POPCORN: Human-in-the-Loop Popularity Debiasing in Conversational Recommender Systems

Zuohui Fu[†*], Yikun Xian[†*], Shijie Geng[†], Gerard de Melo[‡], Yongfeng Zhang[†]
[†]Rutgers University, New Brunswick, NJ [‡]HPI/Univ. of Potsdam, Germany
zuohui.fu@rutgers.edu,yikun.xian@rutgers.edu,sg1309@rutgers.edu,gdm@demelo.org,yongfeng.zhang@rutgers.edu

ABSTRACT

Recent conversational recommender systems (CRS) provide a promising solution to accurately capture a user's preferences by communicating with users in natural language to interactively guide them while pro-actively eliciting their current interests. Previous research on this mainly focused on either learning a supervised model with semantic features extracted from the user's responses, or training a policy network to control the dialogue state. However, none of them has considered the issue of popularity bias in a CRS. This paper proposes a human-in-the-loop popularity debiasing framework that integrates real-time semantic understanding of open-ended user utterances as well as historical records, while also effectively managing the dialogue with the user. This allows the CRS to balance the recommendation performance as well as the item popularity so as to avoid the well-known "long-tail" effect. We demonstrate the effectiveness of our approach via experiments on two conversational recommendation datasets, and the results confirm that our proposed approach achieves high-accuracy recommendation while mitigating popularity bias.

CCS CONCEPTS

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

KEYWORDS

Conversational Recommender System; Dialogue State Management; Popularity Bias; Debiasing

ACM Reference Format:
Zuohui Fu, Yikun Xian, Shijie Geng, Gerard de Melo, Yongfeng Zhang. 2021. POPCORN: Human-in-the-Loop Popularity Debiasing in Conversational Recommender Systems. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3482461

1 INTRODUCTION

Popularity bias has emerged as a crucial problem in many practical recommender systems, where the popular (i.e., frequently rated or purchased) items get substantial exposure, while less popular ones are under-represented. A market that suffers from popularity bias will lack opportunities for users to discover more obscure products and will tend to be dominated by just a few large brands or well-known artists [1, 3, 6]. Recently, numerous approaches have been proposed to mitigate popularity bias in recommender systems [1, 2, 7, 23, 38, 52, 54].

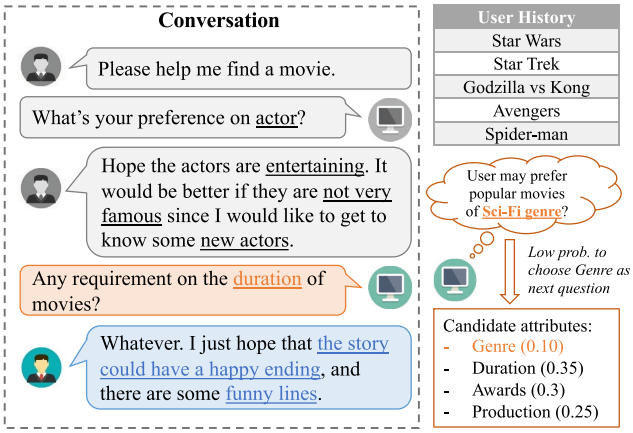


Figure 1: Example of popularity bias in existing CRS. Based on the history, the agent mistakenly assumes the user desires only popular Sci-Fi movies, and lowers the probability of asking about the genre. However, it turns out the user is actually seeking other types of movies rather than Sci-Fi.

At the same time, extensive efforts have also been made to develop intelligent conversational recommender systems (CRS) that interactively guide a user towards a recommendation through dialogue, much like a human shop assistant would [13, 14, 36, 49]. However, it remains unclear how to best reduce popularity bias in this modern CRS setting.

Existing debiasing methods for traditional recommendation cannot be easily applied to the CRS setting directly. Most of them adopt one-shot debiasing for biases stemming from the feedback loop in conventional recommendation [4, 5, 7, 18, 23, 33, 44, 45, 48], where the interactions only include views, clicks, or ratings, while users rarely actively explore the particular attributes of items. In contrast, in CRS settings, we face the risk of a new form of multi-round popularity bias within the conversations. This is because such CRS typically adopt human-in-the-loop (HitL) learning [14, 26], which requires the agent to not only elicit user preferences but also take the initiative in interacting with users. In this paradigm, users are deeply involved in the recommendation process and they gradually

*Both authors contributed equally to this work.

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482461

reveal their preferences with regard to item attributes during the interaction. As illustrated in Fig. 1, it is challenging to study and mitigate the bias arising in multi-round HitL conversational turns. We refer to this issue as the *human-in-the-loop popularity bias*.

At the same time, the modern HitL CRS paradigm requires considering user preference updates during the multi-round conversational turns in order to strike a good balance between the recommendation utility and popularity bias. Some traditional popularity debiasing methods rely on a purely regularization-based method to improve the exposure of long-tail items [2, 7, 38, 54], while others focus on adversarial learning, causal reasoning, and re-rank strategies [1, 23, 52]. However, in CRS settings, it would be harmful to blindly apply a uniform debiasing strategy, as users may have already clarified some of their desires to the agent. Furthermore, how to control the weight of different popularity biases is another essential aspect in the CRS scenario, since the global distribution of item popularity should not determine the assumed preferences of individual users [6]. In order to guarantee a fair exposure of recommendation results for different users, a CRS should gain the capability of dynamically adjusting the individual popularity preferences from the user history and global popularity statistics from the platform in a multi-round scenario.

One of the major streams of research on CRS assumes a modular framework [49], typically including user interaction modeling (UIM), dialogue state management (DSM), and recommendation (REC). Specifically, UIM involves encoding information from interactive user–agent dialogue utterances as well as historical user records, while, based on the UIM's outputs, the DSM pays attention to the importance of dynamic conversational state updates to select a suitable next action in each conversational turn. The DSM decides which question to ask and whether it is the right time to reveal the current recommendation results proposed by the REC module to the user. This interplay between modules in a CRS is similar to guessing what might be the preferred item in a user's mind by asking the user questions [13, 36, 42, 46, 49]. Once the user has provided a response revealing particular preferences with regard to item attributes, the system can prune off irrelevant item candidates.

Typically, traditional DSMs determine which questions to ask (pertaining to specific item attributes) such that the recommendation utility (e.g., in terms of purchases or clicks) is maximized in fewer conversational turns. However, existing CRS methods such as collaborative filtering-based ones implicitly encode the user histories [9, 24, 25], which may mislead it with regard to the user preferences. For example, a user may be known to have watched a number of science fiction movies such as *StarWars* in the past. Such information can lead the DSM to assign a lower probability to questions about the movie *category*, as it may blindly assume the user prefers popular science fiction films such as *Star Trek*. Hence, the agent may emphasize movies with these attributes over others. In reality, a user may at times prefer less popular crime thrillers or even romantic comedies if they have not expressed any contrary interests during the conversation. Unfortunately, none of the existing DSM module designs account for popularity bias in their objective functions, considering both questions in the DSM module and debiasing historical user records in the UIM module. To achieve the flexibility of asking about potential preferences along item attributes, the DSM module ought to be carefully designed so as to consider how to ask questions in each round to alleviate the popularity bias from the UIM module.

To this end, we propose a new method called Popcorn for POPularity debiasing in COnversational RecommendatioN. Specifically, Popcorn consists of three modules for user-interaction modeling (UIM), dialogue state management with popularity debiasing (DSM-PD), and recommendation (REC). The UIM module first encodes real-time user–agent dialogue as well as historical user records to derive a personalized state vector. Since not all historical items the user has interacted with are useful at a given time, the UIM module can automatically identify important items and balance between them and the real-time dialogue context. The DSM-PD module then takes the UIM's state vector as input and predicts the next action: either asking a question to obtain user preferences on a certain attribute or making recommendations at a suitable time. The DSM module explicitly takes into consideration the popularity bias of a potential recommendation and integrates it as a supervised signal into the learning process so that the CRS agent can make correct decisions to balance between recommendation performance and popularity bias. Since the popularity bias signal is non-differentiable, we formalize the DSM-PD training procedure as a multi-objective Markov decision process and propose a new variant of policy gradient to learn the parameters of the DSM-PD module. Finally, the recommendation module (REC) is triggered by the DSM-PD module at a suitable time to make final predictions incorporating both dialogue context and historical user information.

