_{i=1}^{n} \mathcal{T}_L^{(i)} = \{S^1_{\text{pref}}, S^2_{\text{pref}}, \dots\} \tag{2}$$

For example, as shown in Figure 2, the LLM maps a response like "I'd rather have something intense with a psychological twist with action-packed moments" to specific preferences, which can be discretized as {adventure, psychological}. These extracted session preferences are then passed to the external recommendation tools for further processing, a key design in our approach to complement the interaction flow from LLMs to recommendation tools.

4.1.3 Two-stage Filtering in Recommendation Tools. With the session preferences $[S^k_{\text{pref}}, k = 1, 2, 3, \dots]$ provided by LLMs, the external recommendation tools perform a two-stage filtering process called **preference-based filtering** and **collaboration-based filtering**. Firstly, in the preference-based filtering, we extract items that satisfy each session preference S^k_{pref}. Let the set of items corresponding to each preference S^k_{pref} be denoted as C_k. Specifically, for each preference S^k_{pref}, we obtain a candidate set C_k. The first-stage candidate set $\mathscr{C}_{sp}$ is then formed by selecting the intersection of items across all session preferences, i.e.,

$$\mathscr{C}_{sp} = \bigcap_{k=1}^{n} C_k \tag{3}$$

This provides a set of items that align with the user's expressed session preferences. Secondly, in the collaboration-based filtering, recommender systems are pre-trained on users' historical interactions with items and have already learned the users' collaborative preferences. Let $\mathbf{u} \in \mathbb{R}^d$ and $\mathbf{i} \in \mathbb{R}^d$ represent the user and item embeddings, respectively. Using cosine similarity, we compute the relevance between the user and item embeddings to further narrow down the candidate set.

$$\text{sim}(\mathbf{u}, \mathbf{i_j}) = \frac{\mathbf{u} \cdot \mathbf{i_j}}{\|\mathbf{u}\|\|\mathbf{i_j}\|} \tag{4}$$

$$\mathscr{C}_{cp} = \left\{ i_{j_1}, i_{j_2}, \dots, i_{j_k} \mid \forall i_{j_m} \in \mathscr{C}_{sp}, \text{sim}(\mathbf{u}, \mathbf{i_{j_m}}) \geq \text{sim}(\mathbf{u}, \mathbf{i_{j_{m+1}}}) \right\} \tag{5}$$

This two-stage filtering results in a set $\mathscr{C}_{cp}$ that satisfies both the user's session preferences and collaborative preferences. The set $\mathscr{C}_{cp}$ is now sufficiently narrowed down to meet the context limitations of LLMs and is passed back to the LLMs for further refinement.

4.1.4 Preference-aware Refinement. As with existing approaches [21, 28], before making the final recommendations, we allow LLMs to refine the candidate sets provided by recommendation tools using structured user preferences.

Leveraging their natural language understanding abilities, LLMs interpret the user's preferences based on previous conversations C_u and the prompt template P_{pref} is like *"Given the past conversations between the user and our system, infer the user's preferences in a structured way"*. This interpretation results in a structured user preference $Pref_u$, which is used to further refine the candidate set $\mathscr{C}_{\text{cp}}$ and the prompt template P_{refine} is like *"Given the user preferences and candidate items, select the top 10 items that best match the user's preferences"* :

$$Pref_u = f_{\text{LLM}}(C_u, P_{pref}) \tag{6}$$

$$I_{\text{cand}} = f_{\text{LLM}}(\mathscr{C}_{\text{cp}}, Pref_u, P_{refine}) \quad (7)$$

This series of methods in action stage ensures that the final set of recommendations is not only aligned with the user's explicit preferences but also incorporates insights from historical interaction patterns and domain knowledge. As a result, the recommendations are both relevant and enriched, offering a well-rounded and informed list for decision-making.

4.2 Decision Stage

In the decision stage, the primary goal is to determine which items to recommend or what attributes to ask the user in order to refine the recommendations. This decision process is crucial for both effectiveness and efficiency.

The Decision-First paradigm make decisions based solely on previous conversations, lacking insights into future actions. As a result, the module faces a challenge: while the items generated are expected to meet the user's needs, their relevance remains uncertain without further verification. If the decision module is confident in understanding the user's preferences, but the selected items are insufficiently aligned with those needs, the optimal action would be to ask clarifying questions, rather than recommending the items. This highlights a key limitation of the decision-first approach. In contrast, our proposed Action-First paradigm addresses this challenge by enabling the decision stage to be informed of future actions, allowing decisions to be made based on both past interactions and the items selected in the previous action stage.

We introduce an action-aware verification module that evaluates whether the recommended items truly align with the user's actual needs. If the verification confirms the items' relevance, they are recommended to the user. However, if the items are deemed insufficiently relevant, the system triggers the generation of targeted questions to gain additional maximum information.

