Learning to Ask Appropriate Questions in Conversational Recommendation

Xuhui Ren
The University of Queensland
xuhui.ren@uq.net.au

Hongzhi Yin*
The University of Queensland
h.yin1@uq.edu.au

Tong Chen
The University of Queensland
tong.chen@uq.edu.au

Hao Wang
Alibaba Group
cashenry@126.com

Zi Huang
The University of Queensland
huang@itee.uq.edu.au

Kai Zheng
University of Electronic Science and Technology of China
zhengkai@uestc.edu.cn

ABSTRACT

Conversational recommender systems (CRSs) have revolutionized the conventional recommendation paradigm by embracing dialogue agents to dynamically capture the fine-grained user preference. In a typical conversational recommendation scenario, a CRS firstly generates questions to let the user clarify her/his demands and then makes suitable recommendations. Hence, the ability to generate suitable clarifying questions is the key to timely tracing users' dynamic preferences and achieving successful recommendations. However, existing CRSs fall short in asking high-quality questions because: (1) system-generated responses heavily depends on the performance of the dialogue policy agent, which has to be trained with huge conversation corpus to cover all circumstances; and (2) current CRSs cannot fully utilize the learned latent user profiles for generating appropriate and personalized responses.

To mitigate these issues, we propose the **K**nowledge-**B**ased **Q**uestion **G**eneration System (**KBQG**), a novel framework for conversational recommendation. Distinct from previous conversational recommender systems, KBQG models a user's preference in a finer granularity by identifying the most relevant **relations** from a structured knowledge graph (KG). Conditioned on the varied importance of different relations, the generated clarifying questions could perform better in impelling users to provide more details on their preferences. Finially, accurate recommendations can be generated in fewer conversational turns. Furthermore, the proposed KBQG outperforms all baselines in our experiments on two real-world datasets.

CCS CONCEPTS

• **Information systems → Users and interactive retrieval**; • **Computing methodologies → Knowledge representation and reasoning**.

*Corresponding author and having equal contribution with the first author.

SIGIR '21, July 11–15, 2021, Virtual Event, Canada

ACM ISBN 978-1-4503-8037-9/21/07.
https://doi.org/10.1145/3404835.3462839

KEYWORDS

Conversational recommender systems; knowledge graph; clarifying question; preference mining

ACM Reference Format:
Xuhui Ren, Hongzhi Yin*, Tong Chen, Hao Wang, Zi Huang, and Kai Zheng. 2021. Learning to Ask Appropriate Questions in Conversational Recommendation. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '21), July 11–15, 2021, Virtual Event, Canada.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3404835.3462839

1 INTRODUCTION

With personalized recommendation services, e-commerce platforms can easily infer users' preference and generate personalized recommendations based on their interactions with the platform (e.g., searching, reviewing, and purchasing) [18]. Despite the great success achieved, traditional recommender systems are inevitably constrained by its requirement on the passively collected user feedback. This brings information asymmetry between users and recommender systems, as the system can only recommend items based on a user's history instead of her/his real-time intent whenever the service is used [12].

Recently, the emerging language-based intelligent assistants such as Apple Siri and Google Home provide a new dimension in recommendation tasks. It enables an intelligent assistant to actively interact with users via conversations, so as to guide users to clarify their intent and find items that can meet their preferences [1]. This possibility is envisioned as a holistic composition of a dialogue agent and a recommender system, and is termed as Conversational Recommender System [12, 16]. In a general sense, CRS will proactively query a user with a series of clarifying questions to collect her/his demand on desired items. Then, the collected information serves as the preference context at the current timestamp, which helps generate timely and explainable recommendations.

Increasing attempts to develop CRSs for e-commerce have been put forward in recent years, and most of them can be categorized into two types: (1) policy-based approaches [1, 7, 19, 21, 36]; and (2) knowledge graph-based approaches [4, 13, 14, 38]. Policy-based approaches originated from task-oriented dialogue systems, where a policy agent is designed to automatically react during the conversational recommendation (e.g., making a recommendation or querying the user's preference on one specific attribute type). Such

Figure 1: An example of clarifying questions for a conversational recommendation.

methods can imitate the dialogue behaviors covered in the training corpus to generate human-like responses and complete the recommendation. However, their performance heavily relies on the labor- and knowledge-intensive process (i.e., acquisition and annotation) in preparing a comprehensive training dataset. Without enough well-annotated training corpus, most policy-based CRSs cannot learn an accurate dialogue action distribution for diversified dialogue state, thus underpeforming in dialogue generation. On the other hand, KG-based approaches have simplified the conversational recommendation into a "System Ask, User Answer" fashion with the help of knowledge graphs. By incorporating path-based reasoning into the conversation process, these methods can effectively offset the dependence on large training corpus. Each attribute node on the generated path will be used to formulate a yes/no question (e.g., "Do you like pop/rock/country music?") in order to affirm the user's current interest and explore the next node to connect. Finally, an explicit path in the KG will lead to a suitable item for recommendation. However, such exploration on KG only concerns the connectivity between nodes and overlooks the semantics behind users' responses. Apparently, they are less efficient as a node can be linked to multiple nodes, and the users may need to answer numerous questions before the correct item is reached.

Distinct from previous KG-based CRSs, we propose to fully account for the heterogeneous relations when reasoning over the paths in KG, so as to better understand the rationale behind a user's decision making process. As shown in Figure 1, the user recently has viewed several movies. Specifically, we define attribute types (e.g., genre) as relations that connect items with different attribute values (e.g., romantic) With the help of the KG, the main genres of his watched movie can be categorized into three types: *legendary*, *romantic* and *horror*. Our proposed KBQG could infer "genre" plays an important role in helping the user find the desired movie instead of other attribute types like "actor" and "director". Relations between interacted items are highly indicative on a user's intrinsic preferences, thus being beneficial for understanding fine-grained user interests. Therefore, our KBQG directly generates the clarifying questions to affirm the user's preference on the dominating attribute types for making an efficient and explainable recommendation. In light of this, we propose a novel framework named Knowledge-Based Question Generation system (KBQG) as a solution. Specifically, KBQG is designed to address the following key challenges in conversational recommendation tasks:

(1) *What personalized question to ask?* Different users may focus on different aspects when consuming items. When generating clarifying questions, a CRS should prioritize critical item attribute types that influence the user's decision the most. This can make a CRS locate users' demand and avoid irrelevant questions with substantially fewer conversation turns. In KBQG, we devise a KG-based recommender component to infer each user's taste on different item attributes (represented via relations in KG) from historical interaction records. A more important relation means the associated attribute is more crucial to the user, which should be clarified at an early conversation stage to help the recommendation.
(2) *Which system action should be performed?* Though the performance of the policy agent is directly associated with the recommendation quality, it is non-trivial to train an accurate policy agent given the potentially large action space. In this regard, we simplify our policy agent into a question-based paradigm, where the actions are either generating a recommendation or further asking a clarifying question. Before the recommendation module gains enough confidence, the CRS should continuously ask the user about her/his preference on different types of item attributes. Once the information is considered sufficient for an accurate recommendation, the CRS should execute the recommendation action.
(3) *How to react to users' responses?* The user will give their responses after each turn of conversation. Hence, it is crucial to make full use of the contexts within users' responses to further advance the recommendation performance. In KBQG, we characterize two kinds of user responses. First, if a user provides an explicit affirmation on a system-generated clarifying question, KBQG can effectively identify the user's demand and store it for future recommendations. Second, once the user rejects the recommendation generated by the model, KBQG is able to generate more refined questions in order to further adjust its recommendation.

To the best of our knowledge, KBQG's ability to generate personalized clarifying questions by investigating heterogeneous relations within KG is a brand new take on conversational recommendation. KBQG produces preference-aware clarifying questions, which help our system find users' desired items more efficiently. To validate the effectiveness of KBQG, we conduct extensive experiments on two real-world datasets, achieving state-of-the-art performance. Our contributions are summarized as follows:

- We propose a novel framework KBQG for KG-based conversational recommendation, which models the user preference as well as the diverse KG relations for generating personalized, preference-aware clarification questions.
- We reformulate the conversational recommendation scheme and simplify the dialogue action space, releasing the heavy burden for learning a sophisticated policy agent.
- By simulating human conversations, we build two KG-based conversational recommendation datasets that can be a stepstone for future research. Experimental analysis fully showcases the superiority of the proposed KBQG.

2 RELATED WORK

There are two lines of research inspired our proposed work: knowledge graph-based recommendation and conversational recommendation. We will discuss the research work in each relevant field.

2.1 Knowledge Graph-based Recommendation

The great success of recommender systems makes them prevalent in e-commerce platforms to offer recommendations that can meet users' preferences. To predict the user preference, many approaches have been proposed based on collaborative filtering (CF), which assume behaviorally similar users exhibit similar appetites on items [5, 8, 17, 26, 28, 29, 33]. However, CF-based methods usually perform poorly in modelling side information, such as user profiles and item attributes, thus cannot achieve satisfying performance when users and items have few interactions [25]. In real-world applications, the various side information, including reviews, relational data and knowledge graphs (KGs) is important to a model's interpretability, and is also closely associated with the efficiency and persuasiveness of recommender systems [3]. Incorporating side information into recommender systems to generate explainable recommendations has attracted much attention over the last few years, in which KG-based methods show significant performance advancements as KGs can provide rich facets about items.

Based on how graph-structured information is leveraged for recommendation, KG-based recommenders can be roughly categorized into three types, namely path-based [10, 22, 27, 31], propagation-based [23–25] and embedding-based [2, 3, 11, 15, 35] methods. Path-based methods mainly explore various path connectivity patterns among entities in the KG to provide additional guidance for recommendation. The main effort for path-based methods is to deal with the vast amount of paths between entities, which is either solved by devising path selection algorithms [22, 26] or by defining meta-path patterns [10]. Meanwhile, propagation-based methods concentrate on information propagation over the whole KG to augment the information for recommendation. It strengthens entities' semantic representations via the embeddings of their high-order neighbours in KG [23, 24]. The augmented entity embeddings serve as the exploitation of user interests or item properties, and are adopted for modelling user-item interactions [25]. Besides, embedding-based methods mainly project entities and relations of a KG into low-dimensional continuous vector space, in which the rich structured knowledge is preserved to enhance the expressiveness of user and item representations. Despite the great success in KG-based recommenders, few of them can explain which factor influences a user's decision. [32] models the user preference in a fine-grained manner and devises an attention-based neural network to analyze the influence of different relations on the user decision. However, similar to traditional recommenders, it assumes a user's interest is static over the long-time horizon [13], which hinders the recommender from generating accurate recommendations to satisfy users' dynamic preferences.

2.2 Conversational Recommendation

Conversational recommender systems are proved to be an effective solution for explainable and accurate recommendation to meet users' dynamic demand. This emerging research topic originates from task-oriented dialogue systems that help the user find her/his desired information with human-like dialogues [21]. Through multi-turn conversations with users, the dialogue system collects the user's intent and use a retrieval/content-based recommendation method to generate recommendations directly. The main challenge for this kind of method is to train an accurate policy agent to guide the dialogue action during the conversation. Several approaches have been proposed in the last three years for training a policy agent for conversational recommendation [1, 6, 7, 19, 21, 36]. However, the training process for these approaches usually suffers from the scarcity of training data. Also, a successful conversational recommendation requires a policy agent to cope with diverse dialogue actions, which brings even more difficulties to the training process.

Conditioned on the drawbacks of policy-based approaches, a new trend that employs KGs as an auxiliary information source for conversational recommendation has attracted much attention [4, 13, 14, 34, 38]. Early work on KG-based CRSs directly generates the recommendation based on connected neighbours of the user's mentioned items in the KG during the conversation [4, 14]. They can provide users with recommendations and persuasive reasons in a single conversation turn. [13] applies the KG-based method into a multi-round conversation scenario. It utilizes the structural information of KG to query the user with yes/no questions about her/his explicit preference on the specific attribute value (i.e. entity). Through a series of questions and answers between the user and the dialogue agent, the CRS finds a recommendation path in KG for the final recommendation. Even though the affirmative answer from the user could help CRSs find a promising path in KG for recommendation, the entities in a KG may have multiple relations with other nodes, hence the user may need to answer numerous questions before reaching the correct recommendation. This defect of exisiting KG-based recommenders would significantly weaken and recommendation efficiency and the user experience.