In light of the above, our main contributions include:

- We highlight important shortcomings of previous conversational recommender systems (CRS), which fail to consider popularity bias arising in recommendation decisions.
- We conduct data-driven studies to assess the popularity bias over the distributions of attribute values within the items. Additionally, we devise schemes to modify existing evaluation metrics to quantify the popularity bias in a CRS.
- We propose a new popularity debiasing method for HitL conversational recommendation to address this issue by jointly incorporating user interaction modeling and dialogue state management, which leads to debiased behavior via novel reward functions.
- Our model yields high-quality recommendation results on two diverse real-world datasets while narrowing the popularity bias within the recommended results. Extensive experiments along with qualitative results demonstrate the effectiveness of our proposed debiased CRS.

2 RELATED WORK

Conversational Recommender Systems. Traditional static recommender systems rely on offline user–item interaction histories, and suffer from limitations when better estimating user preferences requires a deeper user participation. In contrast, conversational recommender systems (CRS) model the recommendation task under a human-in-the-loop (HitL) learning paradigm [14], which relies on task-oriented multi-turn dialogue to interact with users so as to precisely elicit their preferences and consumption motivations. Earlier CRS attempts focused on collecting user preferences by asking choice-based questions [10, 19, 29]. Recent research in the area

falls into two broad categories – natural language understanding (NLU) [49] and dialogue state management (DSM) [36]. With the recent breakthroughs in NLP on generating and understanding free-form text, communicating with users in a multi-turn conversation becomes a natural way of eliciting their preferences and addressing their needs. Research focusing on NLU develops models to uncover the semantics held within the user responses. The multi-turn CRS strategies of switching between question asking and recommendation is based on the extracted semantic features from the conversation [22, 27, 49] or external knowledge graphs [14, 53]. However, most of these methods learn the switching strategies in a supervised manner and do not automatically explore potentially more optimal strategies. This requires a carefully curated training dataset such that the model learns to pick up only desirable strategies [14]. Different from semantics-based methods, another branch of research pays more attention to designing a dialogue state management (DSM) module that optimizes for a successful recommendation decision in fewer conversational turns [24, 36]. Such approaches consider CRS as a task-oriented dialogue system that asks questions about pre-defined attributes and fills the slot of a given attribute in each conversational turn directly based on the user's response [11, 39, 51]. The ultimate goal is to pick appropriate actions automatically in order to more quickly arrive at a successful recommendation.

Bias and Debiasing in Recommender Systems. Recommendation models are trained on vast amounts of user behavior data. As a result, they inevitably absorb biases inherent in the data as well as those manifested by the feedback loop of recommendation [17], such as selection bias, exposure bias, position bias, popularity bias, etc. [6]. In order to improve the robustness against these biases, many studies on debiasing have recently been put forth. These address different types of biases by proposing new evaluation metrics [4, 5, 15, 44], conducting data augmentation [18, 23, 35, 45], or introducing new training regularization constraints [7, 33, 43, 47, 48]. Among all forms of bias, popularity bias [1] is prevalent in nearly all kinds of recommender systems. Due to the "Matthew Effect" [16], popular items generally have a higher probability of being recommended to users, and they in fact tend to be recommended much more frequently than their original popularity in the dataset [1, 2, 16], even resulting in unfair recommendations [12, 16, 28]. The initial popularity bias may further have a self-reinforcing feedback effect [17]. To mitigate the issues caused by the popularity bias, several methods have been proposed based on regularization [2, 7, 38, 54], adversarial training [23], counterfactual reasoning [28, 52], and re-ranking [1]. However, past techniques are designed for popularity debiasing in conventional recommender systems. In this work, we focus on popularity debiasing a Human-in-the-Loop CRS. We propose a novel reward function for the dialogue state management module to dynamically mitigate popularity bias during decision making.

3 PRELIMINARIES

In this paper, we follow the System Asks – User Responds (SAUR) paradigm [49] to formulate the conversational recommendation problem. The goal of a conversational recommender system (CRS) is to make recommendations at a proper time by asking questions to estimate user preferences from both the multi-turn dialogue and the user's historical records.

Formally, let $\mathcal{U}$ and $\mathcal{V}$ denote the user and item sets, respectively. For each user $u \in \mathcal{U}$, the historical records constitute a subset of items denoted by $V_u \subseteq \mathcal{V}$. To elicit user preferences, the CRS agent asks a question q at each turn selected from the candidate question set $\mathcal{Q} = \{q_1, q_2, \ldots, q_m\}$, each of which is associated with a domain-specific attribute. The user will provide a response p to the question that may directly answer it with an attribute value, or express no preference on the attribute.

The conversation proceeds as follows. Let p_0 denote the initial preference description of a user u as the first query to the CRS agent. Upon completing the t-th turn of the conversation, the agent aims to select the next action a_{t+1} (asking a question or making recommendations) based on the current dialogue state $\mathbf{s}_t = (V_u, \{p_0, \ldots, p_t\}, \{q_1, \ldots, q_t\})$. If the agent decides to request further information about the user's preferences, it will choose the best question q_{t+1} from the set of remaining questions $\mathcal{Q} \setminus \{q_1, \ldots, q_t\}$. If it instead decides to provision a recommendation $\hat{V}_u$, it will pick top K items from the candidate set $\mathcal{V} \setminus V_u$ and display them to the user. The CRS agent keeps communicating with the user until it decides to make a recommendation or the user quits the conversation due to impatience, so in practice the conversational turns should not be designed to continue indefinitely. If the user rejects all recommended items, many systems can be configured to either continue to ask new questions or politely end the conversation [24, 36, 49]. In our experiments, we set a threshold on the maximum number of conversational turns in order to imitate the process of impatient users quitting the conversation.

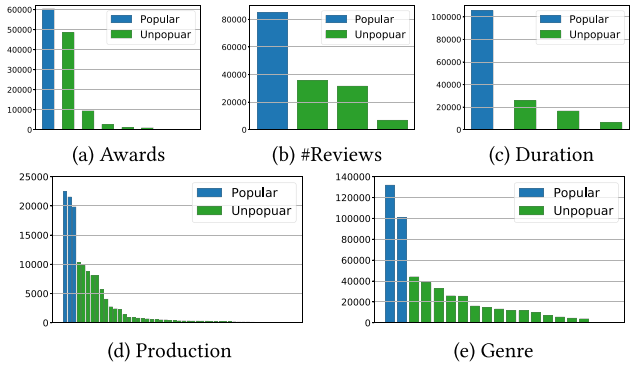
4 POPULARITY BIAS IN CRS

4.1 Data-driven Study

In our setting, the CRS agent interacts with users by asking questions towards specific attributes, and the users may respond with their preferences on attribute values. To investigate the popularity bias in the CRS, we leverage the aspect-value distribution of the items. We split the items in the training set into popular and unpopular groups in terms of the value of attributes. As illustrated in Fig. 2 about a *Movie* dataset, the *production* attribute has over 15 different values, but *Columbia Pictures*, *Lions Gate Films*, and *Disney* dominate, accounting for more than 50% of frequencies in the training data records. A similar *long-tail* effect can also be observed for other attributes and for the *Yelp* dataset introduced later in Section 6. The statistics demonstrate that due to the distribution disparity within the values of attributes, we could classify the items in terms of certain attribute values into either popular or unpopular groups. Hence, we claim that this divergence in the attribute value distributions is an essential factor leading to popularity bias.