4.2.1 Action-aware Decision-Making. In this module, the system must determine whether to recommend pre-selected items or ask clarifying questions. While the items from the action stage are typically chosen from a large item corpus based on the user's long-term and dynamic preferences, they may not always truly align with the user's actual needs This misalignment can stem from limitations in the generation module or insufficient information from the user's responses.

To address this, we leverage LLMs' prompt learning capabilities for decision-making. As shown in Figure 3, the model receives two inputs: the items to be recommended and the previous conversations. Using its reasoning abilities, the model evaluates whether the pre-selected items are appropriate for the user.

> DECISION_PROMPT = Based on the past conversations between the user and our system {**previous_conversations**}, and items selected to be recommended for the user {**items_to_be_recommened**}, determine the appropriate next action.
>
> If the recommended items totally meet the user's preferences mentioned by the conversation and are most likely to include the user's target item, proceed to "**recommend items**". Otherwise, choose to "**ask clarification questions**" to better understand the user's needs.

Figure 3: The Prompt of Action-aware Decision-Making.

If the model determines that the user's preferences are clear and the pre-selected items I_{cand} align with the target, it proceeds with the recommendation. Otherwise, it triggers the response "ask clarification questions," indicating that the pre-selected items are too similar or the user's information is insufficient. This action-aware verification approach ensures that only the most relevant items are recommended, while low-confidence situations prompt the system to request more information.

4.2.2 Information-Maximizing Question Generation. When the action is to recommend, we directly output the pre-selected items, which have been ensured that they align with the user's preferences, making the recommendation reliable. However, if the decision is to ask for clarification, we leverage the decoding abilities of LLMs to generate a targeted and information-maximizing question. To do so, we take into account two key factors:

- **User Preferences** $Pref_u$: The current user preferences, interpreted through the dialogue, as shown in Equation (6).
- **Item Distribution**: The distribution of candidate items that match the current preferences, which helps identify the distinguishing factors between them.

By inputting these two factors into the LLM, we generate a clarifying question that maximizes information gain. The aim is to minimize the number of interactions required to refine the user's preferences, ensuring that each question reduces ambiguity and improves recommendation accuracy. This approach can minimize interaction turns, enhancing the overall user experience and streamlining the recommendation process.

5 Experiments

In this section, we evaluate our method on three interactive recommender benchmarks. To gain deeper insights into the performance of our method, we address the following research questions:

- **RQ1:** How does our method compare with state-of-the-art LLM-based interactive recommender systems?
- **RQ2:** What impact do the various modules of our system have on its overall performance?
- **RQ3:** Does our Action-First paradigm improve decision-making compared to other baselines?
- **RQ4:** How does our bidirectional interaction loop in the action stage enhance overall performance?

5.1 Experimental Setup

5.1.1 Datasets. We evaluate our method on three interactive recommender benchmarks: **ML-1M**[1], and **Steam**[2], and **Amazon-Beauty**[3], each representing different domains. The dataset details are provided in Table 1.

- **ML-1M** is a widely used recommendation dataset containing 1 million ratings on movies from around 6,000 users.
- **Steam** consists of user ratings for games on Steam, a major online video game distribution platform.

[1]https://grouplens.org/datasets/movielens/1m/
[2]https://github.com/kang205/SASRec
[3]http://jmcauley.ucsd.edu/data/amazon/links.html

Table 1: Dataset Statistics

Dataset	#Users	#Items	#Interactions
ML-1M	6,038	3,307	835,789
Steam	281,204	11,961	3,484,497
Amazon-Beauty	15,576	8,678	139,318

- **Amazon-Beauty** represents the "Beauty" category from the Amazon Review dataset, focusing on beauty products, chosen for its high sparsity and variability in user interactions.

For consistency, the preprocessing of all three datasets follows the same procedure: interactions with ratings less than 3 are excluded, and users or items with fewer than 10 interactions are filtered out. Each item in these datasets is also associated with a set of tags. Following prior work [14, 35], we adopt the leave-one-out evaluation strategy. Specifically, the most recent interaction of a user is treated as the test instance, the second most recent as the validation instance, and all prior interactions are used for training.

5.1.2 User Simulator for IRS. Due to the interactive nature of interactive recommender systems, training and evaluation require simulating user interactions. Following [14, 32], we utilize a Large Language Model (LLM) with a single prompt, as shown in Figure 4 as a user simulator to generate human-like responses. For each observed user-item interaction pair (u, i), we simulate a conversation session, treating item i as the ground truth target with a set of attributes A_i and description Des_i. In this setup, the user's preferences are anchored around item i, and relevant information about i is fed into the LLM-based user simulator.

