Search-Based Interaction For Conversation Recommendation via Generative Reward Model Based Simulated User

Xiaolei Wang
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
wxl1999@foxmail.com

Chunxuan Xia
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
xiachunxuan@ruc.edu.cn

Junyi Li
Department of Computer Science, National University of Singapore
Beijing, Singapore
junyi_cs@nus.edu.sg

Fanzhe Meng
School of Information and Software Engineering, University of Electronic Science and Technology of China
Chengdu, China
mengfanzhe16@gmail.com

Lei Huang
Meituan
Beijing, China
huanglei45@meituan.com

Jinpeng Wang
Meituan
Beijing, China
wangjinpeng04@meituan.com

Wayne Xin Zhao[✉]
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
batmanfly@gmail.com

Ji-Rong Wen
Gaoling School of Artificial Intelligence, Renmin University of China
Beijing, China
jrwen@ruc.edu.cn

Abstract

Conversational recommendation systems (CRSs) use multi-turn interaction to capture user preferences and provide personalized recommendations. A fundamental challenge in CRSs lies in effectively understanding user preferences from conversations. Previous research primarily focuses on the issue of insufficient contextual information in conversations. They address this by introducing external knowledge sources, such as knowledge graphs, large language models (LLMs), and conversational recommendation corpora. Based on this, they design specific alignment strategies (*e.g.,* prompt learning and instruction tuning) to integrate such knowledge for user preference understanding and item recommendation. However, user preferences can be multifaceted and complex, posing significant challenges for accurate recommendations even with access to abundant external knowledge. While interaction with users can clarify their true preferences, frequent user involvement may lead to a degraded user experience.

To address this problem, we propose a **G**enerative **R**eward model based **S**imulated **U**ser, named **GRSU**, for *automatic* interaction with CRSs. The simulated user provides feedback to the items recommended by CRSs, enabling them to better capture intricate user preferences through multi-turn interaction. Inspired by generative reward models, we design two types of feedback actions for the simulated user: *i.e., generative item scoring*, which offers coarse-grained feedback, and *attribute-based item critiquing*, which provides fine-grained feedback. To ensure seamless integration, these feedback actions are unified into an *instruction*-based format, allowing the development of a unified simulated user via instruction tuning on synthesized data. With this simulated user, *automatic* multi-turn interaction with CRSs can be effectively conducted. Furthermore, to strike a balance between effectiveness and efficiency, we draw inspiration from the paradigm of *reward-guided search* in complex reasoning tasks and employ beam search for the interaction process. On top of this, we propose an *efficient* candidate ranking method to improve the recommendation results derived from interaction. Extensive experiments on public datasets demonstrate the effectiveness, efficiency, and transferability of our approach.

CCS Concepts

• **Information systems** → **Recommender systems**; **Users and interactive retrieval**.

Keywords

Conversational Recommendation; Simulated User; Reward Model

ACM Reference Format:
Xiaolei Wang, Chunxuan Xia, Junyi Li, Fanzhe Meng, Lei Huang, Jinpeng Wang, Wayne Xin Zhao[✉], and Ji-Rong Wen. 2025. Search-Based Interaction For Conversation Recommendation via Generative Reward Model Based Simulated User. In *Proceedings of the 48th International ACM SIGIR*

[✉] Corresponding author.

SIGIR '25, July 13–18, 2025, Padua, Italy
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-1592-1/2025/07
https://doi.org/10.1145/3726302.3730080

Conference on Research and Development in Information Retrieval (SIGIR '25), July 13–18, 2025, Padua, Italy. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3726302.3730080

1 Introduction

Conversational recommender systems (CRSs) [11, 17] aim to provide personalized recommendations through multi-turn conversations. Typically, a CRS consists of two components: a *recommender* component that provides recommendations based on the user preference from the conversation context, and a *conversation* component that generates responses based on the conversation context and item recommendations from the recommender component.

To develop an effective CRS, it is crucial to understand user preferences from the conversation context. A primary challenge lies in the brevity of typical conversations, which often consist of only a few sentences and lack sufficient contextual information for accurate user preference understanding. To tackle this issue, prior work introduces external knowledge sources, such as knowledge graphs [8, 23, 39], large language models (LLMs) [13, 35], and conversational recommendation corpora [8, 33]. Building on these resources, they design specialized alignment strategies, such as prompt learning [8] and instruction tuning [35], to integrate the additional knowledge for improved user preference understanding and item recommendation. However, recent work [30] reveals that user preferences are often multifaceted and complex, making accurate recommendations challenging even with enriched knowledge. For instance, as illustrated in Table 1, a user prefers both "horror" and "comedy" movies. However, the CRS considers only "horror" in its initial attempt, resulting in a suboptimal recommendation. To address this problem, NBCRS [33] proposes a neighborhood-based approach. It retrieves conversations similar to the current one and utilizes item frequencies from these similar conversations to generate recommendations. While NBCRS can handle the complexity of user preferences by leveraging collective patterns, its dependence on similar conversations limits its ability to generalize to novel user preferences. At its core, this problem stems from the *single-turn* nature of existing CRSs, which generate recommendations based solely on the conversation history. Without ongoing interaction with users, CRSs cannot fully discern their true preference from the conversation history. However, frequent user involvement may negatively impact their experience [18]. Similarly, recent studies on the evaluation of CRSs also highlight the limitations of single-turn interactions [15, 30, 43]. They propose interactive evaluation methods and develop LLM-based simulated users to reduce the need for the involvement of real users. Nevertheless, these simulated users depend on ground-truth user preferences, which are unavailable at inference time. How to develop simulated users in the absence of ground-truth preference labels remains a significant challenge.

To address the aforementioned challenges, we aim to develop a *label-free* simulated user for *automatic* interaction with CRSs to help them understand complex user preference and improve recommendation results. Our approach draws inspiration from the great success of *reward models* [7, 24] in complex reasoning tasks, which evaluate the correctness of candidate reasoning trajectories without ground-truth answers to provide feedback for the task model to revise its reasoning process. In particular, the recently proposed *generative reward models* [25, 37] further unify generation and reward modeling. They represent the reward as the probability of a specific token (*e.g.,* "Yes" or "No"). That is, the reward is modeled in a generative way, which demonstrates better generalization ability. For our simulated user in conversational recommendation, which also needs to provide *generative, label-free* feedback to CRSs to facilitate recommendation refinement, it is feasible to develop it based on generative reward models.

Table 1: An illustrative case of the complexity of user preference. We use blue to mark the entities mentioned by the user, italicized red for successful recommendations, and italicized gray for unsuccessful ones. The user expresses preference for both "horror" and "comedy" movies. Initially, the CRS focuses solely on the "horror" aspect in its recommendations. With feedback from our simulated user, it considers both aspects and generates a successful recommendation.