4.2 Quantifying Popularity Bias

Formally, let $\mathcal{A}$ be a set of m attributes associated with the item set. Given a set of items $V \subseteq \mathcal{V}$ and an attribute $a \in \mathcal{A}$, we denote by $V(a) \subseteq V$ the subset of items whose values on the attribute a is not empty. According to the statistics in the previous section, we can derive two disjoint sets from $V(a)$, one for popular items and the other for unpopular items with respect to the attribute a. In the



(a) Awards (b) #Reviews (c) Duration

(d) Production (e) Genre

Figure 2: Distribution of item frequencies over different attributes.

recommendation scenario, for a specific recommendation set $\hat{V}_u$, we define the probability of resulting popularity bias for attribute $a_i \in \mathcal{A}$ on the recommendations as follows.

$$P(a_i|\hat{V}_u) = \frac{1}{Z}\frac{|\hat{V}_u^{\text{pop}}(a_i)|}{|\hat{V}_u(a_i)|}, \quad i = 1, \ldots, m \tag{1}$$

Here, $\hat{V}_u^{\text{pop}}(a_i)$ is the popular item set derived from $\hat{V}_u(a_i)$, and Z is the normalization term.

Furthermore, motivated by the classical information retrieval evaluation method NDCG [21], we can also define a utility function to quantify the influence degree of popularity bias for each attribute a_i over the recommendations, denoted by $\rho(a_i, \hat{V}_u)$:

$$\rho(a_i, \hat{V}_u) = \sum_{v \in \hat{V}_u} \frac{\mathbb{1}[v \in \hat{V}_u^{\text{pop}}(a_i)]}{\log(\pi(v) + 1)}, \tag{2}$$

where $\mathbb{1}[\cdot] \in \{0, 1\}$ denotes the indicator function and $\pi(v)$ is the rank of v. Note that this utility function always yields a positive value indicating the degree of popularity bias.

Therefore, to measure the overall effect of popularity bias on the recommendation at the t-th turn, we define the expected popularity bias as follows.

$$\mathbb{E}_{A_t}[\hat{V}_{u,t}] = \sum_{i=1}^{m} \mathbb{1}[a_i \in A_t]\, \rho(a_i, \hat{V}_{u,t})\, P(a_i|\hat{V}_{u,t}) \tag{3}$$

A_t is the set of attributes associated with the invoked questions in the t-round conversation, while $\hat{V}_{u,t}$ is the set of potential CRS recommendations made at turn t that are not actually shown to the user. In the CRS, Eq. 3 may be employed to indicate the expected influence of popularity bias that the unused attributes $\{a|a \in \mathcal{A}, a \notin A_t\}$ will bring to the current recommendations.

4.3 Evaluation Metrics

We employ two metrics to measure the attribute-based popularity bias in the recommendations.

Personalized Average Recommendation Popularity (PARP). We first explore and adapt the personalized popularity distribution of recommended items, which is defined as:

$$\text{PARP} = \frac{1}{|\mathcal{U}|}\sum_{u \in \mathcal{U}}\frac{1}{|\hat{V}_u|}\sum_{v \in \hat{V}_u}\frac{|\mathcal{A}_{\text{pop}}(v)|}{|\mathcal{A}(v)|}, \tag{4}$$

where $\hat{V}_u$ denotes the recommendation list for user u, $\mathcal{A}(v)$ is the set of available attributes of item v, and $\mathcal{A}_{\text{pop}}(v)$ is the subset of $\mathcal{A}(v)$ such that the attribute values are popular. Note that a higher PARP value indicates less exposure of unpopular items. The PARP scores are within the range of $[0, 1]$ and the smaller the value, the better the performance of the popularity debiasing.

Popularity-rank correlation for users (PRU). PARP neglects the ranking position of the recommended items. Similar to [54], we adopt another metric called *Popularity-Rank correlation for Users* (PRU) to measure the popularity bias in recommendation defined as follows:

$$\text{PRU} = -\frac{1}{|\mathcal{U}|}\rho(\text{pop}(O_u), \pi(O_u, \hat{V}_u)), \tag{5}$$

where O_u) is the ground truth set of items, $\rho(\cdot, \cdot)$ is *Spearman's Rank Correlation*, $\text{pop}(\cdot)$ calculates the popularity of each item in O_u, and $\pi(\cdot, \cdot)$ denotes the rankings of the ground truth items in the recommendations.

5 OUR METHOD

In this section, we introduce the proposed Popcorn CRS, which consists of three components, as illustrated in Fig. 3, including a user interaction modeling module (UIM), a dialogue state management module with popularity debiasing (DSM-PD), and a bias-aware recommendation module (REC). The UIM module encodes the open-ended descriptive language from user responses and questions to extract pertinent semantic cues. It also integrates such information with a user's historical records by automatically determining to what extent they can contribute towards estimating the user preferences. Subsequently, the UIM's outputs are relayed to the DSM-PD to update the dialogue state and select the next action in each conversational turn. In particular, the latter must decide whether to propose a question regarding potential attributes or to issue a particular recommendation considering popularity bias factors. Finally, the REC module makes recommendations and shows the results to the user with a reranking mechanism to balance between popular and unpopular items.

5.1 User Interaction Modeling

The intelligent agent communicates with the user interactively in natural language to solicit information about their current preferences during the conversational turns. It is crucial to understand the user's responses and keep track of the conversation so that the DSM module can select suitable next actions. Some systems [24, 36] encode the user's response into a set of predefined attribute–value pairs. However, in practice, a user's description of their interests can be vague and often fail to correspond to any pre-defined templates. Thus, we propose the user-interaction modeling module that captures implicit semantic cues in the user responses as well as historical records.

Given a dialogue up to the t-turn $\{p_0, q_1, p_1, \ldots, q_t, p_t\}$, we first encode the semantics of a user's real-time preferences as a latent vector. We represent each question or response as a sequence of l words and concatenate the question and the response at each turn with a special separator "[SEP]", resulting in t sentences denoted by $\{s_i\}_{i \in [t]}$. We use a d-dimensional pre-trained word embedding model to vectorize each word in the input and compute

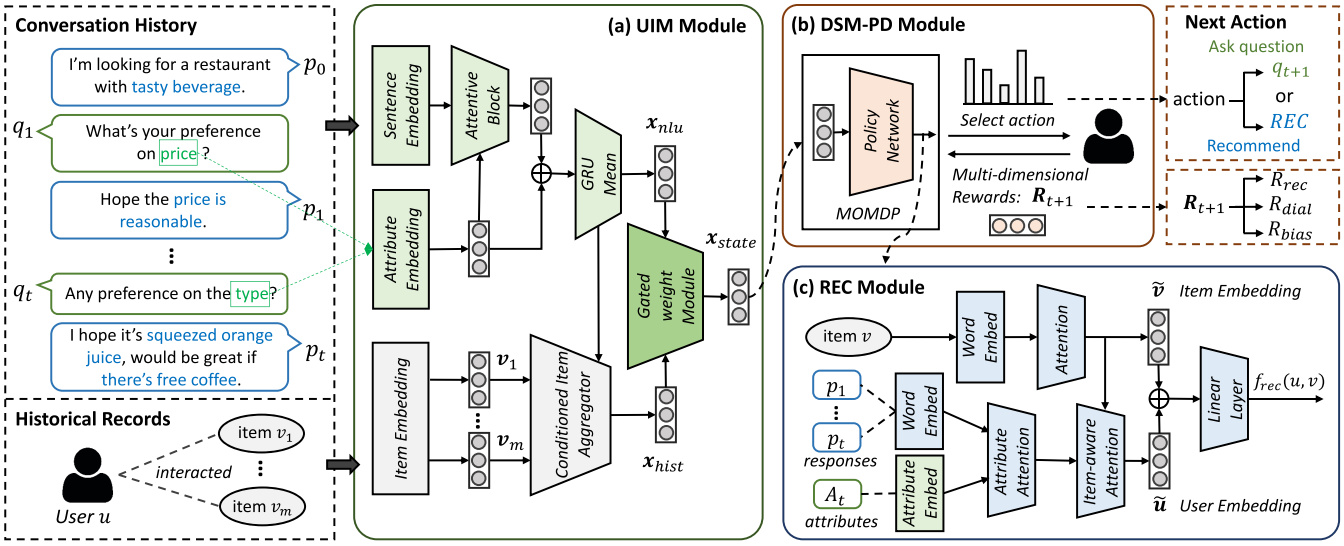


Figure 3: Workflow of our proposed Conversational Recommender System with HitL Popularity Debiasing.