The session begins with the simulated user expressing preferences based on the target item's details. The system then decides whether to ask clarification questions or recommend other items. If the recommended items include i, the user accepts; otherwise, the recommendation is rejected. The session terminates when the user either accepts the recommendation or ends the interaction. The maximum turn is set as 5.

```
USER_SIMULATOR_PROMPT =
You are a user chatting with a recommender for {domain} recommendation in turn.
Your history is {history}. Your target items: {target}.
Here is the information about target you could use: {target_item_info}.

You must follow the instructions below during chat:
If the recommender recommends {target}, you should accept.
If there is no {target} in the recommended items, you should refuse.
If the recommender asks any questions for your preference, you should give the
corresponding answers about {target}.
You should never directly tell the target item title.

Now lets start, you first, act as a user.
Your output is only allowed to be the words from the user you act.
If you think the conversation comes to an ending, output a <END>.
```

Figure 4: The prompt of LLM-based user simulator.

5.1.3 Baseline Methods. To validate our method's effectiveness, we compare it with several strong baselines in the interactive recommender setting and we categorize them into two groups: (1) Methods without external recommendation tools include:

- **Zero-shot** denotes a method that uses a widely-used LLM-based API to directly make decisions and execute actions.
- **ReAct** [46] combines reasoning and action, using chain-of-thought reasoning for decision-making.
- **ChatCRS** [21] uses an external Knowledge Base to decide the next action and generates the response based on it.
- **RAH** [28] emphasizes extracting user preferences and making decisions based on them.

(2) Methods with external recommendation tools include:

- **LLMCRS** [9] leverages LLMs' internal reasoning abilities to determine actions and invoke external tools.
- **InteRecAgent** [14] uses dynamic demonstrations to augment LLMs for planning external tool usage.
- **MACRec** [36] employs multi-agent collaboration, with a manager agent coordinating interactions.

All baselines are LLM-based. For fair comparison, we use a widely used API as the backbone LLM and use SASRec as the same external recommendation tools.

5.1.4 Evaluation Metrics. Following previous studies on interactive recommender systems [14, 17], we evaluate the performance of each IRS using two key metrics: success rate at turn t (SR@t) and average turns (AT). SR@t(t=1,3,5) represents the proportion of successful sessions by turn t within the test set, where a session is considered successful if the user accepts the recommendation. AT denotes the average number of turns across all test sessions. A higher SR@t indicates better recommendation accuracy at a specific turn, while a lower AT reflects greater efficiency in providing accurate recommendations with fewer interactions.

5.1.5 Implementation Details. We use a widely used LLM API as our backbone model due to its strong reasoning and natural language understanding capabilities. ReAct, ChatCRS, and RAH also use the same API as their backbone for directly generating recommendations, with a temperature parameter set to 0. LLMCRS, InteRecAgent, MACRec, and our model utilize a sequential recommender as the external conventional recommender. We select SASRec [15] because it is widely regarded as a representative model [14, 35]. We also train SASRec on three benchmark datasets, with training details following [18]. The number of candidates sent by SASRec to LLM is set to 50. InteRecAgent and MACRec also incorporate a query tool to interpret user responses and help conventional recommenders filter candidates, using SQL as described in [14].

5.2 Overall Performance(RQ1)

Table 2 presents a comparison of our model, AF-IRS, with other baselines across three benchmark datasets. The following observations can be made:

- AF-IFS significantly outperforms all other baselines in terms of evaluation metrics, including higher success rates and

Table 2: Overall Performance of different methods on ML-1M, Amazon-Beauty and Steam. The best result is in bold, while the second-best is in underline. The symbol * indicates statistically significant improvement over the best baseline with $p < 0.01$.

Dataset	ML-1M				Amazon-Beauty				Steam			
Method	**ST@1**	**ST@3**	**ST@5**	**AT**	**ST@1**	**ST@3**	**ST@5**	**AT**	**ST@1**	**ST@3**	**ST@5**	**AT**
Zero-shot	0.06	0.46	0.71	3.87	0.00	0.08	0.15	5.61	0.22	0.57	0.67	3.51
ReAct	0.05	0.44	0.68	4.03	0.00	0.10	0.15	5.59	0.25	0.52	0.64	3.55
ChatCRS	0.05	0.52	0.79	3.72	0.07	0.26	0.35	4.86	0.12	0.41	0.57	4.09
RAH	0.22	0.55	0.74	3.40	0.13	0.22	0.34	4.84	0.34	0.59	0.68	3.17
LLMCRS	0.16	0.53	0.67	3.73	0.15	0.22	0.30	4.92	0.23	0.52	0.64	3.55
InteRecAgent	0.10	0.24	0.41	4.76	0.23	0.44	0.54	3.93	0.15	0.37	0.48	4.31
MACRec	0.04	0.19	0.46	4.89	0.20	0.40	0.52	4.09	0.19	0.39	0.61	4.00
Ours	**0.27***	**0.68***	**0.81***	**3.13***	**0.29***	**0.52***	**0.65***	**3.50***	**0.52***	**0.78***	**0.88***	**2.27***

lower average turns across all datasets. Specifically, compared to the second-best baseline, our method shows notable improvements on all three datasets. For example, in terms of SR@5, AF-IFS improves by +2.53% for ML-1M, +20.37% for AMZ-Beauty, and +29.41% for Steam. In terms of AT, AF-IFS reduces the average turns by 7.9% for ML-1M, 10.9% for AMZ-Beauty, and 28.4% for Steam. These results demonstrate that our method provides more accurate and efficient recommendations in interactive recommendation.