3 PRELIMIARIES

Following [13], we subsume our model under the multi-round conversational recommendation (MCR) paradigm but make some adjustments to improve its real-world applicability. MCR refers to one trial of recommendation as a round, and the CRS is free to interact with the user via attribute-related questions or recommendations multiple times until the task is finished. In this paper, we point out that asking generic questions can better impel the user to describe her/his specific demand compared with querying on explicit attributes. For example, asking "What is your preference on the genre of the movie?" is always more efficient and natural than blindly asking "Do you like romantic/horrible/art/...?" in capturing user preferences. So, we bypass the trial-and-error interaction style of the existing MCR paradigm, thus significantly shrinking the candidate question pool (i.e., action space) and making it more practical for e-commerce applications where a large amount of attributes are usually involved.

Specifically, an item is associated with a set of attribute types (e.g., color) and concrete attributes (e.g., blue) to describe its property. In a KG, an item is connected to an attribute value via an attribute type, e.g., $\textit{Jane Eyre} \xrightarrow{genre} \textit{romantic}$, where the attribute type (i.e., genre in this case) is regarded as a *relation* between these two entities. An item may have multiple descriptions under a same attribute type, e.g. the genre of book *Harry Potter* can be described

by *adventure*, *fantasy* and *fiction*. In a conversational recommendation round, KBQG aims to identify a user's most cared about attribute types (i.e., relations), then obtain her/his fine-grained preferences (i.e., attribute values) towards them by asking customized questions. The obtained preference information will eventually facilitate accurate recommendation.

4 METHODOLOGY

As shown in Figure 2, our proposed KBQG is executed with a three-phase process consisting of conversational policy decision, preference mining, and reaction decision. At the beginning, the process is started by a recommendation request from a user. Conditioned on the current dialogue context, a policy agent is responsible for deciding the response action a_t in the current conversation turn t, that is, whether to ask a clarifying question or to give a recommendation. The action a_t will be passed onto the preference mining phase. If the policy agent decides to ask a clarifying question, the model will infer which relation in the KG is more important to the user from her/his interaction records, and then leverage the identified relations to guide the generation of the question. If the policy agent decides to provide a recommendation, then the preference mining module acts as a recommender to determine the items to be recommended based on the current dialogue belief state $\mathbf{b}_t$, which is a vector storing the user's current interests expressed during the conversation. After the user gives feedback to the system-generated response, KBQG will decide how it should react to the user. If the user provides explicit expressions for a clarifying question, KBQG will keep extracting the user demand and updating the dialogue belief state for conversational policy decision. Once the user accepts the recommendation, or the conversation reaches the maximum conversation turn, the conversation will be terminated; otherwise, if the user rejects the recommendation while the maximum conversation turn is not reached, our KBQG should return to the question-asking phase to generate further clarifying questions to amend the recommendation results. In what follows, we will present the design of each phase in detail.

4.1 Conversational Policy Decision

The policy decision component decides which dialogue action should be conducted in the current conversation turn. As discussed earlier, existing KG-based CRSs commonly face the difficulty in training a sophisticated policy agent to cover diversified dialogue situations. The main reason is that, each dialogue action is defined as one yes/no question about a specific attribute value (e.g., "Do you like blue?"), which leads to an unnecessarily large action space for decision making. It requires to train an estimation module to estimate the information entropy of each attribute value, deducing which attribute value should be selected as the next dialogue action. However, such an estimation module usually cannot obtain satisfying performance when the dataset contains a large quantity of sparse attributes, impeding both the prediction accuracy and model generalizability. Furthermore, it is not as efficient and natural as directly enquiring about the user at the attribute type level (e.g., "Which color do you like?").

We address this problem by simplifying the dialogue actions at the first place. Specifically, KBQG has only two actions, which are either asking clarifying questions or generating recommendations for the user. The questions asked by KBQG is fully tailored based on a users' attention towards different attribute categories, which is fulfilled by the subsequent preference mining component so that the policy agent can focus on choosing between two actions at each conversation turn. Thus, we design a policy agent $\pi(\cdot)$ with multilayer perception, which is formulated as:

$$\pi_\theta(a_t|\mathbf{s}_t) = \sigma(\mathbf{W}_1 \cdot \tanh(\mathbf{W}_2\mathbf{s}_t + \mathbf{b}_1) + \mathbf{b}_2), \tag{1}$$

where σ is the sigmoid activation function, and $\theta = \{\mathbf{W}_1, \mathbf{W}_2, \mathbf{b}_1, \mathbf{b}_2\}$ denotes the weights and biases to learn in the policy agent. $\mathbf{s}_t$ is the internal dialogue state of the dialogue system that represents the current contexts of the dialogue at turn t, and it is defined as:

$$\mathbf{s}_t = \mathbf{b}_t \oplus \mathbf{q}_t \oplus \mathbf{c}_t, \tag{2}$$

where $\mathbf{b}_t$ is the dialogue belief state which is a latent encoding of the user utterance (c.f. Section 4.3), $\mathbf{q}_t$ is a multi-hot vector indicating which attribute type has been clarified by the user, and $\mathbf{c}_t$ is a vector indicating the ratio of candidate items whose affinity score with the user meets a predefined threshold M. The fewer candidate items means the recommender system has more confidence about the recommendation. The policy agent is trained in the RL for searching optimal conversational policy decisions. We adopt deep Q-learning as suggested in [13], to allow for easier and faster convergence. Correspondingly, a reinforcement reward will be given after each conversation turn according to the user feedback. Intuitively, the policy agent needs to keep querying the user when: (1) few user preferences are captured (i.e., $\mathbf{q}_t$ contains too many zeros); or (2) there are too many candidate items (i.e., $\mathbf{c}_t$ represents a large ratio) which need to be narrowed down to ensure accurate recommendation. Naturally, the policy agent will perform the recommendation action when sufficient preference signals are collected and the candidate item pool is small yet dedicated to the information provided. Recall that we define each attribute type as a relation in our KG, and we design the reward in KBQG as follows: (1) $r_{t,p}$, if the user provides an explicit demand on the queried relation; (2) $r_{t,n}$, if the user currently does not provide any detail on the queried relation; (3) $r_{t,a}$, if the user accepts the recommendation; and (4) $r_{t,m}$, if the user refuses the recommendation or the conversation reaches the maximum turn with no successful recommendation. The RL agent selects one of the RL rewards as the evaluation reward r_t to the current action. The policy agent is train towards maximizing the cumulated reward during the conversation. Finally, a policy gradient method is adopted to optimize the policy network via:

$$\nabla_\theta J(\theta) = \nabla_\theta \mathbb{E}[\log \pi_\theta(a_t|\mathbf{s}_t) R_t], \tag{3}$$

where $R_t = \sum_{t'=t}^{T_{max}} \eta^{t'} r_t$ is the cumulated reward from turn t to the end of conversation at T_{max}, and $\eta \in (0, 1)$ is a damping factor that encourages the model to accomplish the task in fewer conversation turns.