their average to obtain the sentence representation,[1] denoted by $\{\mathbf{s}_i\}_{i\in[t]}$ with $\mathbf{s}_i \in \mathbb{R}^d$. Let $\mathbf{S} \in \mathbb{R}^{t\times d}$ be a matrix encoding t input utterances. At the same time, each question is associated with an attribute, which may provide structured information on potential items. Thus, we also incorporate an attribute representation into the dialogue. Let $\{a_1, \ldots, a_t\}$ be the sequence of attributes corresponding to t questions. We embed each attribute in a d-dimensional space and combine all the attribute vectors into a matrix $\mathbf{A} = [\mathbf{a}_1, \ldots, \mathbf{a}_t] \in \mathbb{R}^{t\times d}$. In order to detect the importance and usefulness of the utterances in the past t turns, we use a variant of the self-attention block [37] to learn attentive information over the t-turn dialogue:

$$\mathbf{x}_{\text{nlu}} = \text{mean}\left(\text{GRU}([\text{SelfAtt}(\mathbf{S}, \mathbf{A}, \mathbf{S}); \mathbf{A}])\right), \tag{6}$$

where both dialogue matrix $\mathbf{S}$ and attribute matrix $\mathbf{A}$ are encoded into the latent semantic vector $\mathbf{x}_{\text{nlu}} \in \mathbb{R}^d$.

Next, we connect real-time user information with historical records to obtain a more informed estimate of the user's preferences. The goal is to select the most salient features that capture current user preferences. Suppose that user u has interacted with m historical items, i.e., $V_u = \{v_1, \ldots, v_m\}$. We encode each item $v_j \in V_u$ into a vector denoted by $\mathbf{v}_j,\ j \in [m]$. In order to know which items are useful to model the user's current preferences conditioned on the real-time dialogue information, we propose a component called *Conditioned Item Aggregator* (CIA), which learns an attention-based importance factor α_j of each item v_j automatically.

$$z_j = \langle \text{ReLU}(\mathbf{v}_j)\mathbf{W}_3, \text{ReLU}(\mathbf{x}_{\text{nlu}})\mathbf{W}_4 \rangle \tag{7}$$

$$\alpha_j = \frac{e^{z_j}}{\sum_{k=1}^{m} e^{z_k}} \tag{8}$$

Here, $\langle , \rangle$ is the dot product operation, $\mathbf{W}_3, \mathbf{W}_4 \in \mathbb{R}^{d\times d}$ are learnable parameters that map two vectors $\mathbf{v}_j$ and $\mathbf{x}_{\text{nlu}}$ into the same space.

[1]We adopt GloVe [31] as the default word embedding, but it can be replaced by any other state-of-the-art language model. The selection of word embeddings is not the focus of this work.

Therefore, the historical user preferences can be represented by the weighted sum over all item embeddings:

$$\mathbf{x}_{\text{hist}} = \text{CIA}(\{v_j\} \mid \mathbf{x}_{\text{nlu}}) = \sum_{j=1}^{m} \alpha_j \mathbf{v}_j, \tag{9}$$

where $\mathbf{x}_{\text{hist}}$ is the output vector that captures historical records.

To aggregate both real-time and historical information, we introduce the *Gated Weight Module* (GWM) to encode the state vector by automatically balancing between the real-time information and historical user preferences. Specifically, it learns a weighting factor

$$\beta = \sigma(\text{ReLU}([\mathbf{x}_{\text{nlu}}; \mathbf{x}_{\text{hist}}]\mathbf{W}_5)\mathbf{W}_6), \tag{10}$$

where $\mathbf{W}_5 \in \mathbb{R}^{d\times d}, \mathbf{W}_5 \in \mathbb{R}^{d\times 1}$ are learnable parameters. Then, the final state vector can be computed as follows:

$$\mathbf{x}_{\text{state}} = \text{GWM}(\mathbf{x}_{\text{nlu}}, \mathbf{x}_{\text{hist}}) = \beta\mathbf{x}_{\text{nlu}} + (1-\beta)\mathbf{x}_{\text{hist}} \tag{11}$$

This latent state vector encodes rich information regarding user preferences from both the ongoing real-time conversation and from the long-term historical behavior, and the balance between them is automatically adjusted by the weighting factor β.

5.2 Dialogue State Management with Popularity Debiasing

The goal of a traditional dialogue state management module is to take as input the semantic information encoded by the user interaction module and determine the optimal actions (e.g., asking proper questions or making recommendations) leading to a promising recommendation. However, existing work neglects the issue of popularity bias during action selection, which may be detrimental to the final recommendation performance and lead to an imbalanced exposure of less popular items. To address this, we formulate the DSM-PD module as a *multi-objective Markov decision process* (MOMDP) [34] and seek to learn an optimal policy that achieves a good balance between both objectives. Formally, we define the MOMDP as follows.

State The state $\mathbf{s}_t = \mathbf{x}_{\text{state}}$ is the vector computed by the UIM module. Unlike previous work, in which each dimension of the state vector corresponds to a predefined attribute–value pair, the state is represented based on implicit semantic features of the user's descriptive response and user preferences estimated from the current dialogue as well as historical records.

Action space There are two main types of actions in our system, i.e., asking a new question and making a recommendation, similar to previous work [36, 41, 49]. The size of the action space is equal to one plus the number of questions remaining in the candidate pool.

Rewards In the MOMDP, the reward is defined to be a vector where each component represents one type of feedback from the environment with respect to a specific objective. In the CRS scenario, we consider three important factors related to the recommendation performance, user experience, and item popularity bias. Specifically, at the t-th turn of the conversation, an immediate reward vector $\mathbf{R}_t \in \mathbb{R}^3$ is assigned to the action when the user's response is collected:

$$\mathbf{R}_t = [R_{\text{rec},t}, R_{\text{dial},t}, R_{\text{bias},t}]. \tag{12}$$

The first reward function $R_{\text{rec},t}$ refers to the recommendation success state, with value r_{fail} if the user rejects the recommendation or directly leaves, or value r_{success} if the user accepts the recommendation. The second reward function $R_{\text{dial},t}$ is associated with the user experience in each dialogue. It elicits the value r_{reply} if the user provides a preference answer to an attribute-based question asked by the agent, or r_{empty} if the user does not provide a response, indicating that the user does not have specific preference over the attribute. The third reward $R_{\text{bias},t} = \exp(-\mathbb{E}_{A_t}[\hat{V}_{u,t}])$ provides a signal regarding the popularity bias defined in Eq. 3 over the remaining attributes $\mathcal{A} \setminus A_t$ not associated with the past questions.