- External tools are crucial in limited-knowledge domains. In domains like beauty products, where LLMs lack detailed understanding, models such as ChatCRS and RAH perform well in areas like movies and games due to the widely used API's extensive world knowledge. However, in more specialized domains, performance is weaker. Models like InteRecAgent and MACRec, which use external tools, perform better, and our model also shows significant improvements in beauty domain.
- However, methods that rely on simple external filtering tools, such as SQL (e.g., InteRecAgent), often struggle to handle complex textual user preferences in conversations. These tools typically map user inputs to pre-defined attributes, which is particularly limiting in datasets with sparse attributes like ML-1M. As a result, many irrelevant candidates remain even after filtering, leading to suboptimal recommendations. In contrast, our approach builds a preference tree to extract finer-grained session preferences, enabling more precise filtering. Coupled with a bidirectional interaction loop between the LLM and external tools, our model achieves better generalization and performance across diverse datasets.

5.3 Ablation Studies(RQ2)

To assess the effectiveness of our proposed Action-First paradigm in decision stage and bidirectional interaction loop in action stage, we conduct ablation experiments by testing various model configurations. We design three variants to better understand the contribution of each module:

- **w/o Action-First:** This variant follows the Decision-First paradigm. The model first decides the action (e.g., recommend or ask) using the LLMs. The action module is same as our Action-First paradigm.
- **w/o LLM2RS:** In this version, we omit the interaction flow from LLMs to the external recommender. The LLMs do not extract session preferences to send to the RS, which can only process the entire item corpus.
- **w/o LLM-refine:** We omit LLM's preference-aware refinement and the final recommendation are the items provided by the external recommender.

As shown in the results, our full model outperforms all variants across all metrics, confirming the effectiveness of each individual module. First, **w/o Action-First** demonstrates a noticeable performance drop, indicating the difficulty of relying solely on past dialogue to determine the action. By being informed of items to be recommended, our decision module gains a more comprehensive global perspective, leading to improved performance. Regarding the action stage, the bidirectional information flow proves crucial. **w/o LLM2RS** demonstrates that LLMs play a significant role in helping conventional recommenders interpret textual-based user preferences. **w/o LLM-refine** underscores that the importance of LLM's ability to refine recommendations based on the extracted user preferences, further integrating dynamic user needs. With the bidirectional flow, our model significantly improves recommendation accuracy and overall performance.

5.4 In-depth Analysis(RQ3 & RQ4)

5.4.1 Decision Stage(RQ3). This section explores whether our Action-First paradigm leads to better decision-making compared to other baselines. An ideal decision module should choose to recommend when user preferences are clear, i.e., when the target item is present in the upcoming recommendation list. If user preferences are unclear, the system should choose to ask for more information. To assess the decision-making ability, we introduce a new metric: **Decision Success Rate (DSR@t)**. DSR@t measures the proportion of correct decisions (either recommend or ask) within the first t turns, where t=5 in our case. A correct decision occurs when the system recommends if target item is presented in the recommended list, and triggers an ask when the target item is absent.

Table 3: Ablation studies of different variants on ML-1M, Amazon-Beauty and Steam.

Dataset	ML-1M				Amazon-Beauty				Steam			
Variants	ST@1	ST@3	ST@5	AT	ST@1	ST@3	ST@5	AT	ST@1	ST@3	ST@5	AT
Ours	**0.27**	**0.68**	**0.81**	**3.13**	**0.29**	**0.52**	**0.65**	**3.50**	**0.52**	**0.78**	**0.88**	**2.27**
w/o Action-First	0.24	0.57	0.75	3.47	0.17	0.46	0.47	4.40	0.37	0.67	0.73	2.95
w/o LLM2RS	0.22	0.54	0.66	3.64	0.12	0.25	0.32	4.85	0.36	0.61	0.68	3.18
w/o LLM-refine	0.06	0.26	0.36	4.84	0.09	0.23	0.37	5.02	0.28	0.49	0.56	3.74