USER:	Hello!
SYSTEM:	What kind of movies do you like?
USER:	When I was younger I really enjoyed the A Nightmare on Elm Street (1984).
SYSTEM:	Oh, you like scary movies? I recently watched Happy Death Day (2017).
USER:	I do enjoy some of the newer horror movies that I have seen as well. I also like comedy movies.
SYSTEM: (human)	Have a try for *Happy Death Day 2U (2019)*. You will like it!
SYSTEM: (CRS)	I would recommend *A Quiet Place (2018)*, a popular horror movie.
USER: (simulated)	Thanks for your advice! But I would like to see more comedy elements, like Happy Death Day (2017).
SYSTEM: (CRS)	I know, you should watch *Happy Death Day 2U (2019)*, a sequel to Happy Death Day (2017).

To this end, in our paper, we propose a **G**enerative **R**eward model based **S**imulated **U**ser, named **GRSU**, for automatic interaction with CRSs. Our simulated user can provide feedback through two designed actions inspired by generative reward models: *generative item scoring* as coarse-grained feedback and *attribute-based item critiquing* as fine-grained feedback. These two actions are unified into the format of *instruction*, enabling the development of a unified simulated user via instruction tuning. Based on this, *automatic* multi-turn interaction can be conducted between various CRSs and our simulated user. Furthermore, to achieve a balance between effectiveness and efficiency, we draw inspiration from the paradigm of *reward-guided search* [28, 29] in complex reasoning tasks to leverage beam search in the interaction process. On top of this, we propose an *efficient* candidate ranking method to improve the recommendation results derived from the interaction process.

To validate the effectiveness of our approach, we conduct experiments on public datasets. Experimental results show that with our simulated user, general LLM-based CRSs without fine-tuning outperform several trained CRS models. Notably, with just one round of interaction, our approach achieves superior performance compared to the best baseline model on the INSPIRED dataset, underscoring its efficiency. Additionally, the simulated user trained on high-resource datasets can effectively support CRSs in low-resource datasets, highlighting the transferability of our approach.

2 Related Work

Our work is related to the following two research directions.

Conversational recommendation. Conversational recommender systems (CRSs) aim to provide item recommendations through multi-turn interaction. One line of work [19, 20, 22] focuses on the optimization of interaction policy. They simplify the interaction to pre-defined actions (*e.g.,* asking questions or making recommendations) and handcrafted templates. Based on this, they optimize CRSs to give accurate recommendations within as few turns as possible. Another line of work [31, 32, 41] focuses on the elicitation and understanding of user preference in more free-form natural language conversations. Since conversations usually lack sufficient contextual information, existing work introduces knowledge from external resources, such as knowledge graphs [8], large language models (LLMs) [13, 35], and conversational recommendation corpora [8, 33]. Our work follows the second category and proposes a simulated user, which can automatically interact with CRSs to help them discern the true user preference from a complex conversation.

Generative reward models. Reward models [7, 24] have become an emerging topic for solving complex reasoning tasks. For example, in the commonly used "Best-of-N" strategy [7], a task model first generates several candidate solutions, then a reward model ranks these candidates and selects the best one as the final prediction. Recently, generative reward models [25, 37] have been proposed, which unify generation and reward modeling by representing reward as the probability of a specific token. Based on this, critiques can be introduced in generation for better reward modeling [25, 37]. In this work, we take inspiration from generative reward models to design the actions of our simulated user.

3 Approach

In this section, we present our simulated user based on generative reward models, name **GRSU**, for automatic interaction with CRSs, which is illustrated in Figure 1.

3.1 Overview of the Approach

Task formulation. Conversational recommender systems (CRSs) aim to provide recommendations through multi-turn interaction. At each turn, the system either makes recommendations or chats with the user. Such a process ends when the user accepts recommended items or leaves. A typical CRS comprises two core components: the recommender component and the conversation component. In this paper, we focus on the *recommender* component given the demonstrated prowess of LLMs in conversation [4, 40] following many LLM-based CRSs [13, 33]. Formally, let u demote a user and i denote an item from the item set $\mathcal{I}$. A conversation can be denoted as $C = \{s_t, \mathcal{I}_t\}_{t=1}^{\mathcal{T}}$, where s_t denotes the utterance at the t-th turn, and $\mathcal{I}_t$ denotes the items mentioned in s_t ($\mathcal{I}_t$ is empty if no item is mentioned). With the above definitions, the task of the recommender component in a CRS can be formalized as follows: At the k-th turn, given the conversation history $C_{k-1} = \{s_t, \mathcal{I}_t\}_{t=1}^{k-1}$, the recommender component needs to generate a ranked item list $\hat{\mathcal{I}}_k$ that best matches the ground-truth items in $\mathcal{I}_k$.

LLM-based CRSs. We use LLMs to build CRSs in our approach, as they have demonstrated strong performance in both item recommendation and response generation for conversational recommendation [13, 30]. Given that our idea is to improve CRSs through automatic interaction with simulated users, fine-tuning LLMs for this task is unnecessary. Thus, we directly employ general-purpose LLMs as CRSs. Specifically, following existing work [13], an LLM M^r is prompted with a conversation history C_{k-1}, user feedback f_k on the predicted recommendations $\hat{\mathcal{I}}_k$ (optional), and task description d^r, which can be represented as follows:

$$\hat{\mathcal{I}}_k = \Phi^r(M^r(C_{k-1}, \hat{\mathcal{I}}_k, f_k, d^r)), \tag{1}$$

where Φ^r is a post-processor that maps the items generated by the LLM M^r to those in the item set $\mathcal{I}$. Here, we use fuzzy matching[1] following existing work [13]. Notably, existing single-turn LLM-based CRSs [13, 33, 35] can be regarded as a special case of Eq. 1, wherein only one recommendation is made, and the prompt only contains conversation history C_{k-1} and task description d^r.

Interaction with simulated user. Our approach is inspired by the recent progress on *LLM-based user simulation* for the evaluation of CRSs [15, 30, 43], which utilizes LLMs for role-play with ground-truth user preferences. In our approach, we aim to improve the recommendation of CRSs through automatic interaction with a *label-free* simulated user. As the basis, we fine-tune an LLM as the simulated user (Section 3.2). It can provide feedback through two designed actions inspired by generative reward models: *i.e., generative item scoring* as coarse-grained feedback and *attribute-based item critiquing* as fine-grained feedback. With this simulated user, automatic interaction can be conducted with CRSs (Section 3.3), where CRSs can iteratively refine their recommendations. The approach of interaction with simulated users can be viewed as the paradigm of *reward-guided search* [28, 29]. Thus, we utilize beam search for interaction to balance effectiveness and efficiency.

3.2 Generative Reward Model Based Simulated User

The goal of simulated users in our approach is to provide feedback for CRSs, thereby enabling them to refine their recommendations. Similarly, in complex reasoning tasks, the goal of reward models is to evaluate and provide feedback on reasoning trajectories, guiding task models toward more accurate solutions [7, 24]. Since they share a similar target, it is feasible to develop our simulated user based on reward models. In particular, we draw inspiration from *generative reward models* [25, 37], as the feedback provided to CRSs takes the form of natural language responses. In this section, we first describe the two behaviors of our simulated user, followed by a detailed explanation of how to synthesize instruction data for each behavior to train the simulated user.