Objectives The DSM's goal turns to learning an optimal policy π_θ parametrized by θ that simultaneously maximizes three objectives:

$$\mathbf{J}(\pi_\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T-1} \gamma^t \mathbf{R}_{t+1}\right], \tag{13}$$

where $\mathbf{J}(\pi_\theta) \in \mathbb{R}^3$ and γ is a discount factor that dampens the influence of long-term rewards. To optimize such a vectorized objective, a naive approach is to use a weighted sum to aggregate all three objectives. However, as shown in previous work [34], such linear combinations fail to provide sufficient control to balance between different objectives, and thus a non-linear mapping over multiple objectives is preferred. Inspired by previous work [40], we introduce a non-linear generalized Gini social welfare function G_{swf} and cast the learning objective of CRS as:

$$\pi_\theta^* = \arg\max_{\pi_\theta} G_{\text{swf}}(\mathbf{J}(\pi_\theta)), \tag{14}$$

$$G_{\text{swf}}(\mathbf{x}) = \sum_{i=1}^{3} \mathbf{w}_i \tilde{\mathbf{x}}_i, \tag{15}$$

where $\tilde{\mathbf{x}}$ is the vector whose components are copied from $\mathbf{x}$ but sorted in ascending order, and $\mathbf{w} \in [0,1]^3$ is a weight vector satisfying $\sum_{i=1}^{3} \mathbf{w}_i = 1$. The optimal policy π_θ^* in Eq. 14 can provably [40] achieve Pareto optimality, i.e., a good balance among recommendation performance, user experience, and item popularity bias.

To solve the optimization of Eq. 14, we define a variant of policy gradient derived as follows.

$$\nabla_\theta G_{\text{swf}}(\mathbf{J}(\pi_\theta)) = \nabla_{\mathbf{J}(\pi_\theta)} G_{\text{swf}}(\mathbf{J}(\pi_\theta)) \cdot \nabla_\theta \mathbf{J}(\pi_\theta) \tag{16}$$

$$= \tilde{\mathbf{w}}^T \nabla_\theta \mathbf{J}(\pi_\theta), \tag{17}$$

where $\tilde{\mathbf{w}}$ is a vector with components derived and sorted from weight vector $\mathbf{w}$. Note that $\nabla_\theta \mathbf{J}(\pi_\theta) \in \mathbb{R}^{3\times|\theta|}$ is equivalent to the classic policy gradient over 3 objectives. The DSM module thereby gradually learns to select the optimal actions based on the current state in consideration of multiple factors.

5.3 Recommendation

In our model, the recommendation module (REC) is invoked when the DSM-PD module decides to make a recommendation. REC measures the relevance between the user and the candidate items based on the conversation history and item features, and selects the top items most likely to satisfy the user's needs. Both the user preferences and the item features are captured by understanding the semantics of natural language inputs. The user representation is generated based on the conversation history and the item representation is generated based on its textual description. Suppose the DSM-PD module decides to make a recommendation at turn t given the collected t responses $\{p_i\}_{i\in[t]}$.

Item Representation. Different from dialogue utterances that tend to be fairly short within a single conversation, the number of candidate items could be very large in real-world scenarios and the REC module has to measure the relevance between the user and every candidate item. Therefore, we aim to generate the item representation from its description in a simpler way to balance efficiency and effectiveness. Specifically, we first encode the description of an item $v \in \mathcal{V}$ into a sequence of word embeddings $\{\mathbf{d}_j \in \mathbb{R}^d\}_{j\in[l]}$. Then, a weighted average of word embeddings yields the item representation $\tilde{\mathbf{v}}$ with an attention mechanism, which has been shown effective in previous work [8, 22].

$$\tilde{\mathbf{v}} = \sum_{j}^{l} \alpha_j \mathbf{d}_j,\ \alpha_j = \text{softmax}(\mathbf{v}_s^\intercal \tanh(\mathbf{W}_s \mathbf{d}_j)) \tag{18}$$

Here, $\mathbf{W}_s \in \mathbb{R}^{d\times d}, \mathbf{v}_s \in \mathbb{R}^d$ are parameters of the attention module.

User Representation. Each response p_i reveals the user preference on the question q_i at turn $i \in [t]$, and hence the user representation should explicitly encode the information of all such preferences. We first encode each response p_i of l words into a sequence of word embeddings $\mathbf{P}_i \in \mathbb{R}^{l\times d}$. The self-attention block is again recruited to encode the word sequence of each response, i.e., $\tilde{\mathbf{P}}_i = \text{SelfAtt}(\mathbf{P}_i, \mathbf{P}_i, \mathbf{P}_i) \in \mathbb{R}^{l\times d}$. However, not all words in the response are equally important in expressing the user's preferences, and the importance of a single word may also depend on the particular attribute. Therefore, when generating the representation of response p_i, we apply another attribute-aware attention mechanism [30] to enable the model to focus on more informative words:

$$\mathbf{r}_i = \sum_{j}^{l} \alpha_{i,j} \mathbf{h}_{i,j},\ \alpha_{i,j} = \text{softmax}(\mathbf{v}_r^\intercal \tanh(\mathbf{W}_r \mathbf{h}_{i,j} + \mathbf{W}_a \mathbf{a}_i)) \tag{19}$$

Dataset	#Users	#Items	#Attr	#Conv	Avg. Turns
Yelp	103,189	37,164	8	239,788	6.41
Movie	6,038	3,378	15	574,965	6.62

Table 1: Statistics of the datasets. "Attr" represents attributes and "Conv" represents conversations.

Here, $\mathbf{h}_{i,j} \in \mathbb{R}^d$ is the hidden state of the j-th word in the user's response and $\mathbf{a}_i \in \mathbb{R}^{d_a}$ is the attribute embedding at turn i. $\mathbf{W}_r \in \mathbb{R}^{d\times d}, \mathbf{W}_a \in \mathbb{R}^{d_a\times d}, \mathbf{v}_r \in \mathbb{R}^d$ are parameters of the attention module. The output $\mathbf{r}_i \in \mathbb{R}^d$ is the representation of the user's response at turn i.

To summarize the information in the conversation history up to turn t, all responses $\{\mathbf{r}_i\}_{i\in[t]}$ of user u are encoded into a dense representation via item-aware attention with a candidate item v:

$$\tilde{\mathbf{u}} = \sum_i^t \alpha'_i \mathbf{r}_i,\ \alpha'_t = \text{softmax}(\mathbf{v}_u^\top \tanh(\mathbf{W}_u \mathbf{r}_i + \mathbf{W}_i \tilde{\mathbf{v}})) \tag{20}$$

Here, $\mathbf{W}_u \in \mathbb{R}^{d\times d}, \mathbf{W}_i \in \mathbb{R}^{d\times d}, \mathbf{v}_u \in \mathbb{R}^d$ are parameters of the attention module.

Prediction. We predict the relevance score between a user u and an item v via concatenation of their representations followed by a linear layer

$$f_{\text{rec}}(u, v) = \mathbf{v}_y[\tilde{\mathbf{u}}; \tilde{\mathbf{v}}] + b_y, \tag{21}$$

where $\mathbf{v}_y \in \mathbb{R}^{2d}, b_p \in \mathbb{R}$ are parameters for this linear layer.

We use the objective of Bayesian Personalized Ranking (BPR) [32] to learn the parameters in the recommendation module, i.e.,

$$\mathcal{L}_{\text{rec}}(\Theta) = \sum_{(u,v^+,v^-)} \log \sigma(f_{\text{rec}}(u, v^+) - f_{\text{rec}}(u, v^-)), \tag{22}$$

where v^+ and v^- refer to a positive item and a sampled negative item for the user u

6 EXPERIMENTS

6.1 Experimental Setup

Datasets. To validate our proposed CRS popularity debiasing framework, we consider a publicly available dataset in the movie domain, constructed for developing a CRS with state control to learn an action selection strategy [36] based on the *Movielens* [20] dataset. We also construct a second dataset based on real-world data from Yelp[2], which provides user reviews of hotels and restaurants. Specifically, we follow the approach of Zhang et al. [49] and extract attribute-related content within reviews as user responses toward questions in a conversation. We extract aspects in reviews with a public toolkit [50], and further categorize these into a set of groups as attributes, e.g., "bread", "beef", and "shrimp" are classified as pertaining to the *Food* attribute, based on the assumption that similar aspects in the same group are relevant to the same topic. Thus, each review is transformed into a conversation with questions related to attributes and user responses are diverse naturally occurring sentences referencing corresponding aspects.