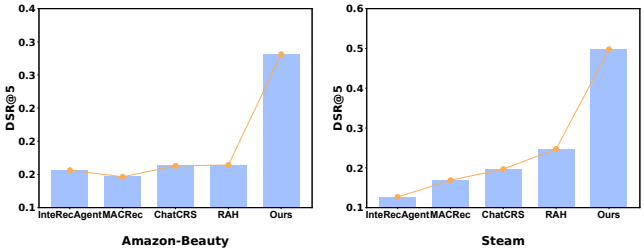

Figure 5: Decision Sucess Rate of different models

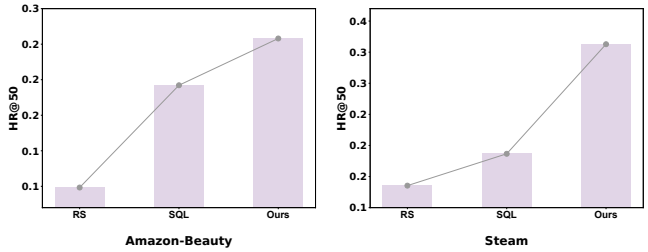

Figure 6: Hit Rate of candidates sent from recommendation tools to LLMs

We compare our model against the top-performing baselines for each dataset. From Figure 5, we observe that our model's decision success rate significantly outperforms other baselines, demonstrating the advantage of our Action-First paradigm over the Decision-First paradigm. While models like ChatCRS and RAH use external knowledge bases and user preferences to assist decision-making, they still lack insight into future recommendations. In contrast, our Action-First paradigm provides the LLMs with more information, enabling better decision-making, which results in higher recommendation success rates and fewer conversation turns.

5.4.2 Action Stage(RQ4). This section aims to examine the impact of our bidirectional interaction loop. Due to the limitations of LLMs in terms of context and domain expertise, external recommendation tools are necessary to provide candidates for LLM refinement. The key here is that the candidates provided by the recommendation tools should ideally contain the target item so that the LLM can identify and recommend it correctly. Therefore, we introduce a metric called **Hit Rate@50**, which calculates the probability that the candidates sent from the recommendation tools to the LLM include the target item. This metric helps us understand how our bidirectional interaction loop improves overall performance.

Our model is compared against two scenarios which include all baselines except without external recommendation tools. (1) The first scenario uses **traditional recommendation tools** to select the top 50 candidates from the entire item pool, without considering user preferences expressed in natural language(*e.g.*, LLMCRS). (2) The second scenario follows **InteRecAgent**'s approach, where SQL is used to match user responses with predefined attributes, narrowing down the candidate pool before the recommendation system selects the top 50 candidates(*e.g.*, InteRecAgent, MACRec). (3) Our approach introduces a **bidirectional interaction loop**, where LLMs extract session-specific preferences and pass them to the RS. The RS then filters candidates based on these preferences and selects the top 50 using collaborative filtering, providing a more personalized and context-aware recommendation.

As shown in Figure 6, our model outperforms others by generating candidate sets with a significantly higher probability of containing the target item. This underscores the importance of constructing an effective information flow from LLMs to recommendation tools. The preference tree built by the LLM captures finer-grained, personalized user preferences, improving user representation. Consequently, the first-stage filtering yields fewer but more accurate candidates, while the second-stage collaborative filtering further refines the selection, boosting the likelihood of including the target item and enhancing overall recommendation performance.

6 Conclusion

In this work, we identify two key information gaps in Interactive Recommender Systems (IRSs): insufficient information in decision-making and unidirectional information flow during recommendation. To address these gaps, we propose AF-IRS, which leverages the Action-First paradigm informed by future recommendations and a bidirectional interaction loop to enhance the interpretation of user preferences. Our experiments on three benchmark datasets demonstrate that AF-IRS outperforms baselines, achieving higher success rates and fewer average turns.

Acknowledgments

The Shanghai Jiao Tong University team is supported by National Key R&D Program of China (2022ZD0114804), Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102) and National Natural Science Foundation of China (624B2096, 62322603, 62177033). The work is also sponsored by Huawei Innovation Research Program. We thank MindSpore [1] for its partial support.

References

[1] 2020. MindSpore. https://www.mindspore.cn/

[2] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 1007–1014.

[3] Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. 2019. Towards Knowledge-Based Recommender Dialog System. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan (Eds.). Association for Computational Linguistics, Hong Kong, China, 1803–1813. doi:10.18653/v1/D19-1189

[4] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards Conversational Recommender Systems *(KDD '16)*. Association for Computing Machinery, New York, NY, USA, 815–824. doi:10.1145/2939672.2939746

[5] Huy Dao, Yang Deng, Dung D. Le, and Lizi Liao. 2024. Broadening the View: Demonstration-augmented Prompt Learning for Conversational Recommendation *(SIGIR '24)*. Association for Computing Machinery, New York, NY, USA, 785–795. doi:10.1145/3626772.3657755

[6] Yang Deng, Yaliang Li, Fei Sun, Bolin Ding, and Wai Lam. 2021. Unified Conversational Recommendation Policy Learning via Graph-based Reinforcement Learning. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*. ACM, 1431–1441.