4.2 KG-based Preference Mining

The preference mining module in KBQG has two main functionalities. First, when guiding the generation of clarifying questions, it infers the most important relations for the user within the KG,

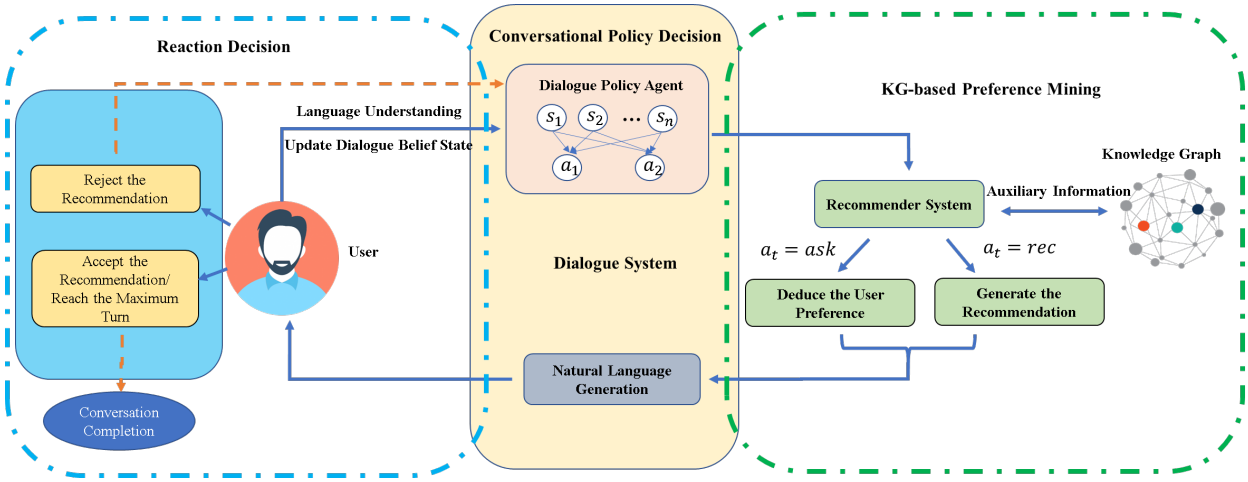

Figure 2: The overall diagram of KBQG. KBQG is executed with a three-phase process: Conversational Policy Decision, KG-based Preference Mining, and Reaction Decision.

which are used for composing the questions. Second, when performing the recommendation, it is responsible for generating personalized recommendations based on the user preference. Different users tend to focus on different attribute types when shopping online, e.g., some are more concerned on the price while some have preference on specific brands. Such user preferences can be captured from users' historical interactions. To better account for each relation's impact on the user's decision process, we propose a translation-based approach in KBQG for preference mining.

Conventional translation-based methods model the user preference by learning an additional translation vector $\mathbf{p}$ from a set of latent factors $\mathcal{P}$ for each user-item pair, such that given the representations of user $\mathbf{u}$ and item $\mathbf{i}$, we have $\mathbf{u}+\mathbf{p} \approx \mathbf{i}$ [2] if u prefers i. The translation vector $\mathbf{p}$ serves as a unique relation between users and items, and can be interpreted as the user's item-specific preference representation. The optimization objective is to minimize the $L1$ distance $||\mathbf{u}+\mathbf{p}-\mathbf{i}||$ for all positive user-item pairs. However, such method fails to consider the fine-grained user preference on attribute types when an interaction is made, hence are less suited for our goal of picking the most important attribute type for question generation.

Correspondingly, we design our preference mining module based on TransH [30], which assumes each relation has its own hyperplane, and the preference-driven translation from the user to the item is valid only if they are projected onto the same hyperplane. Hence, in KBQG, we propose the hyperplane-based preference translation function as follows:

$$f(u, i \mid p_u) = ||\mathbf{u}^{\perp} + \hat{\mathbf{p}}_u - \mathbf{i}^{\perp}||, \tag{4}$$

where $\mathbf{u}^{\perp}$ and $\mathbf{i}^{\perp}$ are projected user and item representations, and $\hat{\mathbf{p}}_u$ is the user's preference-driven translation vector. First, $\mathbf{u}^{\perp}$ and $\mathbf{i}^{\perp}$ are obtained via:

$$\begin{aligned} \hat{\mathbf{u}} &= \mathbf{u} + \mathbf{b}, \\ \mathbf{u}^{\perp} &= \hat{\mathbf{u}} - \mathbf{w}_u^{\mathrm{T}} \hat{\mathbf{u}} \mathbf{w}_u, \\ \mathbf{i}^{\perp} &= \mathbf{i} - \mathbf{w}_u^{\mathrm{T}} \mathbf{u} \mathbf{w}_u, \end{aligned} \tag{5}$$

where $\hat{\mathbf{u}}$ is the user embedding enhanced by the dialogue belief state $\mathbf{b}$, thus encoding the user's current preference during the conversation. Noticeably, to make this preference translation more suited to conversational recommendation, we further incorporate conversational contexts by augmenting the user embedding with dialogue belief state. $\mathbf{w}_u$ is the projection weight. Specifically, they are learned from the user's previously interacted items, where relations (i.e., attribute types) are thoroughly modelled to extend their semantic expressiveness. This also provides an explicit explanation for user's potential interactions with items. First, $\hat{\mathbf{p}}_u$ can be computed via:

$$\hat{\mathbf{p}}_u = \sum_{r \in \mathcal{R}} \alpha_{r,u} \mathbf{r}, \tag{6}$$

where $\mathbf{r}$ is the representation of each relation in the relation set $\mathcal{R}$, and $\alpha_{r,u}$ is user u's attention weight on each specific relation. We define $\alpha_{r,u}$ with the standard softmax function:

$$\alpha_{r,u} = \frac{\exp(a(r,u))}{\sum_{r' \in R} \exp(a(r',u))}, \tag{7}$$

where $a(r,u)$ is the attention score of user u towards relation r. We learn it with an attention network:

$$a(r,u) = \mathbf{h}^{\mathrm{T}}(ReLU(\mathbf{W}_3(\mathbf{u} \oplus \mathbf{r}) + \mathbf{b}_3)) \tag{8}$$