3.2.1 User Behavior Description. To guide CRSs to effectively refine their recommendations, we draw inspiration from generative reward models to design two behaviors for the simulated user.

[1] https://github.com/rapidfuzz/RapidFuzz

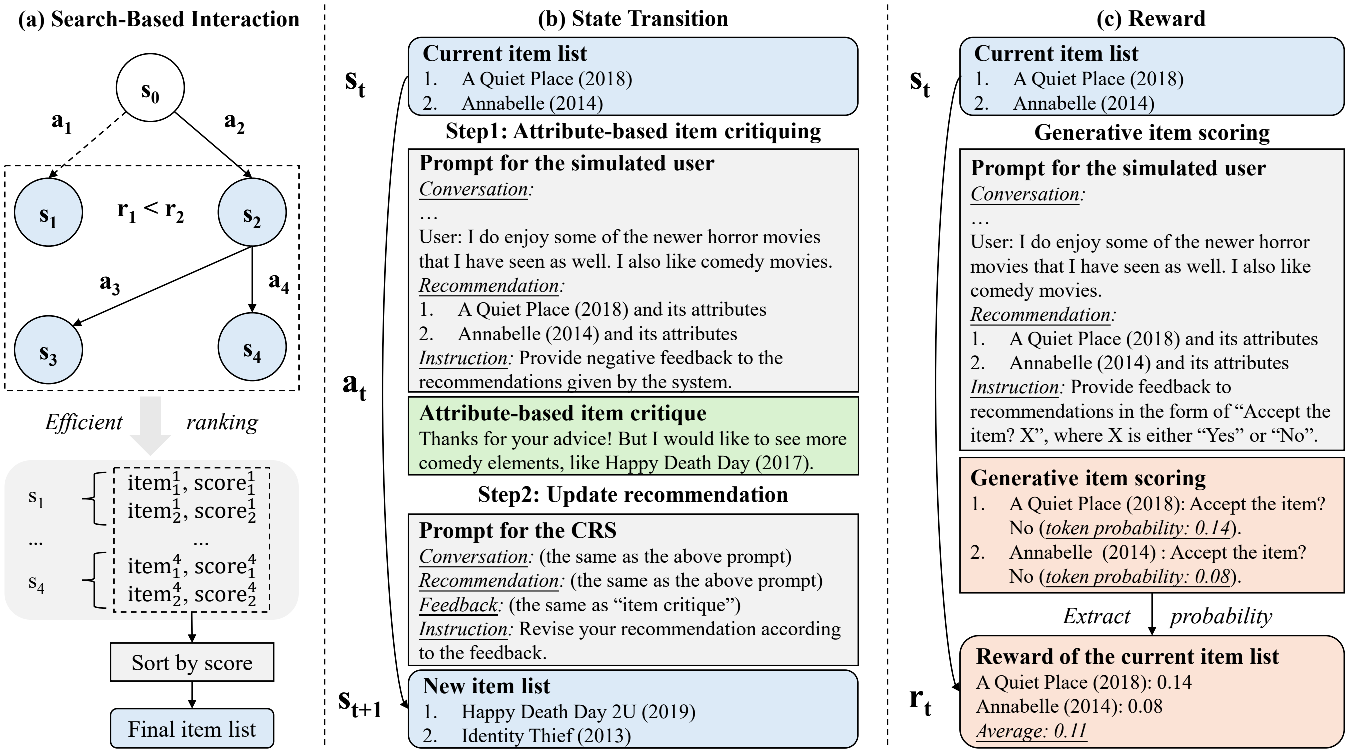
(a) Search-Based Interaction
s0
a1
a2
s1
r1 < r2
s2
a3
a4
s3
s4
Efficient ranking
s1
item¹₁, score¹₁
item¹₂, score¹₂
...
s4
item⁴₁, score⁴₁
item⁴₂, score⁴₂
Sort by score
Final item list
(b) State Transition
st
Current item list
1. A Quiet Place (2018)
2. Annabelle (2014)
Step1: Attribute-based item critiquing
at
Prompt for the simulated user
Conversation:
…
User: I do enjoy some of the newer horror movies that I have seen as well. I also like comedy movies.
Recommendation:
1. A Quiet Place (2018) and its attributes
2. Annabelle (2014) and its attributes
Instruction: Provide negative feedback to the recommendations given by the system.
Attribute-based item critique
Thanks for your advice! But I would like to see more comedy elements, like Happy Death Day (2017).
Step2: Update recommendation
Prompt for the CRS
Conversation: (the same as the above prompt)
Recommendation: (the same as the above prompt)
Feedback: (the same as "item critique")
Instruction: Revise your recommendation according to the feedback.
st+1
New item list
1. Happy Death Day 2U (2019)
2. Identity Thief (2013)
(c) Reward
st
Current item list
1. A Quiet Place (2018)
2. Annabelle (2014)
Generative item scoring
Prompt for the simulated user
Conversation:
…
User: I do enjoy some of the newer horror movies that I have seen as well. I also like comedy movies.
Recommendation:
1. A Quiet Place (2018) and its attributes
2. Annabelle (2014) and its attributes
Instruction: Provide feedback to recommendations in the form of "Accept the item? X", where X is either "Yes" or "No".
Generative item scoring
1. A Quiet Place (2018): Accept the item? No (token probability: 0.14).
2. Annabelle (2014) : Accept the item? No (token probability: 0.08).
Extract probability
rt
Reward of the current item list
A Quiet Place (2018): 0.14
Annabelle (2014): 0.08
Average: 0.11

Figure 1: Approach overview. (a) We use search-based interaction between CRSs and our simulated user to generate candidate items, which is followed by an efficient ranking method to derive the final item list. (b) A simplified state transition example: Starting from a current state (*i.e.,* an item list), the simulated user provides attribute-based item critiques, which the CRS uses to refine its previous recommendations, thereby transitioning to the next state. (c) A simplified reward example: Given a current state (*i.e.,* an item list), the simulated user provides generative item scoring. The probability of the reward token (*i.e.,* "No") is extracted as the score for each item, and their average is the reward of the current state (*i.e.,* the item list).