We additionally augment the conversational turns in both the *Movie* and *Yelp* datasets using template-based question generation with manual verification following the procedure proposed by [14]. It should be noted that for the dataset constructed by Sun [36], a set of attribute–value pairs are predefined and each of the user responses is related to one pair. Our constructed dataset does not rely on predefined knowledge and aims at evaluating the methods on more complex instances. Dataset statistics are given in Table 1.

We randomly split each dataset into training, development, and testing subsets with an 80%, 10%, 10% split. 50,000 samples of the training set and 5,000 samples of the testing and development sets are used for learning the policy network and the remainder is exploited for recommendation learning. Unlike previous work [24, 36], our system does not depend on prior knowledge of attribute–value pairs, and the UIM module is optimized together with the policy network without a pretraining process.

Implementation Details. In the UIM module, the embedding size is $d = 300$. In the DSM-PD module, the policy network receives rewards $r_{\text{SUCCESS}} = 1$, $r_{\text{FAIL}} = -1$, $r_{\text{REPLY}} = -0.1$, $r_{\text{EMPTY}} = -0.5$. The weight vector in the function G_{swf} is $[0.5, 0.25, 0.125]$ before normalization, corresponding to the objectives with rewards $[R_{\text{rec}}, R_{\text{bias}}, R_{\text{dial}}]$. The policy network is modeled as a two-layer neural network size with parameter sizes of 300×64 and $64 \times |\mathcal{A}|$. Three modules are jointly trained with Adam optimization with a learning rate of 2×10^{-4}. The REC module is pretrained with Adam optimization with an initial learning rate of 10^{-3}. We tune the model on the development set and use it to report the test results.

Evaluation Metrics. For recommendation performance, we measure *the success rate of the recommendation before achieving the maximum conversation turn* and *the average number of turns to achieve a successful recommendation* as metrics. Specifically, the Success rate (SR) @10 and Mean reciprocal rank (MRR) are used to assess the overall recommendation quality. Moreover, we report the results of our proposed measurement **PARP** and the previously established **PRU** to evaluate the popularity bias, as defined in Section 4.3.

Baselines. We compare POPCORN with several state-of-art baseline approaches. **Max Entropy (MaxEnt)** [36] is the rule-based strategy of asking for the attribute with the maximum entropy according to the candidate items. This strategy requires a knowledge base, in which each item is related to a group of attribute–value pairs. When the user specifies the preferred value of an attribute, items that contain other values are eliminated from the candidate set. The entropy of each attribute is recomputed at each turn. **CRM** [36] is the state-of-the-art CRS with natural language understanding and state control. It learns a belief tracker to encode the user's response to a set of attribute–value pairs as the dialogue state. **MMN** [49] is a Multi-Memory Network learning a supervised model to ask questions and perform recommendation based on semantic features extracted from the conversation. **EAR** [24] can be regarded as the state-of-the-art multi-round conversational recommender system. It relies on an Estimation–Action–Reflection approach for the interaction between conversation and recommendation modules, but it does not consider that users may respond with vague utterances and it disregards historical records entirely. Note that *MaxEnt*, *CRM*, and *EAR* assume that items and user responses are strictly associated with the predefined attribute–value pairs, which are not available in our setting. In order to enable an experimental

[2]https://www.yelp.com/dataset/

Methods	Movie				Yelp			
	SR↑	MRR↑	PARP↓	PRU↓	SR↑	MRR↑	PARP↓	PRU↓
MaxEnt	0.548	0.324	0.847	0.909	0.464	0.301	0.794	0.921
CRM	0.656	0.470	0.754	0.822	0.649	0.314	0.709	0.793
MMN	0.622	0.362	0.801	0.830	0.658	0.343	0.713	0.840
EAR	0.787	0.583	0.797	0.764	0.673	0.349	0.694	0.802
POPCORN	**0.813**	**0.608**	**0.513**	**0.317**	**0.676**	**0.357**	**0.522**	**0.367**

Table 2: Overall recommendation performance comparison on the two benchmark datasets. Success rate (SR) and Mean reciprocal rank (MRR) are used as evaluation metrics. The best performance is in boldface. We report the best recommendation results of 7 conversational rounds on the Movie dataset and 5 conversational rounds on Yelp. We also report the PARP and PRU for the best achieved MRR.

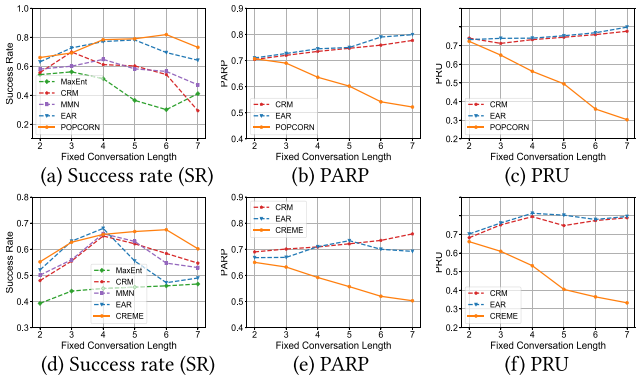
comparison, we modify the inputs of these models by replacing the predefined values with the dialogue semantic vector $\mathbf{x}_{\text{nlu}}$ generated by our UIM module. This guarantees that no prior knowledge of the attribute–value pairs can leak to the conversational agents.

6.2 Overall Results

As shown in Table 2, we compare different methods from two perspectives: (1) recommender performance, including whether it is able to attain a higher success rate with the same number of turns (accuracy) as well as whether it is able to perform a successful recommendation in fewer turns, and (2) whether the top-k ($k = 10$) recommended results contain more unpopular items in terms of our proposed metric PARP. Since all the baseline methods neglect the popularity of the recommendation decisions, our proposed method POPCORN outperforms all the others in terms of popularity bias. Note that all experimental improvements of POPCORN over other baselines are statistically significant for $p < 0.01$.

Further, POPCORN also outperforms the best-performing baseline approach *EAR* in terms of the recommendation quality on both *Movie* and *Yelp*. Note that *EAR* could be viewed as a practical upper bound of a real-world CRS that does not attempt to interpret varied linguistic utterances[3]. However, our model considering the user's semantic information, historical records, and popularity bias still outperforms *EAR*. A possible reason is that the conversational turns in *EAR* are defined as a set of correct attribute candidates using language templates, which greatly reduces the amount of complex semantics. The improvement of our method compared with *EAR* shows that it is beneficial to integrate semantic understanding, explicit user history, and debiasing techniques within a single CRS. We further find that although *MaxEnt* for attribute selection is an effective strategy, it is only based on prior knowledge of items (i.e., attribute values of items). As we can see, the RL-based method *CRM* is substantially more competitive than *MaxEnt*, which shows that it can select better questions based on the conversational history. *MMN* learns a question selection model in a supervised way, while the order of turns in the training data may not always be ideal. Compared with all the baselines, our method is able to explore a strategy automatically instead of following the order of questions in the training corpus and achieves a higher recommendation accuracy on both datasets.

[3] Although one approach [25] outperforms EAR, it achieves this by utilizing KG structure, while POPCORN does not use any external knowledge.



(a) Success rate (SR) (b) PARP (c) PRU

(d) Success rate (SR) (e) PARP (f) PRU

Figure 4: Recommendation performance and popularity bias metrics under different fixed lengths of conversations.

6.3 Analysis for Fixed Conversation Lengths

Moreover, we also evaluate the effect of popularity bias for different conversational lengths. As shown in Figure 4, where the CRS frameworks are required to provide recommendation results given the specific conversational turn. Together with Table 2, both demonstrate that our system achieves a high success rate on recommendations compared to baseline methods conducting one additional conversational turn on average. For example, on the *Movie* dataset, the success rate of our system surpasses 80% within 6 turns. Moreover, as more conversational turns are involved, the substantial improvement of our CRS framework compared with prior work confirms the effectiveness of the popularity debiasing method. This is because our debiasing mechanism is integrated with each HitL conversational turn such that the DSM module can ask suitable questions in each round and avoid pruning off items with respect to unpopular attribute values.