$\mathbf{W}_3$ and $\mathbf{b}_3$ are weight matrix and bias vector, and $\mathbf{h}$ projects the calculated latent representation into a scalar score. A higher attention score means user u exhibits higher interest on r when searching for the next item to interact with. All relations are ranked in a descending order with their corresponding attention weights. KBQG sequentially pick an unused attribute type in each conversation turn for generating clarifying questions. As such, instead of learning a latent preference to connect items and users as in [3], we incorporate the rich semantics from the relations in KG to infer the user's most concerned attribute types, making it possible to generate appropriate questions for conversational recommendation.

Similarly, the projection vector $\mathbf{w}_u$ can be obtained with the projection vector $\mathbf{w}_r$ on each relation hyperplane:

$$\mathbf{w}_u = \sum_{r \in R} \beta_{r,u} \mathbf{w}_r, \tag{9}$$

where another set of attention weights $\beta_{r,u}$ are computed, and $\mathbf{w}_r$ denotes the relation-specific projection weight. By replacing $\mathbf{r}$ in Eq.6 with $\mathbf{w}_r$ in a parallel attention network, we can calculate $\beta_{r,u}$ and $\mathbf{w}_u$ analogously.

Thus, the auxiliary information from KG and dialogue content could enhance the semantic representation and learn the connections between users and items. For each user u, we define Bayesian Personalized Ranking [20] loss to encourage u's translation distance between all interacted items $i \in \mathcal{I}_u^+$ to be smaller than randomly selected negative items $i' \in \mathcal{I}_u^-$:

$$\mathcal{L}_f = \sum_{(u,i) \in \mathcal{H}^+} \sum_{(u,i') \in \mathcal{H}^-} -\ln(\sigma(f(u,i' \mid p_u) - f(u,i \mid p_u))), \tag{10}$$

where $\mathcal{H}^+ = \{\mathcal{I}_u^+\}_{\forall u}$ and $\mathcal{H}^- = \{\mathcal{I}_u^-\}_{\forall u}$ are all positive and negative interaction instances used for training. In order to effectively preserve the relations between entities within the KG, we jointly optimize the recommendation loss with a relation modelling loss in KG. According to the assumption of TransH, the translation distance between the head entity e_h and the tail entity e_t is defined as:

$$f(e_h, e_t, r) = ||\mathbf{e}_h^\perp + \mathbf{r} - \mathbf{e}_t^\perp|| \tag{11}$$

where the projected embeddings of the head and tail entities can be obtained via:

$$\begin{aligned} \mathbf{e}_h^\perp &= \mathbf{e}_h - \mathbf{w}_r^\mathrm{T} \mathbf{e}_h \mathbf{w}_r \\ \mathbf{e}_t^\perp &= \mathbf{e}_t - \mathbf{w}_r^\mathrm{T} \mathbf{e}_t \mathbf{w}_r \end{aligned} \tag{12}$$

where $\mathbf{w}_r$ is the relation-specific projection vector as in Eq.9. The training objective of TransH is to maximize the margin between the observed triples in $\mathcal{KG}^+$ and randomly sampled false triples in $\mathcal{KG}^-$, which is defined as the following:

$$\mathcal{L}_g = \sum_{(e_h,e_t,r) \in \mathcal{KG}} \sum_{(e'_h,e'_t,r') \in \mathcal{KG}^-} [f(e_h,e_t,r) - f(e'_h,e'_t,r')]. \tag{13}$$

The final training objective function can be written as:

$$\mathcal{L} = \lambda \mathcal{L}_f + (1-\lambda)\mathcal{L}_g \tag{14}$$

where λ is used to trade-off the influence of two parts.

4.3 Reaction Decision

After the user gives corresponding feedback on the system-generated action (i.e. clarifying question or recommendation), KBQG is supposed to react accordingly based on the information carried by the user feedback. When KBQG has not collected enough information about the user interest, the user will respond to the generated questions to clarify her/his demand on specific attribute types. In this condition, we introduce an entity linking algorithm proposed by [9] to accurately identify the descriptive phrases that appeared in the users' replies. These identified entities (i.e., detailed attribute values) $z_1, z_2, \ldots, z_t$ are recognized as the user's current explicit interest on the attribute types asked, whose embeddings are used for composing the dialogue belief state $\mathbf{b}_t$:

$$\mathbf{b}_t = \sum_{t'=1}^{t} \mathbf{z}_{t'}, \tag{15}$$

When the policy agent decides to make recommendation, KBQG will search in the database with recommendation score $f(u, i \mid p_u)$ and output top-K items (i.e., K items with lowest $f(u,i,p_u)$ scores) for recommendation. The user can respond to the recommendation list with acceptance or rejection. If the user accepts the recommendation, this conversation is considered as a success; otherwise, our system needs to further collect user interest and narrow down the recommendation candidates until the maximum of the conversation turn T. Meanwhile, the failed recommendations are stored in a negative set $\mathcal{S}_u^-$ and are avoided in the next round of recommendation. That is, the item candidate set for u is $\mathcal{S}_u^{cand} = \mathcal{S}_u^+ \backslash \mathcal{S}_u^-$, where $\mathcal{S}_u^+$ is the set of candidate items whose predicted distance to u is smaller than the threshold M described in Section 4.1. Finally, if the system does not find the user's desired item until the maximum conversation turn, we regard it as a failed recommendation attempt.

4.4 Training

The whole training process includes two parts: (1) offline training for preference mining; and (2) online training for conversational policy decision. The training objective for the first part is to minimize the distance between the embeddings of a user and her/his preferred items. A lower score of $f(u, i \mid p)$ indicates higher interest of the user to the item. For each user, we regard all entities connected to her/his interacted items as the user's affirmed preference, and store them in the b_t to train the recommender. A multi-task training method is adopted to optimize both the recommendation and KG modelling losses as mentioned in Section 4.2.