[7] Yang Deng, Yaliang Li, Fei Sun, Bolin Ding, and Wai Lam. 2021. Unified conversational recommendation policy learning via graph-based reinforcement learning. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1431–1441.

[8] Yang Deng, Lizi Liao, Zhonghua Zheng, Grace Hui Yang, and Tat-Seng Chua. 2024. Towards human-centered proactive conversational agents. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 807–818.

[9] Yue Feng, Shuchang Liu, Zhenghai Xue, Qingpeng Cai, Lantao Hu, Peng Jiang, Kun Gai, and Fei Sun. 2023. A large language model enhanced conversational recommender system. *arXiv preprint arXiv:2308.06212* (2023).

[10] Luke Friedman, Sameer Ahuja, David Allen, Zhenning Tan, Hakim Sidahmed, Changbo Long, Jun Xie, Gabriel Schubiner, Ajay Patel, Harsh Lara, et al. 2023. Leveraging large language models in conversational recommender systems. *arXiv preprint arXiv:2305.07961* (2023).

[11] Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and challenges in conversational recommender systems: A survey. *AI open* 2 (2021), 100–126.

[12] Zhankui He, Zhouhang Xie, Rahul Jha, Harald Steck, Dawen Liang, Yesu Feng, Bodhisattwa Prasad Majumder, Nathan Kallus, and Julian McAuley. 2023. Large language models as zero-shot conversational recommenders. In *Proceedings of the 32nd ACM international conference on information and knowledge management*. 720–730.

[13] Chenhao Hu, Shuhua Huang, Yansen Zhang, and Yubao Liu. 2022. Learning to Infer User Implicit Preference in Conversational Recommendation *(SIGIR '22)*. Association for Computing Machinery, New York, NY, USA, 256–266. doi:10.1145/3477495.3531844

[14] Xu Huang, Jianxun Lian, Yuxuan Lei, Jing Yao, Defu Lian, and Xing Xie. 2023. Recommender ai agent: Integrating large language models for interactive recommendations. *arXiv preprint arXiv:2308.16505* (2023).

[15] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE international conference on data mining (ICDM)*. IEEE, 197–206.

[16] Sara Kemper, Justin Cui, Kai Dicarlantonio, Kathy Lin, Danjie Tang, Anton Korikov, and Scott Sanner. 2024. Retrieval-Augmented Conversational Recommendation with Prompt-based Semi-Structured Natural Language State Tracking. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2786–2790.

[17] Heeseon Kim, Hyeongjun Yang, and Kyong-Ho Lee. 2023. Confident Action Decision via Hierarchical Policy Learning for Conversational Recommendation *(WWW '23)*. Association for Computing Machinery, New York, NY, USA, 1386–1395. doi:10.1145/3543507.3583536

[18] Anton Klenitskiy and Alexey Vasilev. 2023. Turning Dross Into Gold Loss: is BERT4Rec really better than SASRec?. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 1120–1125.

[19] Wenqiang Lei, Xiangnan He, Yisong Miao, Qingyun Wu, Richang Hong, Min-Yen Kan, and Tat-Seng Chua. 2020. Estimation-action-reflection: Towards deep interaction between conversational and recommender systems. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 304–312.

[20] Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. 2020. Interactive path reasoning on graph for conversational recommendation. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 2073–2083.

[21] Chuang Li, Yang Deng, Hengchang Hu, Min-Yen Kan, and Haizhou Li. 2024. Incorporating External Knowledge and Goal Guidance for LLM-based Conversational Recommender Systems. *arXiv preprint arXiv:2405.01868* (2024).

[22] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. *Advances in neural information processing systems* 31 (2018).

[23] Shuokai Li, Ruobing Xie, Yongchun Zhu, Xiang Ao, Fuzhen Zhuang, and Qing He. 2022. User-centric conversational recommendation with multi-aspect user modeling. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 223–233.

[24] Jianghao Lin, Bo Chen, Hangyu Wang, Yunjia Xi, Yanru Qu, Xinyi Dai, Kangning Zhang, Ruiming Tang, Yong Yu, and Weinan Zhang. 2024. Clickprompt: CTR models are strong prompt generators for adapting language models to CTR prediction. In *Proceedings of the ACM Web Conference 2024*. 3319–3330.

[25] Jianghao Lin, Xinyi Dai, Yunjia Xi, Weiwen Liu, Bo Chen, Hao Zhang, Yong Liu, Chuhan Wu, Xiangyang Li, Chenxu Zhu, et al. 2023. How can recommender systems benefit from large language models: A survey. *arXiv preprint arXiv:2306.05817* (2023).

[26] Jianghao Lin, Yanru Qu, Wei Guo, Xinyi Dai, Ruiming Tang, Yong Yu, and Weinan Zhang. 2023. Map: A model-agnostic pretraining framework for click-through rate prediction. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1384–1395.