However, we can infer that it is impractical to let a conversation proceed for an overly long number of turns. If we force the agent to keep asking questions, the superfluous conversational turns may end up introducing noise into the decision making process of the CRS. Nevertheless, our framework exhibits better robustness compared to other baselines as the number of turns grows, as it keeps mitigating the popularity bias.

6.4 Analysis with Flexible Number of Turns

Note that all experimental results in Table 2 are based on the assumption that the user will not quit the conversation until the maximum turn is reached and we record the corresponding average best recommendation performance and corresponding popularity debiasing performance within the fixed number of turns. In practice, users usually become impatient when interacting with an intelligent agent after a number of conversational rounds. We take this scenario for further analysis. Referring to rows 1, 2, 3 in Table 3, POPCORN is able to achieve the best performance when systems are able to choose to automatically exit the conversations. This allows the system to present successful recommendations, while allowing slightly more rounds of interactive turns with the user. This appears intuitive for popularity debiasing purposes, as the CRS framework should be permitted to ask additional questions to

Methods	Movie					Yelp				
	SR↑	Turns↓	MRR↑	PARP↓	PRU↓	SR↑	Turns↓	MRR↑	PARP↓	PRU↓
1. CRM	0.614	**4.06**	0.405	0.744	0.741	0.645	4.92	0.284	0.726	0.730
2. EAR	0.760	5.13	0.447	0.749	0.779	0.615	**3.94**	0.301	0.738	0.802
3. POPCORN	**0.794**	5.72	**0.501**	**0.528**	**0.492**	**0.679**	5.33	**0.344**	**0.534**	**0.381**
4. POPCORN w/ final debias	0.797	5.24	0.509	0.694	0.644	0.718	4.82	0.347	0.702	0.614
5. POPCORN w/o debias	0.825	5.01	0.533	0.774	0.784	0.723	4.35	0.359	0.753	0.813
6. POPCORN w/o history	0.733	5.95	0.474	0.499	0.447	0.648	5.62	0.313	0.397	0.301
7. POPCORN w/o history & debias	0.808	5.88	0.524	0.690	0.715	0.686	5.41	0.350	0.702	0.690

Table 3: Evaluation of the CRS when DSM automatically decides to make recommendations. Success rate (SR) and Mean reciprocal rank (MRR) are used as recommendation evaluation metrics. We also report the PARP and PRU. All the metrics are reported based on the average number of turns (Turns). Best results among methods 1–3 are highlighted in boldface.

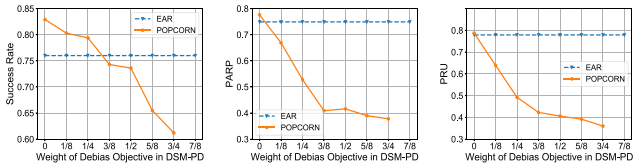
capture user preferences in order to make better trade-offs between the recommendation utility and popularity bias.

6.5 Analysis of Debiasing Strategies

Debiased DSM-PD. For further analysis, we compare the effectiveness of different popularity debiasing strategies within POPCORN, as shown in Table 3. Specifically, we consider the effect of eliminating the popularity debiasing, while leaving all other settings unchanged. Further, we also assess to what extent the CRS framework benefits from applying the debias module over each conversational turn as opposed to just the last. The results, given by rows 4 and 5 compared with row 3 in Table 3, lead to two notable observations. First, without popularity debiasing in the DSM module, the recommendation results in row 4 will be as biased as the baselines, as it predominantly considers the signals of popular attribute values during the conversational turns. Although the CRS framework achieves the best recommendation utility, it obtains a sub-optimal popularity bias, as the PARP and PRU results are not comparable with row 3. We further consider another training pipeline conducting popularity debiasing only on the final conversational turn. The result, shown in row 5, reveals a less satisfactory popularity bias compared to row 4. Intuitively, the popularity bias is amplified during the interaction with users if no debiasing is invoked to guide the selection of attributes to ask the user about.

Effect of User Histories. Next, we study how users' historical records influence the model performance. Specifically, as shown in row 6 of Table 3, we assess the effect of eliminating the explicit historical records. Given the lower recommendation quality in row 6 compared to our proposed version in row 3 along with the improved popularity bias results, we conclude that with less historical information the model cannot accurately know to what extent the user is expressing new desires or instead might have retained certain prior preferences. We further in row 7 report the results obtained when neither explicit user history, nor a user interaction modeling (UIM) module, nor any popularity debiasing in the DSM module are included. We observe that it retains the recommendation performance while the popularity bias results deteriorate. This is intuitive, since there is no debiasing operation.

Ablation Study. As shown in Figure 5 (a), the recommendation performance decreases as the weight of the debiasing objective in DSH-PD grows. However, the popularity bias gets alleviated as this weight increases in Figure 5 (b) and (c). Intuitively, there is a trade-off between the two goals in the DSM module, so the ratio needs to be selected carefully by setting the weight of the debiasing objective within an acceptable range.



(a) Success rate (SR) (b) PARP (c) PRU

Figure 5: Recommendation performance (SR) and popularity bias (PARP, PRU) results on *Movie* dataset when varying the weight of the debiasing objective in DSH-PD.

7 CONCLUSION

In this paper, we shed light on the problem of human-in-the-loop popularity bias in conversational recommendation, where the interaction across different rounds may lead to biased recommendations. We first provide an empirical analysis and new metrics.

We then propose a new method called POPCORN consisting of three modules. The user-interaction modeling component takes both dialogue context and historical records as input and emits a semantic vector representing user preferences. Our unique debiased dialogue state management module then learns a bias-aware policy network under a multi-objective MDP to select next actions. At a suitable time, it triggers the final recommendation module to provide recommendation decisions. The experimental results on two conversational recommendation datasets establish the effectiveness of our model with regard to both the conversational recommendation quality and the success in debiasing the results.

8 ACKNOWLEDGMENTS

The authors thank Dr. Sixing Wu for the enlightening conversations and assistance during his visit at Rutgers. We also thank the reviewers for their valuable review suggestions.

REFERENCES

[1] Himan Abdollahpouri. 2019. Popularity bias in ranking and recommendation. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*. 529–530.

[2] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Controlling popularity bias in learning-to-rank recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 42–46.

[3] Himan Abdollahpouri, Masoud Mansoury, Robin Burke, and Bamshad Mobasher. 2019. The unfairness of popularity bias in recommendation. *arXiv preprint arXiv:1907.13286* (2019).

[4] Aman Agarwal, Kenta Takatsu, Ivan Zaitsev, and Thorsten Joachims. 2019. A general framework for counterfactual learning-to-rank. In *Proceedings of the 42nd International ACM SIGIR*. 5–14.

[5] Aman Agarwal, Ivan Zaitsev, Xuanhui Wang, Cheng Li, Marc Najork, and Thorsten Joachims. 2019. Estimating position bias without intrusive interventions. In *Proceedings of the Twelfth ACM WSDM*. 474–482.

[6] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and Debias in Recommender System: A Survey and Future Directions. *arXiv preprint arXiv:2010.03240* (2020).

[7] Zhihong Chen, Rong Xiao, Chenliang Li, Gangfeng Ye, Haochuan Sun, and Hongbo Deng. 2020. Esam: Discriminative domain adaptation with non-displayed items to improve long-tail performance. In *Proceedings of the 43rd International ACM SIGIR*. 579–588.

[8] Jin Yao Chin, Kaiqi Zhao, Shafiq Joty, and Gao Cong. 2018. ANR: Aspect-based neural recommender. In *Proceedings of the 27th ACM CIKM*. 147–156.

[9] Konstantina Christakopoulou, Alex Beutel, Rui Li, Sagar Jain, and Ed H Chi. 2018. Q&R: A two-stage approach toward interactive recommendation. In *Proceedings of the 24th ACM SIGKDD*. 139–148.

[10] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards conversational recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 815–824.