For online training, we introduce a user simulator to interact with KBQG to train the policy agent. The user simulator is a simplified version of the one used in [13]. In our case, the user simulator provides the user-side information based on the user's historical interactions. For each conversation session, the user's current preference is predefined by the attribute types and values associated with her/his interacted items in KG. The conversation between KBQG and the user simulator includes two parts. Firstly, KBQG sequentially asks clarifying questions during the conversation. If the queried relations match the user's preference on attribute types, the user simulator will respond with detailed attribute values. Secondly, when KBQG provides the recommendation list to the user, the user simulator will accept the results if the ground-truth item i is present. Otherwise, the recommendation is rejected.

5 EXPERIMENTS

We evaluate KBQG on two real-world datasets. Specifically, we aim to answer the following research questions (RQs):

RQ1: How does our proposed KBQG compare with state-of-the-art conversational recommendation methods?

RQ2: Are our user preference inference scheme and simplified policy agent really effective?

Table 1: Statistics of MovieLens-1M and DBbook2014

		MovieLens	DBbook2014
Interactions	Users	6,040	5,576
	#Items	3,240	2,680
	#Ratings	998,539	65,961
	#Avg. Interactions	165	12
KG	#Entity	9,457	8,793
	#Relation	13	9
	#Triple	159,492	194,710

RQ3: Can KBQG help comprehend user preferences and provide convincing yet explainable recommendation with conversations?

5.1 Datasets

We conduct experiments on two publicly available datasets from two different domains, namely MovieLens-1M for movie recommendation[1] and DBbook2014 for book recommendation[2]. For both datasets, we follow [3] to transform users' explicit ratings into implicit feedback where ratings meeting a predefined threshold are considered as positive interactions. The rating threshold is 4 for MovieLens-1M and is 1 for DBbook2014 due to data sparsity. For each user, we also sample an equal amount of negative items to match the positive interactions.

To build KGs for both datasets, we map items to DBPedia entities (if available), and filter out relations that are infeasible for generating clarifying questions, e.g., non-textual data like URLs. We preprocess both datasets by filtering out infrequent users and items (i.e. retaining users and items with at least 10 interactions). Table 1 shows the statistics of two datasets after processing. There are 6,040 users, 3,240 items and 998,539 interaction records in MovieLens-1M dataset. On average, each user has 165 interaction records. DBbook2014 has 5,576 users, 2,680 items and 65,961 interaction records, leading to an average of 12 interactions per user. The KG triples used in both datasets are at the same scale, where the subgraph for MovieLens-1M composes of 159,492 triples with 87,492 entities and 13 relations, while the subgraph for DBbook has 194,710 triples with 8,793 entities and 9 relations.

5.2 Experimental Settings

5.2.1 Training Details. We randomly split each dataset for training, validation and test with the ratio of $7:2:1$. When making recommendations, we set $K = 10$ for generating the top-K item list. For each dataset, the maximum conversation turn is set to fit the total number of KG relations plus one for recommendation, i.e., $T = 14$ for MovieLens-1M and $T = 10$ for DBbook2014, respectively. The rationale is that, if a recommender has already obtained users' preferences on all relations, it should start making recommendations as no further information can be acquired from the user. We respectively set the hyperparameters for offline training and online training phases. For offline training, the embedding size is 100, and the batch size is 256. The optimizers for the recommendation module and KG modelling are Adagrad and Adam with the learning rate of learning rate of 0.003 and 0.001, respectively. For recommendation, we also impose an L_2 regularization weighted by 10^{-4}. We set the trade-off coefficient λ to 0.5 for MovieLens-1M and 0.7 for DBbook2014 following [3]. For online training, the policy agent is trained with deep Q-learning using RMSprop. The specific values of RL rewards are designed as: $\{r_{t,p} = 0.1, r_{t,n} = -0.1, r_{t,a} = 1, r_{t,m} = -0.3\}$. The deep Q-learning has the batch size of 128, the experience memory size of 100,000, and the damping factor η of 0.9. After performing grid search in $\{0.25, 0.5, 0.75\}$, we set the recommendation threshold M in the dialogue belief state to 0.25 and 0.5 on MovieLens-1M and DBbook2014, respectively.

Considering generating extremely human-like responses is not the main research task in this paper, we currently design a series of response templates for constructing different clarifying questions. The selected relations are filled in the vacant position of the templates to conduct the conversation. We leave the exploration of language generation in the future work. [3]

5.2.2 Baselines. Studies on CRS are emerging in recent years, which focus on different application scenarios. To verify the performance of our proposed KBQG, we select the state-of-the-art methods that have similar task settings as ours. The baselines are summarized as follows:

- **CRM:** This is a policy-based CRS that integrates a task-oriented dialogue system with a recommender [21]. The main idea is to involve a semi-structured user query to represent the dialogue history as well as the user's current preference. This query is then used for generating personalized recommendations.
- **EAR:** This method is designed in MCR scenario and is equipped with an Estimation-Action-Reflection framework for CRS [12]. It estimates the importance of each attribute with information entropy and utilizes a policy agent to select a suitable attribute from in each conversation turn to formulate a yes/no question, acquiring the user preference information for recommendation.
- **SCPR:** This method proposes to incorporate path searching on KG into CRS in MCR scenario [13]. It formulates a series of yes/no questions to collect the user's feedback and use these feedback as the guidance search on KG to find the recommendation.

Although there are other recent CRSs [4, 14, 16, 19, 37], they are inapplicable for comparison due to different task settings. For example, [14, 37] is designed to recommend a similar recommendation to a specific prototype without clarifying the user's real-time preference, and [16, 19] are more concerned on language understanding and generation.

5.2.3 Evaluation Metrics. We follow [13] to evaluate the performance of each CRS with success rate at turn T (SR@T) and average turn (AT) of conversations. SR@T is the accumulative task success rate by turn T, and we set $T = T_{max}$ by default for overall performance comparison. AT is the average number of conversational turns needed for a successful recommendation session. If a user

[1] https://grouplens.org/datasets/movielens/
[2] http://2014.eswc-conferences.org/important-dates/call-RecSys.html
[3] The source code will be release at https://github.com/XuhuiRen/KBQG.

Table 2: Performance comparison of all methods on two datasets by SR and AT, where the best performance is bold-faced ($p < 0.01$ in one-sample paired t-tests when comparing all our results with baselines' scores).