Generative item scoring. Item scoring (*e.g.,* accepting or rejecting) is the most common behavior exhibited by users in conversational recommendation [2, 3]. Existing work [34, 35] imitates the item scoring behavior in a *discriminative* way, *i.e.,* by learning to assign numerical scores to candidate items. However, such an approach requires separate modeling of the item scoring behavior, as simulated users respond in a *generative* way. Similarly, in LLM-based reward models, most discriminative approaches [7, 24] add a separate module (*e.g.,* a linear head) to the LLM for reward modeling, which do not fully utilize the text-generation capabilities of LLMs. To address this, a recent method, the *generative reward model* [25, 37], proposes representing the reward as the probability of generating a specific reward token (*e.g.,* "Yes" or "No"), and adopting next-token prediction as the training objective. This approach unifies reward and language modeling, which effectively leverages the text-generation capabilities of LLMs for reward modeling. Inspired by this, we propose to model item scoring behavior in a *generative* way. Specifically, we frame item scoring as a single-choice question that requires the simulated user to select one option. The predicted probability of the selected option token is then interpreted as the item score. In addition, considering that CRSs give recommendations in the format of a list, we propose to score each item individually and take the average as the overall score for the item list. This is easier to learn compared with directly scoring the entire list. Formally, at the k-th turn, a generative item scoring f_k^s from our simulated user M^u can be represented as follows:

$$f_k^s = \frac{1}{|\hat{\mathcal{I}}_k^a|} \sum_{j=1}^{|\hat{\mathcal{I}}_k^a|} \Phi^s(M^u(C_{k-1}, \hat{i}_j^a, d^s)), \quad (2)$$

where C_{k-1} is the conversation history, $\hat{\mathcal{I}}_k^a$ is the list of items recommended by CRSs with their attributes, $\hat{i}_j^a$ is one of the items in the list with its attributes, d^s is the task description, and Φ^s is a post-processor that extracts the probability of the reward token. The task description employed for this behavior is as follows: *"You are the user to look for recommendations. You need to provide feedback on the recommendations given by the system. For each recommendation, you should give feedback in the form of "Accept the recommendation (Yes/No)? X", where X is either "Yes" or "No"."*

Attribute-based item critiquing. Critiquing is a representative method for eliciting user preferences using pre-defined attributes in early CRSs [5]. While these critiquing-based methods are straightforward to design, they inherently constrain the expressiveness of the user preference space. Recently, LLMs have shown promise in simulating general human behaviors due to their generative nature [26]. However, when employed as specific user simulators for conversational recommendation, LLMs exhibit limitations in aligning their preferences with those of real users [36]. This problem also exists in reward models, which tend to predict rewards based on superficial features instead of real human preferences. To solve this, natural language critiques have been introduced to calibrate the rewarding process [25, 37]. Inspired by this, we propose to improve LLM-based user simulators with the *critiquing* behavior. Specifically, LLMs are prompted to provide critiques about the items recommended by CRSs. Since the goal of simulated users is to assist CRSs in refining their recommendations, the critique should be specific and constructive. To facilitate this, we add the attributes of the recommended items, enabling the simulated user to generate critiques from the perspective of *attributes*. Formally, at the k-th turn, an attribute-based item critique f_k^c from our simulated user M^u can be represented as follows:

$$f_k^c = M^u(C_{k-1}, \hat{\mathcal{I}}_k^a, d^c), \tag{3}$$

where C_{k-1} is the conversation history, $\hat{\mathcal{I}}_k^a$ is the list of recommended items given by CRSs with their attributes, and d^c is the task description (*"You are the user to look for recommendations. Read the conversation and provide feedback on the recommendations given by the system."*).

3.2.2 *Instruction Data Synthesis For Model Training.*

As discussed earlier, both user behaviors share the same contextual information (*i.e.,* conversation history and items recommended by CRSs) and differ only in the task description, as shown in Eq. 3 and 2. This shared context allows for the development of a unified simulated user through instruction tuning, which involves fine-tuning an LLM using instruction data with distinct instructions for each user behavior. Such an instruction-tuned LLM can effectively and efficiently facilitate the interaction with CRSs. In this part, we first introduce how to synthesize instruction data and then describe how to utilize these data for model training.

Instruction Data Synthesis. Instruction data is usually composed of an instruction that contains the task description and its contextual information and a targeted output [38]. Here, we detail how to synthesize instruction data for the two user behaviors introduced in Section 3.2.1.

- *Attribute-based item critiquing*: For this behavior, although real user feedback on recommendations is available in the dataset, it is often in the style of chit-chat and limited in information density (*e.g.,* lacking detailed item attributes) [30]. This limitation makes it challenging for CRSs to effectively refine their recommendations. Thus, we opt to synthesize such instruction data using an LLM. Specifically, for each recommendation turn k in a conversation of the dataset, we take the conversation history C_{k-1}, recommended item list $\hat{\mathcal{I}}_k^a$, user preference p_k, and task description d^c to construct the instruction $I_k^c = [C_{k-1}; \hat{\mathcal{I}}_k^a; p_k; d^c]$. For the recommended item list $\hat{\mathcal{I}}_k^a$, to cover as many cases as possible, we adopt two kinds of construction methods. Assuming the list contains l items, the first method incorporates ground-truth items by sampling l_p items ($l_p \leq l$) from the ground-truth set $\mathcal{I}_k$ and $l_n = l - l_p$ items as negatives from the remaining set $\mathcal{I} - \mathcal{I}_k$. In contrast, the second method assumes no ground-truth items in the list, sampling l items exclusively from $\mathcal{I} - \mathcal{I}_k$. Compared to Eq. 3, we incorporate a new element, *user preference*, into the instruction, which includes ground-truth items $\mathcal{I}_k$ and their attributes. This addition aims to align the response more closely with the real user behavior. Using the instruction I_k^c, we prompt the LLM to generate the output o_k^c. At inference time, however, ground-truth user preferences are unavailable. Thus, for instruction tuning, we modify the instruction to exclude this element, yielding $I'^c_k = [C_{k-1}; \hat{\mathcal{I}}_k^a; d^c]$. The resulting instruction is paired with o_k^c to create the training data.

- *Generative item scoring*: For this behavior, we adopt the same instruction, $I_k^s = [C_{k-1}; \hat{i}_j^a; d^s]$, as defined in Eq. 2 for data collection. For the recommended item $\hat{i}_j^a$, corresponding ground-truth items $\mathcal{I}_k$ are already present in the dataset. Additionally, we sample k negative items from the set $\mathcal{I} - \mathcal{I}_k$ for each ground-truth item as negative samples. As for the output associated with I_k^s, we follow existing work of generative reward models [25, 37] to use a single-choice question q (*i.e.,* "`Accept the recommendation (Yes/No)?`") concatenated with the option token w (*i.e.,* "`Yes`" or "`No`"). This output is represented as $o_k^s = [q; w]$.

Model Training. With the constructed instruction data $\mathcal{D}$, *i.e.,* $\{(I'^c_k, o_k^c)\}_{k=1}^{n^c}$ and $\{(I_k^s, o_k^s)\}_{k=1}^{n^s}$, we can fine-tune an LLM as our simulated user through instruction tuning. Since both instructions and target outputs are represented in natural language, the training objective can be unified under a sequence-to-sequence loss function. Formally, we minimize the following loss function for model training:

$$L = -\sum_{(I,o) \in \mathcal{D}} \sum_{i=1}^{|o|} \log \Pr(o_i | o_{<i}, I; \Theta^u), \tag{4}$$

where I and o denote the instruction and output, respectively, Θ^u is the parameters of the LLM used as the simulated user, and $o_{<i}$ refers to the tokens preceding the i-th position.

3.3 Search-Based Interaction Between CRS and Simulated User

With the simulated user introduced in the previous section, we can conduct multi-turn interaction with various CRSs. Similarly, in complex reasoning tasks, reward models can be involved in the search process to guide task models toward the correct solution. Drawing inspiration from this, we conceptualize the multi-turn interaction between CRSs and our designed simulated user as a *search* process guided by the simulated user. In this section, we first present the formalization of this interaction, followed by a detailed explanation of how to search for recommendation candidates and efficiently rank them to get the final recommendation result.