[11] Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 2017. Towards end-to-end reinforcement learning of dialogue agents for information access. *ACL* (2017).

[12] Zuohui Fu, Yikun Xian, Ruoyuan Gao, Jieyu Zhao, Qiaoying Huang, Yingqiang Ge, Shuyuan Xu, Shijie Geng, Chirag Shah, Yongfeng Zhang, and Gerard de Melo. 2020. Fairness-aware explainable recommendation over knowledge graphs. In *SIGIR*. 69–78.

[13] Z. Fu, Yikun Xian, Yongfeng Zhang, and Y. Zhang. 2020. Tutorial on Conversational Recommendation Systems. *Fourteenth ACM Conference on Recommender Systems* (2020).

[14] Zuohui Fu, Yikun Xian, Yaxin Zhu, Shuyuan Xu, Zelong Li, Gerard de Melo, and Yongfeng Zhang. 2021. HOOPS: Human-in-the-Loop Graph Reasoning for Conversational Recommendation. In *Proceedings of SIGIR 2021*. ACM.

[15] Ruoyuan Gao, Yingqiang Ge, and Chirag Shah. 2021. FAIR: Fairness-Aware Information Retrieval Evaluation. *arXiv preprint arXiv:2106.08527* (2021).

[16] Yingqiang Ge, Shuchang Liu, Ruoyuan Gao, Yikun Xian, Yunqi Li, Xiangyu Zhao, Changhua Pei, Fei Sun, Junfeng Ge, Wenwu Ou, et al. 2021. Towards Long-term Fairness in Recommendation. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 445–453.

[17] Yingqiang Ge, Shuya Zhao, Honglu Zhou, Changhua Pei, Fei Sun, Wenwu Ou, and Yongfeng Zhang. 2020. Understanding echo chambers in e-commerce recommender systems. In *SIGIR*. 2261–2270.

[18] Sahin Cem Geyik, Stuart Ambler, and Krishnaram Kenthapadi. 2019. Fairness-aware ranking in search & recommendation systems with application to linkedin talent search. In *KDD*. 2221–2231.

[19] Mark P Graus and Martijn C Willemsen. 2015. Improving the user experience during cold start through choice-based preference elicitation. In *Proceedings of the 9th ACM Conference on Recommender Systems*. 273–276.

[20] F. M. Harper and J. Konstan. 2015. The MovieLens Datasets: History and Context. *ACM Trans. Interact. Intell. Syst.* 5 (2015), 19:1–19:19.

[21] K. Järvelin and Jaana Kekäläinen. 2017. IR evaluation methods for retrieving highly relevant documents. *SIGIR Forum* 51 (2017), 243–250.

[22] Dongyeop Kang, Anusha Balakrishnan, Pararth Shah, Paul Crook, Y-Lan Boureau, and Jason Weston. 2019. Recommendation as a communication game: Self-supervised bot-play for goal-oriented dialogue. *EMNLP-IJCNLP* (2019).

[23] Adit Krishnan, Ashish Sharma, Aravind Sankar, and Hari Sundaram. 2018. An adversarial approach to improve long-tail performance in neural collaborative filtering. In *CIKM*. 1491–1494.

[24] 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 *WSDM*.

[25] 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. *KDD* (2020).

[26] J. Li, Alexander H. Miller, S. Chopra, Marc'Aurelio Ranzato, and J. Weston. 2017. Dialogue Learning With Human-In-The-Loop. *ArXiv* abs/1611.09823 (2017).

[27] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. In *Advances in neural information processing systems*. 9725–9735.

[28] Yunqi Li, Hanxiong Chen, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. 2021. User-oriented Fairness in Recommendation. In *Proceedings of the Web Conference 2021*. 624–632.

[29] Benedikt Loepp, Tim Hussein, and Jüergen Ziegler. 2014. Choice-based preference elicitation for collaborative filtering recommender systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 3085–3094.

[30] Thang Luong, Hieu Pham, and Christopher D. Manning. [n.d.]. Effective Approaches to Attention-based Neural Machine Translation. In *EMNLP*.

[31] Jeffrey Pennington, R. Socher, and Christopher D. Manning. 2014. Glove: Global Vectors for Word Representation. In *EMNLP*.

[32] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *UAI* (2012).

[33] Yuta Saito, Suguru Yaginuma, Yuta Nishino, Hayato Sakata, and Kazuhide Nakata. 2020. Unbiased recommender learning from missing-not-at-random implicit feedback. In *Proceedings of the 13th International Conference on WSDM*. 501–509.

[34] Umer Siddique, Paul Weng, and Matthieu Zimmer. 2020. Learning Fair Policies in Multi-Objective (Deep) Reinforcement Learning with Average and Discounted Rewards. In *International Conference on Machine Learning*. PMLR, 8905–8915.

[35] Harald Steck. 2013. Evaluation of recommendations: rating-prediction and ranking. In *Proceedings of the 7th ACM conference on Recommender systems*. 213–220.

[36] Yueming Sun and Yi Zhang. 2018. Conversational recommender system. In *The 41st international ACM SIGIR*. 235–244.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, L. Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. *ArXiv* abs/1706.03762 (2017).

[38] Jacek Wasilewski and Neil Hurley. 2016. Incorporating Diversity in a Learning to Rank Recommender System.. In *FLAIRS Conference*. 572–578.

[39] Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. 2017. A network-based end-to-end trainable task-oriented dialogue system. *EACL* (2017).

[40] John A Weymark. 1981. Generalized Gini inequality indices. *Mathematical Social Sciences* 1, 4 (1981), 409–430.

[41] Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. 2019. Reinforcement knowledge graph reasoning for explainable recommendation. In *SIGIR*. 285–294.

[42] Kerui Xu, Jingxuan Yang, Jun Xu, Sheng Gao, Jun Guo, and Ji-Rong Wen. 2021. Adapting User Preference to Online Feedback in Multi-round Conversational Recommendation. In *Proceedings of the 14th ACM WSDM*. 364–372.

[43] Shuyuan Xu, Yingqiang Ge, Yunqi Li, Zuohui Fu, Xu Chen, and Yongfeng Zhang. 2021. Causal Collaborative Filtering. *arXiv preprint arXiv:2102.01868* (2021).

[44] Longqi Yang, Yin Cui, Yuan Xuan, Chenyang Wang, Serge Belongie, and Deborah Estrin. 2018. Unbiased offline recommender evaluation for missing-not-at-random implicit feedback. In *RecSys*. 279–287.

[45] Hsiang-Fu Yu, Mikhail Bilenko, and Chih-Jen Lin. 2017. Selection of negative samples for one-class matrix factorization. In *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 363–371.

[46] Tong Yu, Yilin Shen, and Hongxia Jin. 2019. A visual dialog augmented interactive recommender system. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 157–165.

[47] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. 2013. Learning fair representations. In *ICML*. PMLR, 325–333.

[48] Yongfeng Zhang, Xu Chen, et al. 2020. Explainable Recommendation: A Survey and New Perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101.

[49] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W Bruce Croft. 2018. Towards conversational search and recommendation: System ask, user respond. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 177–186.

[50] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 83–92.

[51] Zheng Zhang, Minlie Huang, Zhongzhou Zhao, Feng Ji, Haiqing Chen, and Xiaoyan Zhu. 2019. Memory-augmented dialogue management for task-oriented dialogue systems. *ACM Transactions on Information Systems (TOIS)* 37, 3 (2019).

[52] Yu Zheng, Chen Gao, Xiang Li, Xiangnan He, Depeng Jin, and Yong Li. 2020. Disentangling User Interest and Conformity for Recommendation with Causal Embedding. *arXiv preprint arXiv:2006.11011* (2020).

[53] 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. *KDD* (2020).

[54] Ziwei Zhu, Yun He, Xing Zhao, Yin Zhang, Jianling Wang, and James Caverlee. 2021. Popularity-Opportunity Bias in Collaborative Filtering. In *WSDM*. 85–93.