[27] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2024).

[28] Yubo Shu, Hansu Gu, Peng Zhang, Haonan Zhang, Tun Lu, Dongsheng Li, and Ning Gu. 2023. Rah! recsys-assistant-human: A human-central recommendation framework with large language models. *arXiv preprint arXiv:2308.09904* (2023).

[29] Ruixuan Sun, Xinyi Li, Avinash Akella, and Joseph A Konstan. 2024. Large Language Models as Conversational Movie Recommenders: A User Study. *arXiv preprint arXiv:2404.19093* (2024).

[30] Lingzhi Wang, Huang Hu, Lei Sha, Can Xu, Kam-Fai Wong, and Daxin Jiang. 2021. RecInDial: A unified framework for conversational recommendation with pretrained language models. *arXiv preprint arXiv:2110.07477* (2021).

[31] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A survey on large language model based autonomous agents. *Frontiers of Computer Science* 18, 6 (2024), 186345.

[32] Xiaolei Wang, Xinyu Tang, Xin Zhao, Jingyuan Wang, and Ji-Rong Wen. 2023. Rethinking the Evaluation for Conversational Recommendation in the Era of Large Language Models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 10052–10065. doi:10.18653/v1/2023.emnlp-main.621

[33] Xiaolei Wang, Kun Zhou, Xinyu Tang, Wayne Xin Zhao, Fan Pan, Zhao Cao, and Ji-Rong Wen. 2023. Improving conversational recommendation systems via counterfactual data simulation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2398–2408.

[34] Xiaolei Wang, Kun Zhou, Ji-Rong Wen, and Wayne Xin Zhao. 2022. Towards unified conversational recommender systems via knowledge-enhanced prompt learning. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1929–1937.

[35] Yancheng Wang, Ziyan Jiang, Zheng Chen, Fan Yang, Yingxue Zhou, Eunah Cho, Xing Fan, Xiaojiang Huang, Yanbin Lu, and Yingzhen Yang. 2023. Recmind: Large language model powered agent for recommendation. *arXiv preprint arXiv:2308.14296* (2023).

[36] Zhefan Wang, Yuanqing Yu, Wendi Zheng, Weizhi Ma, and Min Zhang. 2024. Multi-Agent Collaboration Framework for Recommender Systems. *arXiv preprint arXiv:2402.15235* (2024).

[37] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* 35 (2022), 24824–24837.

[38] Yunjia Xi, Weiwen Liu, Jianghao Lin, Xiaoling Cai, Hong Zhu, Jieming Zhu, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. 2024. Towards open-world recommendation with knowledge augmentation from large language models. In *Proceedings of the 18th ACM Conference on Recommender Systems*. 12–22.

[39] Yunjia Xi, Weiwen Liu, Jianghao Lin, Bo Chen, Ruiming Tang, Weinan Zhang, and Yong Yu. 2024. MemoCRS: Memory-enhanced Sequential Conversational Recommender Systems with Large Language Models. *arXiv preprint arXiv:2407.04960* (2024).

[40] Yunjia Xi, Weiwen Liu, Jianghao Lin, Muyan Weng, Xiaoling Cai, Hong Zhu, Jieming Zhu, Bo Chen, Ruiming Tang, Yong Yu, et al. 2024. Efficient and deployable knowledge infusion for open-world recommendations via large language models. *ACM Transactions on Recommender Systems* (2024).

[41] Yunjia Xi, Hangyu Wang, Bo Chen, Jianghao Lin, Menghui Zhu, Weiwen Liu, Ruiming Tang, Weinan Zhang, and Yong Yu. 2024. A decoding acceleration framework for industrial deployable LLM-based recommender systems. *arXiv*

preprint arXiv:2408.05676 (2024).

[42] Bowen Yang, Cong Han, Yu Li, Lei Zuo, and Zhou Yu. 2022. Improving Conversational Recommendation Systems' Quality with Context-Aware Item Meta-Information. In *Findings of the Association for Computational Linguistics: NAACL 2022*, Marine Carpuat, Marie-Catherine de Marneffe, and Ivan Vladimir Meza Ruiz (Eds.). Association for Computational Linguistics, Seattle, United States, 38–48. doi:10.18653/v1/2022.findings-naacl.4

[43] Dayu Yang, Fumian Chen, and Hui Fang. 2024. Behavior Alignment: A New Perspective of Evaluating LLM-based Conversational Recommendation Systems. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2286–2290.

[44] Ting Yang and Li Chen. 2024. Unleashing the Retrieval Potential of Large Language Models in Conversational Recommender Systems. In *Proceedings of the 18th ACM Conference on Recommender Systems* (Bari, Italy) *(RecSys '24)*. Association for Computing Machinery, New York, NY, USA, 43–52. doi:10.1145/3640457.3688146

[45] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2024. Tree of thoughts: Deliberate problem solving with large language models. *Advances in Neural Information Processing Systems* 36 (2024).