The formulation of interaction. Recall that LLM-based CRSs revise recommendations solely based on the most recent set of item recommendations and the feedback on them, as defined in Eq. 1. This makes it possible to model the interaction process as a Markov Decision Process (MDP) with the tuple $(\mathcal{S}, \mathcal{A}, T, r)$. We define the state $s_k \in \mathcal{S}$ as the item list $\hat{\mathcal{I}}_k$ recommended by CRSs and the action $a_k \in \mathcal{A}$ as the critique f_k^c given by the simulated user (defined in

Table 2: Statistics of the datasets after preprocessing.

Datasets	#Conversations	#Turns	#Users	#Items
INSPIRED	999	35,686	999	1,967
ReDial	11,348	139,557	764	6,281

Eq. 3). The transition function T integrates the recommended items $\hat{\mathcal{I}}_k$ from the current state s_k and the critique f_k^c from the action a_k into the recommendation prompt (defined in Eq. 1) for LLM-based CRSs to generate a new item list $\hat{\mathcal{I}}_{k+1}$ as the next state s_{k+1}. The reward function $r_k = r(s_k, a_k)$ measures the quality of the action a_k applied to the state s_k. Since ground-truth labels are unavailable during inference, we use the score f_k^s generated by the simulated user (defined in Eq. 2) as the reward.

Searching the recommendation candidates. The formulation of multi-turn interaction as an MDP allows us to leverage advanced search algorithms. Here, to balance simplicity and performance, we adopt beam search [29] to optimize the recommendation candidates. Specifically, we define a fixed beam width B, expansion width $N(N >= B)$, and search depth D. In the beginning, we generate the initial states by sampling N^2 item lists following Eq. 1, where we only include the conversation history and task description in the prompt. During each search iteration, we proceed as follows: (1) the newly expanded states are scored using the reward function r; (2) only the top B states with the highest rewards are retained; (3) each retained state is expanded into N new states by sampling N actions per state and applying the transition function T to each $(\text{state}, \text{action})$ pair. This process is repeated for D iterations.

Efficient candidate ranking. The aforementioned search process produces multiple candidate item lists for recommendation. However, these item lists may not necessarily be optimal. Recall that the score of an item list is the average of its individual items, as stated in Section 3.2.1. Consequently, a high reward for an item list does not guarantee that every item within the list is of high quality. Similarly, the item lists with lower rewards may still contain valuable items. In light of this, to derive the most suitable item list from the search process, we propose to rank the items from all the generated item lists and take the top-l items with the highest scores as the final item list for recommendation. Note that such a ranking method is *efficient* because the scores for individual items are already computed during the search process, and the only additional step required is a sorting operation based on these scores.

4 Experiment

In this section, we first set up the experiments and then report the results and give a detailed analysis.

4.1 Experimental Setup

Datasets. To verify the effectiveness of our approach, we conduct experiments on two widely used conversational recommendation datasets, following existing work [8, 13, 35], *i.e.,* ReDial [21] and INSPIRED [12]. The ReDial dataset is a conversational recommendation dataset about movie recommendations and is constructed by employing crowdsourcing workers on Amazon Mechanical Turk (AMT). Similar to ReDial, the INSPIRED dataset also centers on conversational movie recommendations, but with a much smaller size. The statistics of both datasets are summarized in Table 2. We use the same dataset splits as a recent LLM-based CRS work [13, 33], which removes repeated items within each conversation for more reliable evaluation.

Baselines. We make comparisons with two groups of baselines:

(1) General LLMs as zero-shot CRSs [13]:

• **Llama-3.1-8B-Instruct** [10]: It is a representative 8B parameter open-source model.

• **Phi-4** [1]: It is a 14B parameter open-source model trained using strategically synthetic data and novel post-training methods, which surpasses GPT-4o on representative reasoning benchmarks.

• **Llama-3.3-70B-Instruct** [10]: It is a representative 70B parameter open-source model.

• **GPT-4o** [16]: It is a versatile, high-intelligence flagship closed-source model. We use the `gpt-4o-2024-08-06` version for it.

(2) Representative CRS models:

• **TREA** [23]: It constructs a multi-hierarchical, scalable tree for reasoning over the mentioned entities in the conversation.

• **VRICR** [39]: It proposes a variational Bayesian method to dynamically refine the incomplete knowledge graph and select relevant knowledge.

• **DCRS** [8]: It designs a demonstration retriever with knowledge-aware contrastive learning and an adaptive prompt learning approach to utilizing retrieved exemplars.

• **NBCRS** [33]: It is a neighbourhood-based CRS that utilizes the frequency of items in similar conversations.

• **ReFICR** [35]: It fine-tunes an LLM as CRS with retrieval and generation instructions constructed from existing datasets.

Evaluation metrics. Given the demonstrated prowess of LLMs in conversation [4], we follow existing work [13, 33] to focus our evaluation on the performance of the recommendation subtask. We adopt Recall@k, NDCG@k, and MRR@k (k=10, 50) for performance evaluation. For all the above metrics, we calculate and report the average scores on all the test examples from three runs.

Implementation details. For both CRSs and our simulated user, we set the length of all the item lists to 10 and utilize the attributes *genre*, *actor*, *writer*, and *director* for each item (movie). In our main experiments, we employ `Phi-4` for LLM-based CRSs, as it offers a balance between excellent performance and appropriate model size. For detailed analysis, we additionally consider models with varying parameter scales, including `Llama-3.1-8B-Instruct`, `Llama-3.3-70B-Instruct`, and `GPT-4o`. For the simulated user, we adopt `Llama-3.1-8B-Instruct` as the base model and leverage `Llama-3.3-70B-Instruct` for generating instruction data. During instruction data synthesis, we construct item lists for attribute-based item critiquing by including all possible subsets of ground-truth items with lengths less than 10 for lists containing ground-truth items. For item lists without ground-truth items, we sample one list per conversation history. For generative item scoring, we sample one additional item as a negative sample for each ground-truth item. For instruction tuning, we train for one epoch with a batch size of 4. We utilize the AdamW optimizer with default parameters, a linear warmup over 1,000 steps, and a cosine decay learning rate scheduler, which decays to 10% of the peak learning rate after the decay period. The learning rate is set to 10^{-6}, with a weight decay of 0.01 and gradient norm clipping at 1.0. For tuning LLMs, we use bf16 16-bit mixed precision training, FlashAttention-2 [9], gradient checkpointing [6], and DeepSpeed with ZeRO-3 [27]. For searching recommendation candidates, we set the beam width and expand width to 4 and the search depth to 5. For the implementation of baseline methods, we use CRSLab [42]. Our code is available at the link: https://github.com/RUCAIBox/GRSU.