	DBbook2014		MovieLens-1M	
	SR	AT	SR	AT
CRM	0.286	8.593	0.761	8.117
EAR	0.247	9.369	0.594	10.362
SCPR	0.275	9.255	0.659	10.099
KBQG	**0.323**	**8.156**	**0.816**	**7.335**

cannot find his desired item till T_{max}, we return T_{max} for computing SR. A higher score on SR represents better recommendation accuracy, whereas a lower AT score indicates a model could more efficiently deliver a successful conversational recommendation using fewer conversations.

5.3 Performance Comparison (RQ1)

Table 2 shows the overall performance of our proposed system as well as the baseline systems. It is obvious that our proposed system achieves the best performance among all methods on both datasets regarding SR and AT.

Compared with KG-based methods EAR and SCPR, CRM and our proposed KBQG exhibit significant advantages on both DBbook2014 and MovieLens-1M. The main reason behind this phenomenon is the items on both KGs for two datasets are linked with a lot of attributes. However, the expansion of the item attributes requires a huge amount of training set to train an estimation module that can estimate the information entropy on the whole attribute when generating the clarifying questions, which is a key component in EAR and SCPR. Without an accurate entropy estimation on the potential attributes, their system cannot find the suitable attributes to ask for the user preference, leading to a terrible performance on task completion and conversion efficiency. What's more, the clarifying mode for EAR and SCPR collects the user preference with a series of yes/no questions on the specific attributes. It inherently cannot achieve a satisfying efficiency when there is a large attribute pool to describe the items. Our proposed KBQG and CRM clarify the user preference with a set of generic questions. The user can describe his preference on each attribute type with the guidance of clarifying questions, achieving a better conversational recommendation experience.

Compared with policy-based method CRM, our proposed KBQG still presents remarkable improvement. The reason is that the policy agent of CRM mainly relies on the dialogue action distribution learned from the training corpus to generate the next clarifying question. It lacks the ability to analyze the importance weight of different relations on the user's decision. Therefore, it usually requires more user preference and more conversation turn for the recommendation; and the ultimate process of CRM is decomposed to a traditional recommender system Factorization Machine [21] to recommend the items. Hence, it fails to utilize the connection information on KG to enhance the entity embedding.

Figure 3 presents a finer-grained performance comparison of the task cumulative success rate on two datasets. From the figure, we can observe that our proposed KBQG could reach a higher

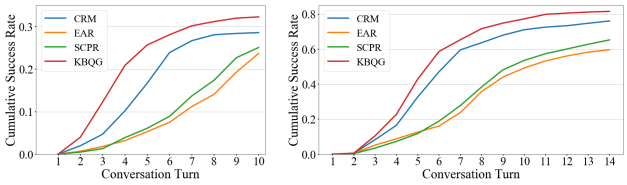

Figure 3: Cumulative success rate of compared methods at different conversation turns on DBbook2014 (left) and MovieLens-1M (right).

task success rate at the early stage of the conversation. The reason is that our proposed KBQG could inference which relation is more important for the user to make a decision, and give priority to the user preference on the relations that occupy a high attention weight for generating the recommendation. It can filter many unimportant relations during the clarifying period to accelerate the user preference collecting process. Also, the recommendation performance in our system is benefit from the relational modelling with TransH, and achieves better performance compared with a traditional recommender system.

5.4 Analysis on Key Designs (RQ2)

The key design in our proposed KBQG is the simplification and optimization of the conventional policy-based CRS. The policy agent in our proposed KBQG only accounts for deducing a dialogue action, asking a clarifying question or presenting the recommendation, which means the dialogue action space is greatly decreased. Hence, it can achieve superior performance on the dialogue action generation. KBQG could model the user decision policy from the rich resource of historical interactions, providing an effective way to infer which relation is more appropriate for formulating the next clarifying question. To verify its effectiveness, we conduct ablation experiments by designing two variants of our KBQG, marked with **KBQG-P** and **KBQG-A**. KBQG-P replaces the current policy agent with a standard policy agent as in [21] that has $R + 1$ dialogue actions. It decides the next dialogue action based the distribution of dialogue actions in the training corpus. The policy agent behaves similarly compared with other policy-based methods in each conversation turn when selecting the next relation and deciding asking/recommending. All other components in KBQG-P are inherited from KBQG. KBQG-A maintains the policy agent in KBQG, but replaces the attentive relation aggregation with average pooling, hence all relations share the same possibility of being selected to formulate the clarifying question.

As can be observed from Figure 5, our proposed KBQG still keeps the best performance over these two variants. The performance drop on each variant proves the effectiveness of our proposed method. KBQG-P shows inferior performance at the early stage of the conversation. This is because KBQG-P needs to train a policy agent to simulate the dialogue distribution, which is easily influenced by the quality of the training corpus, especially when the training corpus is synthesized by a simulated user. Therefore, it cannot distinguish which relation is more important to the user decision, and can only deduce the next dialogue action conditioned on the current dialogue belief state. KBQG-A randomly selects relations to formulate clarifying questions during the conversation, as each relation share the same attention weight for influencing

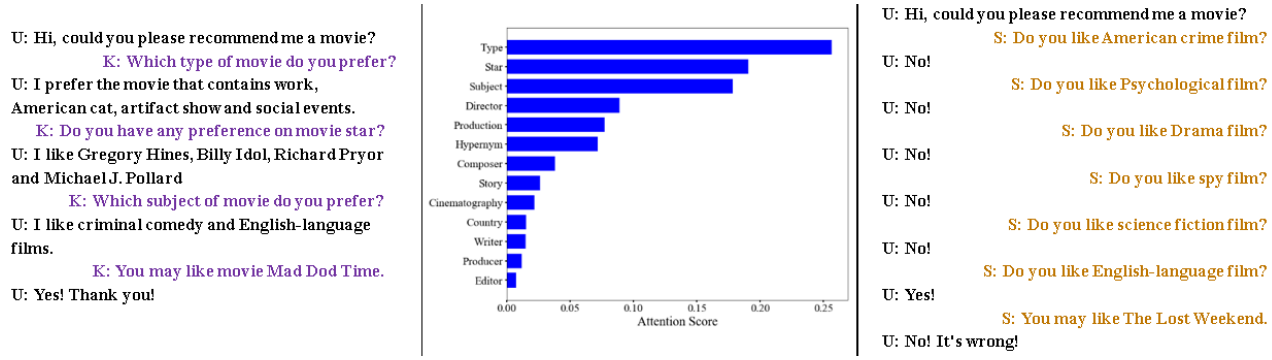

Figure 4: Sample conversations generated by KBQG (left) and SCPR (right) and attention scores of different relations (middle).