[46] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629* (2022).

[47] Se-eun Yoon, Zhankui He, Jessica Maria Echterhoff, and Julian McAuley. 2024. Evaluating Large Language Models as Generative User Simulators for Conversational Recommendation. *arXiv preprint arXiv:2403.09738* (2024).

[48] An Zhang, Yuxin Chen, Leheng Sheng, Xiang Wang, and Tat-Seng Chua. 2024. On generative agents in recommendation. In *Proceedings of the 47th international ACM SIGIR conference on research and development in Information Retrieval*. 1807–1817.

[49] Gangyi Zhang. 2023. User-Centric Conversational Recommendation: Adapting the Need of User with Large Language Models. In *Proceedings of the 17th ACM Conference on Recommender Systems* (Singapore, Singapore) *(RecSys '23)*. Association for Computing Machinery, New York, NY, USA, 1349–1354. doi:10.1145/3604915.3608885

[50] Gangyi Zhang, Chongming Gao, Hang Pan, Runzhe Teng, and Ruizhe Li. 2024. Reformulating Conversational Recommender Systems as Tri-Phase Offline Policy Learning. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 3135–3144.

[51] Weinan Zhang, Junwei Liao, Ning Li, Kounianhua Du, and Jianghao Lin. 2024. Agentic information retrieval. *arXiv preprint arXiv:2410.09713* (2024).

[52] Xiaoyu Zhang, Xin Xin, Dongdong Li, Wenxuan Liu, Pengjie Ren, Zhumin Chen, Jun Ma, and Zhaochun Ren. 2023. Variational reasoning over incomplete knowledge graphs for conversational recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*. 231–239.

[53] Yiming Zhang, Lingfei Wu, Qi Shen, Yitong Pang, Zhihua Wei, Fangli Xu, Bo Long, and Jian Pei. 2022. Multiple Choice Questions based Multi-Interest Policy Learning for Conversational Recommendation. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 2153–2162. doi:10.1145/3485447.3512088

[54] Zeyuan Zhang, Tanmay Laud, Zihang He, Xiaojie Chen, Xinshuang Liu, Zhouhang Xie, Julian McAuley, and Zhankui He. 2024. RecWizard: A Toolkit for Conversational Recommendation with Modular, Portable Models and Interactive User Interface. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 23856–23858.

[55] Rongmei Zhao, Shenggen Ju, Jian Peng, Ning Yang, Fanli Yan, and Siyu Sun. 2022. Two-Level Graph Path Reasoning for Conversational Recommendation with User Realistic Preference *(CIKM '22)*. Association for Computing Machinery, New York, NY, USA, 2701–2710. doi:10.1145/3511808.3557482

[56] Sen Zhao, Wei Wei, Xian-Ling Mao, Shuai Zhu, Minghui Yang, Zujie Wen, Dangyang Chen, and Feida Zhu. 2023. Multi-view Hypergraph Contrastive Policy Learning for Conversational Recommendation *(SIGIR '23)*. Association for Computing Machinery, New York, NY, USA, 654–664. doi:10.1145/3539618.3591737

[57] Kun Zhou, Wayne Xin Zhao, Shuqing Bian, Yuanhang Zhou, Ji-Rong Wen, and Jingsong Yu. 2020. Improving conversational recommender systems via knowledge graph based semantic fusion. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*. 1006–1014.

[58] Kun Zhou, Yuanhang Zhou, Wayne Xin Zhao, Xiaoke Wang, and Ji-Rong Wen. 2020. Towards Topic-Guided Conversational Recommender System. In *Proceedings of the 28th International Conference on Computational Linguistics*, Donia Scott, Nuria Bel, and Chengqing Zong (Eds.). International Committee on Computational Linguistics, Barcelona, Spain (Online), 4128–4139. doi:10.18653/v1/2020.coling-main.365

[59] Lixi Zhu, Xiaowen Huang, and Jitao Sang. 2024. How Reliable is Your Simulator? Analysis on the Limitations of Current LLM-based User Simulators for Conversational Recommendation. In *Companion Proceedings of the ACM on Web Conference 2024*. 1726–1732.

[60] Lixi Zhu, Xiaowen Huang, and Jitao Sang. 2024. A LLM-based Controllable, Scalable, Human-Involved User Simulator Framework for Conversational Recommender Systems. *arXiv preprint arXiv:2405.08035* (2024).

[61] Jie Zou, Aixin Sun, Cheng Long, and Evangelos Kanoulas. 2024. Knowledge-Enhanced Conversational Recommendation via Transformer-Based Sequential Modeling. *ACM Transactions on Information Systems* 42, 6 (2024), 1–27.