4.2 Main Results

Table 3 presents the recommendation performance of baseline and our methods. For general LLM-based zero-shot CRSs, the performance order is consistent across both datasets, *i.e.,* `GPT-4o` > `Llama-3.3-70BInstruct` > `Phi-4` > `Llama-3.1-8B-Instruct`. This ranking aligns with their performance on other benchmarks. Notably, while `Phi-4` performs closely to `Llama-3.3-70B-Instruct`, especially on the ReDial dataset, its model size is significantly smaller. This efficiency is why we employ `Phi-4`.

For representative CRS models, we can see that DCRS and ReFICR mostly outperform the other baseline methods in all the metrics, while the former performs better on the `INSPIRED` dataset and the latter performs better on the `ReDial` dataset. Both methods leverage external knowledge to enhance the understanding of user preference from the conversation context. Specifically, DCRS incorporates the knowledge from a conversational recommendation corpus by retrieval, while ReFICR utilizes the knowledge from an LLM by instruction tuning. Given the abundant size of the ReDial dataset, ReFICR effectively utilizes the semantic and item knowledge encoded in the LLM for improved user preference understanding and item recommendation. This advantage enables it to outperform DCRS on this high-resource dataset. However, on the low-resource `INSPIRED` dataset, the instruction tuning of ReFICR may be insufficient. In contrast, retrieval from an existing corpus compensates for the data scarcity, which makes DCRS outperform ReFICR.

Lastly, our proposed approach, GRSU, consistently surpasses all the baseline methods. We design the behavior of attribute-based item critiquing for GRSU to provide detailed feedback in the interaction process, which can help CRSs generate diverse and relevant item recommendations. In addition, we design the behavior of generative item scoring, which can effectively leverage the text generation capabilities of LLMs to align closely with the real user preference to give accurate item scores. Notably, GRSU achieves a substantial performance improvement on the low-resource INSPIRED dataset. This is because we focus on optimizing the simulated user, which provides feedback on recommended items instead of directly generating recommendations. This design allows us to utilize other datasets (*e.g.,* ReDial) for training, effectively mitigating the challenge of data scarcity. For this point, we provide a detailed analysis in Section 4.3.3. Furthermore, our approach employs a general LLM as the CRS without fine-tuning it on the datasets. The results demonstrate significant performance gains compared to the original LLM. This highlights that general LLMs inherently possess substantial knowledge about target items, and the interaction with our simulated user effectively aligns them with target application distributions, thereby unlocking their potential as high-performing CRSs.

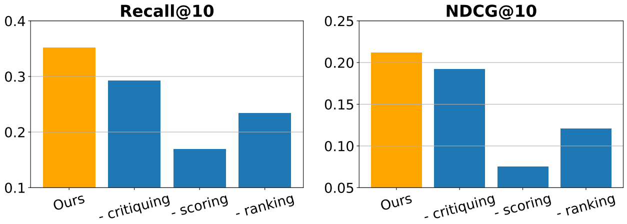

Figure 2: Ablation study on the INSPIRED dataset. "critiquing" refers to attribute-based item critiquing, "scoring" refers to generative item scoring, and "ranking" refers to efficient candidate ranking.

4.3 Detailed Analysis

4.3.1 Ablation Study. Our approach incorporates several components to enhance the interaction between the CRS and the simulated user, ultimately improving recommendation performance. To verify the effectiveness of each component, we conduct an ablation study on the INSPIRED dataset using `Phi-4` as the CRS. The evaluation metrics include Recall@10 and NDCG@10, as the length of all item lists is fixed at 10. The ablation study considers the removal of specific user behaviors, namely attribute-based item critiquing and generative item scoring. Additionally, we examine the impact of excluding the efficient ranking method.

The results are shown in Figure 2. We can see that removing any component would lead to a drop in the performance. It indicates that all the components in our approach are useful to improve the performance of recommendation. Notably, the removal of generative item scoring results in the most significant performance drop. This finding underscores the importance of generative item scoring in our approach, as it predicts user acceptance of recommended items and directly influences CRS performance. For the two metrics Recall@10 and NDCG@10, the impact is more significant for NDCG@10. This is attributed to the focus of NDCG on the ranking quality of ground-truth items, which depends heavily on the generative item scoring behavior.

4.3.2 Implementation of User Behaviors. For our simulated user GRSU, we design two user behaviors for it. In the above part, we have verified the effectiveness of both behaviors. In this part, we further justify our implementations by comparing them with alternative designs. Specifically, we conduct the study on the INSPIRED dataset using `Phi-4` as the CRS. The evaluation metrics include Recall@10, Recall@50, NDCG@10, and NDCG@50. For attribute-based item critiquing, we consider alternative implementations, including removing instruction tuning and item attributes. For generative item scoring, we consider variations including removing instruction tuning, adopting majority voting, discriminative scoring, and combining majority voting with generative item scoring

Table 3: Main results of our approach and baseline methods. Following existing work [13, 33], we remove repeated items within each conversation for more reliable evaluation. The best method is marked in bold. Numbers marked with * indicate that the improvement is statistically significant compared with the baseline (t-test with p-value < 0.05).

Methods	ReDial						INSPIRED					
	Recall		NDCG		MRR		Recall		NDCG		MRR	
	@10	@50	@10	@50	@10	@50	@10	@50	@10	@50	@10	@50
Llama-3.1-8B-Instruct	0.147	0.265	0.077	0.103	0.055	0.061	0.166	0.285	0.090	0.117	0.066	0.072
Phi-4	0.173	0.313	0.095	0.126	0.071	0.077	0.145	0.277	0.077	0.106	0.056	0.062
Llama-3.3-70B-Instruct	0.174	0.317	0.096	0.128	0.073	0.079	0.166	0.311	0.093	0.127	0.070	0.079
GPT-4o	0.187	0.355	0.104	0.141	0.079	0.087	0.217	0.378	0.124	0.159	0.096	0.103
NBCRS	0.130	0.287	0.065	0.104	0.044	0.052	0.123	0.204	0.068	0.085	0.057	0.057
VRICR	0.148	0.327	0.066	0.105	0.041	0.049	0.082	0.244	0.039	0.075	0.026	0.034
TREA	0.175	0.365	0.083	0.124	0.055	0.063	0.136	0.307	0.080	0.115	0.063	0.069
DCRS	0.179	0.366	0.091	0.132	0.064	0.073	0.192	0.362	0.123	0.161	0.101	0.109
ReFICR	0.216	0.467	0.099	0.155	0.065	0.076	0.147	0.326	0.074	0.113	0.051	0.060
GRSU	**0.250***	**0.488***	**0.135***	**0.187***	**0.100***	**0.111***	**0.352***	**0.559***	**0.212***	**0.258***	**0.170***	**0.180***

Table 4: Ablation study of the two behaviors on the INSPIRED dataset. The best one is marked in bold. Numbers marked with * indicate that the improvement is statistically significant compared with the baseline (t-test with p-value < 0.05).

Methods	Recall@10	Recall@50	NDCG@10	NDCG@50
GRSU	**0.352***	**0.559***	**0.212***	**0.258***
Attribute-based item critiquing				
- instruction tuning	0.300	0.511	0.184	0.232
- attribute	0.299	0.487	0.169	0.212
Generative item scoring				
- instruction tuning	0.156	0.382	0.083	0.131
discriminative scoring	0.287	0.503	0.124	0.191
majority voting	0.189	0.387	0.093	0.136
majority voting + generative scoring	0.205	0.486	0.095	0.158

Table 5: Transferability of the simulated user on the INSPIRED dataset. The simulated user is trained using three configurations: the high-resource ReDial dataset, the low-resource INSPIRED dataset, and a combined dataset incorporating both. The best one is marked in bold. Numbers marked with * indicate that the improvement is statistically significant compared with the baseline (t-test with p-value < 0.05).

Training datasets	Recall@10	Recall@50	NDCG@10	NDCG@50
INSPIRED (low resource)	0.234	0.463	0.127	0.178
ReDial (high resource)	0.304	0.539	0.176	0.229
ReDial+INSPIRED	**0.352***	**0.559***	**0.212***	**0.258***

(*i.e.,* using the scores from generative item scoring as weights for majority voting).

The results are shown in Table 4. We can see that any alternative implementation would lead to performance degradation. It indicates the effectiveness of our implementations for the two user behaviors. For attribute-based item critiquing, the most significant performance decline occurs when item attributes are removed. This highlights the importance of attributes in item critiques, as they encapsulate specific user preference information that CRSs can leverage to recommend more relevant items. For generative item scoring, the largest performance drop is observed when instruction tuning is removed. This finding emphasizes the critical role of alignment with real user preferences in item scoring, consistent with observations from a recent work [14]. Furthermore, generative item scoring significantly outperforms majority voting and discriminative scoring, demonstrating the distributional gap between LLMs and real users. Thus, even if we incorporate the scores given by generative item scoring into majority voting, there still exists a significant performance gap with our implementation.

4.3.3 Transferability of the Simulated User. One important limitation of existing CRSs is their requirement for specialized training tailored to each dataset due to the differences in item sets. Our simulated user provides a potential solution to this issue. This is because it only needs to provide feedback on recommended items instead of directly generating item recommendations. To validate the transferability of our simulated user from high-resource datasets to low-resource datasets, we conduct experiments on the ReDial and INSPIRED datasets using `Phi-4` as the CRS. The evaluation metrics include Recall@10, Recall@50, NDCG@10, and NDCG@50. Specifically, we train the simulated user under three configurations: using only the high-resource ReDial dataset, using only the low-resource INSPIRED dataset, and using a combined dataset that integrates both. The performance is then assessed on the test set of the low-resource INSPIRED dataset.

The results are shown in Table 5. As we can see, the simulated user trained only on the high-resource ReDial dataset performs

Table 6: The performance of various search methods with the simulated user on the INSPIRED dataset. The best one is marked in bold.

Search methods	Recall@10	Recall@50	NDCG@10	NDCG@50
No search (zero shot)	0.145	0.277	0.077	0.106
Monte Carlo search	0.293	0.484	0.192	0.237
Greedy search (small)	0.317	0.436	0.196	0.223
Greedy search (large)	0.338	**0.580**	0.210	**0.265**
Beam search (ours)	**0.352**	0.559	**0.212**	0.258

much better than directly using the low-resource INSPIRED dataset itself. This substantial improvement underscores the transferability of our simulated user across different datasets. Furthermore, incorporating the INSPIRED dataset can further improve the performance. This indicates that the simulated user can effectively utilize the data from diverse sources in training. We leave the scaling of training data for CRSs and simulated users as future work.

4.3.4 Impact of the Search Strategy. Recall that in Section 3.3, we employ beam search to facilitate the interaction between CRSs and our simulated user. To systematically investigate the impact of search strategies within our approach, we compare beam search with several alternative search strategies, including Monte Carlo search and greedy search. Monte Carlo search performs multiple one-step sampling operations. That is, it limits the search depth to one. Greedy search chooses the best among multiple samples at each step. That is, it limits the beam width to one. For all the search strategies, we use the same formulation described in Section 3.3 and only replace beam search with each search method. To ensure comparability, we maintain the same number of overall generated item lists across all search strategies, thus preserving a similar search space. Additionally, for greedy search, we examine a variant with the same expand width as beam search, referred to as *greedy search (small)*, alongside the standard version where the number of generated item lists matches beam search, referred to as *greedy search (large)*. We conduct this comparison on the INSPIRED dataset, utilizing `Phi-4` as the CRS. The evaluation metrics include Recall@10, Recall@50, NDCG@10, and NDCG@50.

The results are shown in Table 6. As we can see, all the search methods significantly outperform the no-search baseline, suggesting the importance of multi-turn interaction for CRSs. When maintaining a similar search space (*i.e.,* Monte Carlo, greedy (large), and beam search), we observe comparable performance for greedy and beam search. It indicates the effectiveness of our efficient ranking method, which takes advantage of all the searched items at a low cost. In addition, they perform better than Monte Carlo search. This is because the search depth of Monte Carlo search is limited to one, which does not involve the feedback of attribute-based item critiquing from our simulated user. Furthermore, when we set the expand width of greedy search to the same as beam search, we find a drop in its performance. It demonstrates the importance of search space for effective interaction.

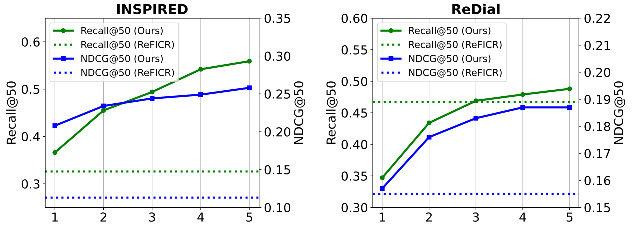

Figure 3: The performance changes with respect to the depth of search. As a reference, we add the performance of ReFICR.

4.3.5 Efficiency of Search-Based Interaction. In addition to the superior performance, the efficiency is also vital to make the computation cost of the search manageable. Thus, we analyze the search efficiency by plotting performance changes with respect to the search depth. Specifically, we conduct the study on the ReDial and INSPIRED datasets using `Phi-4` as the CRS. We report Recall@50 and NDCG@50 metrics across search depths ranging from one to five. For comparison, we include the performance of the best baseline method, ReFICR, as a reference point.

As illustrated in Figure 3, we observe a continuous performance improvement on both datasets. It indicates the effectiveness of the attribute-based item critiquing behavior of our simulated user, which can consistently provide valuable feedback for CRSs to revise recommendations. Compared with ReFICR, our approach achieves better performance at the first and third turns on the INSPIRED and ReDial datasets, respectively. Even within a limited cost, our method achieves superior performance, underscoring its efficiency.

5 Conclusion

In this paper, we propose a generative reward model based simulated user, named **GRSU**, for search-based interaction with CRSs to solve the issue of complex user preferences. First, we take inspiration from generative reward models to design two actions for our simulated user: *i.e., generative item scoring* as coarse-grained feedback and *attribute-based item critiquing* as fine-grained feedback. Then, we unify these two actions into the format of instruction, which enables us to develop a unified simulated user with instruction tuning on synthesized data. Based on this simulated user, automatic interaction is conducted with CRSs to help them understand complex user preferences and revise recommendations with feedback of different granularity. Furthermore, we leverage beam search for the interaction process to achieve a balance between effectiveness and efficiency. Finally, we propose an *efficient* candidate ranking method to improve the recommendation results derived from the interaction process. Extensive experiments demonstrate the effectiveness, efficiency, and transferability of our approach.

As future work, we will investigate the scaling of model size and dataset size for simulated users in conversation recommendation.

Acknowledgments

This work was partially supported by National Natural Science Foundation of China under Grant No. 92470205 and 62222215, Beijing Municipal Science and Technology Project under Grant No. Z231100010323009, and Beijing Natural Science Foundation under Grant No. L233008, and the Outstanding Innovative Talents Cultivation Funded Programs 2022 of Renmin University of China. Xin Zhao is the corresponding author.

References

[1] Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. 2024. Phi-4 technical report. *arXiv preprint arXiv:2412.08905* (2024).

[2] Wanling Cai and Li Chen. 2019. Towards a Taxonomy of User Feedback Intents for Conversational Recommendations. *RecSys (Late-Breaking Results)* 2431 (2019), 51–55.

[3] Wanling Cai and Li Chen. 2020. Predicting user intents and satisfaction with dialogue-based conversational recommendations. In *Proceedings of the 28th ACM Conference on User Modeling, Adaptation and Personalization*. 33–42.

[4] Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. 2024. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology* 15, 3 (2024), 1–45.

[5] Li Chen and Pearl Pu. 2012. Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction* 22 (2012), 125–150.

[6] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174* (2016).

[7] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168* (2021).

[8] Huy Dao, Yang Deng, Dung D Le, and Lizi Liao. 2024. Broadening the view: Demonstration-augmented prompt learning for conversational recommendation. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 785–795.

[9] Tri Dao. [n. d.]. FlashAttention-2: Faster Attention with Better Parallelism and Work Partitioning. In *The Twelfth International Conference on Learning Representations*.

[10] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783* (2024).

[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] Shirley Anugrah Hayati, Dongyeop Kang, Qingxiaoyang Zhu, Weiyan Shi, and Zhou Yu. 2020. Inspired: Toward sociable recommendation dialog systems. *arXiv preprint arXiv:2009.14306* (2020).

[13] 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.

[14] Zhankui He, Zhouhang Xie, Harald Steck, Dawen Liang, Rahul Jha, Nathan Kallus, and Julian McAuley. 2024. Reindex-Then-Adapt: Improving Large Language Models for Conversational Recommendation. *arXiv preprint arXiv:2405.12119* (2024).

[15] Chen Huang, Peixin Qin, Yang Deng, Wenqiang Lei, Jiancheng Lv, and Tat-Seng Chua. 2024. Concept–An Evaluation Protocol on Conversation Recommender Systems with System-and User-centric Factors. *arXiv preprint arXiv:2404.03304* (2024).

[16] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. 2024. Gpt-4o system card. *arXiv preprint arXiv:2410.21276* (2024).

[17] Dietmar Jannach, Ahtsham Manzoor, Wanling Cai, and Li Chen. 2021. A survey on conversational recommender systems. *ACM Computing Surveys (CSUR)* 54, 5 (2021), 1–36.

[18] Yucheng Jin, Li Chen, Wanling Cai, and Pearl Pu. 2021. Key qualities of conversational recommender systems: From users' perspective. In *Proceedings of the 9th International Conference on Human-Agent Interaction*. 93–102.

[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] 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).

[22] Shijun Li, Wenqiang Lei, Qingyun Wu, Xiangnan He, Peng Jiang, and Tat-Seng Chua. 2021. Seamlessly unifying attributes and items: Conversational recommendation for cold-start users. *ACM Transactions on Information Systems (TOIS)* 39, 4 (2021), 1–29.

[23] Wendi Li, Wei Wei, Xiaoye Qu, Xian-Ling Mao, Ye Yuan, Wenfeng Xie, and Dangyang Chen. 2023. TREA: Tree-Structure Reasoning Schema for Conversational Recommendation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 2970–2982.

[24] Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. [n. d.]. Let's Verify Step by Step. In *The Twelfth International Conference on Learning Representations*.

[25] Dakota Mahan, Duy Van Phung, Rafael Rafailov, Chase Blagden, Nathan Lile, Louis Castricato, Jan-Philipp Fränken, Chelsea Finn, and Alon Albalak. 2024. Generative reward models. *arXiv preprint arXiv:2410.12832* (2024).

[26] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*. 1–22.

[27] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3505–3506.

[28] Charlie Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. 2024. Scaling llm test-time compute optimally can be more effective than scaling model parameters. *arXiv preprint arXiv:2408.03314* (2024).

[29] Ziyu Wan, Xidong Feng, Muning Wen, Stephen Marcus McAleer, Ying Wen, Weinan Zhang, and Jun Wang. 2024. Alphazero-like tree-search can guide large language model decoding and training. In *Forty-first International Conference on Machine Learning*.

[30] Xiaolei Wang, Xinyu Tang, Wayne 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*. 10052–10065.

[31] 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.

[32] 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.

[33] Zhouhang Xie, Junda Wu, Hyunsik Jeon, Zhankui He, Harald Steck, Rahul Jha, Dawen Liang, Nathan Kallus, and Julian McAuley. 2024. Neighborhood-Based Collaborative Filtering for Conversational Recommendation. In *Proceedings of the 18th ACM Conference on Recommender Systems*. 1045–1050.

[34] 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*. 38–48.

[35] 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*. 43–52.

[36] Se-eun Yoon, Zhankui He, Jessica Echterhoff, and Julian McAuley. 2024. Evaluating Large Language Models as Generative User Simulators for Conversational Recommendation. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. 1490–1504.

[37] Lunjun Zhang, Arian Hosseini, Hritik Bansal, Mehran Kazemi, Aviral Kumar, and Rishabh Agarwal. [n. d.]. Generative Verifiers: Reward Modeling as Next-Token Prediction. In *The 4th Workshop on Mathematical Reasoning and AI at NeurIPS'24*.

[38] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, et al. 2023. Instruction tuning for large language models: A survey. *arXiv preprint arXiv:2308.10792* (2023).

[39] 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.

[40] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A Survey of Large Language Models. *arXiv preprint arXiv:2303.18223* (2023).

[41] Zhipeng Zhao, Kun Zhou, Xiaolei Wang, Wayne Xin Zhao, Fan Pan, Zhao Cao, and Ji-Rong Wen. 2023. Alleviating the long-tail problem in conversational recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems*. 374–385.

[42] Kun Zhou, Xiaolei Wang, Yuanhang Zhou, Chenzhan Shang, Yuan Cheng, Wayne Xin Zhao, Yaliang Li, and Ji-Rong Wen. 2021. CRSLab: An Open-Source Toolkit for Building Conversational Recommender System. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: System Demonstrations*. 185–193.

[43] 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).