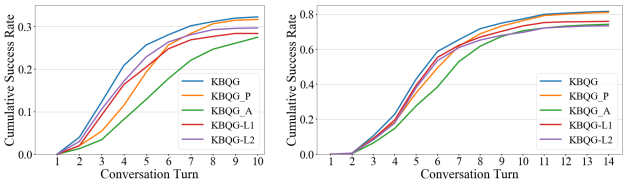

Figure 5: Cumulative success rate of compared methods at different conversation turns on DBbook2014 (left) and MovieLens-1M (right).

the user decision. This schema may let the system ask clarifying questions about the unimportant relations, which would cripple the model efficiency.

In addition, we further analyze the influence of joint learning parameter λ in Section 4.2 to the overall model performance. By default, we set λ to 0.5 on MovieLens-1M and 0.7 on DBbook2014. We alternate the value of λ of KBQG, where **KBQG-L1** uses $\lambda = 0.3$ on both datasets, and **KBQG-L2** uses $\lambda = 0.7$ on MovieLense, and $\lambda = 0.5$ on DBbook2014. As we can tell from Figure 5, KBQG-L1 and KBQG-L2 perform worse than our default setting. KBQG-L1 performs worse on DBbook2014 but obtains better performance on MovieLens-1M compared with KBQG-L2. The main reason is that MovieLens-1M contains more interaction records for the recommender to capture implicit user preferences. Putting a higher weight on the relation modelling could enhance the entity embeddings to improve the recommendation performance. Whereas DBbook2014 exhibits fewer historical records for modelling the user preference, it is more suitable to lay more emphasis on the recommendation loss, which would benefit the overall recommendation performance. Both the KG relation modelling and recommendation parts simultaneously take effects on the eventual recommendation quality, and λ should be adjusted according to the property of each dataset for optimal recommendation performance.

5.5 Qualitative Analysis (RQ3)

We further conduct qualitative analysis to show how KBQG is able to improve its understanding of the user preferences and generate appropriate questions for explainable recommendations. We randomly select one real conversation record from KBQG and SCPR on MovieLens based on the same test instance. SCPR is a representative of graph-based approaches that designs its action space based on all attribute values. The conversation is initiated by the user (ID 4607) requesting for a movie recommendation.

Figure 4 demonstrates the conversations between the user and KBQG/SCPR. It also visualizes the user's attention weights on different relations learned by KBQG. Clearly, KBQG asks clarifying questions following the rank of attention weights on different relations to collect user preference information. From the conversations, we can find KG-based method that relies on the user's feedback to find the recommendation path is substantially less efficient when the dataset has a large attribute pool. Hence, SCPR requires a longer conversation with the user to find a suitable recommendation. In contrast, our proposed KBQG is designed to generate the conversation in a totally different fashion. It substantially uses the recommender system to rank the attention weight of different relations, formulating the most appropriate clarifying questions for obtaining a user's preferences. It is well-suited to the large KG dataset. With the help of attention scores learned on different relations, KBQG is able to filter unnecessary clarifying questions to improve the conversation efficiency.

6 CONCLUSION

In this paper, we redefine the conversational recommender system and propose a novel KG-based conversational recommender system, KBQG, where the recommender system and the dialogue system closely cooperate with each other so as to efficiently and accurately generate recommendations in a short conversation. Specifically, the preference mining module in KBQG mainly extracts rich auxiliary information from the KG to explicitly explore users preferences from historical records. Conditioned on the explored preference, KBQG can effectively tailor the clarifying questions by prioritizing attribute types that are important to the user. After multiple conversation turns, personalized recommendations will be given when the user has sufficiently clarified her/his real-time interests. With extensive experiments on two real-world datasets, KBQG presents superior performance compared with state-of-the-art policy-based and knowledge-based CRSs.

ACKNOWLEDGMENTS

This work was supported by ARC Discovery Project (Grant No. DP190101985) and ARC Training Centre for Information Resilience (Grant No. IC200100022).

REFERENCES

[1] Keping Bi, Qingyao Ai, Yongfeng Zhang, and W. Bruce Croft. 2019. Conversational Product Search Based on Negative Feedback. In *CIKM'19*. 359–368.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NIPS*, Vol. 26. 2787–2795.

[3] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preferences. In *WWW*. 151–161.

[4] Qibin Chen, Junyang Lin, Yichang Zhang, Ming Ding, Yukuo Cen, Hongxia Yang, and Jie Tang. 2019. Towards Knowledge-Based Recommender Dialog System. In *EMNLP-IJCNLP*. 1803–1813.

[5] Tong Chen, Hongzhi Yin, Hongxu Chen, Rui Yan, Quoc Viet Hung Nguyen, and Xue Li. 2019. AIR: Attentional Intention-Aware Recommender Systems. In *ICDE*. 304–315.

[6] Tong Chen, Hongzhi Yin, Guanhua Ye, Zi Huang, Yang Wang, and Meng Wang. 2020. Try This Instead: Personalized and Interpretable Substitute Recommendation *(SIGIR '20)*. 891–900.

[7] Konstantina Christakopoulou, Filip Radlinski, and Katja Hofmann. 2016. Towards Conversational Recommender Systems *(KDD '16)*. 815–824.

[8] Hui Cui, Lei Zhu, Jingjing Li, Yang Yang, and Liqiang Nie. 2019. Scalable deep hashing for large-scale social image retrieval. *IEEE Transactions on image processing* 29 (2019), 1271–1284.

[9] Paolo Ferragina and Ugo Scaiella. 2010. TAGME: On-the-Fly Annotation of Short Text Fragments (by Wikipedia Entities) *(CIKM '10)*. 1625–1628.

[10] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S. Yu. 2018. Leveraging Meta-Path Based Context for Top- N Recommendation with A Neural Co-Attention Model *(KDD '18)*. 1531–1540.

[11] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. 687–696.

[12] 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 *(WSDM '20)*. 304–312.

[13] 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 *KDD'2020*.

[14] Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards Deep Conversational Recommendations *(NIPS '18)*. 9725–9735.

[15] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion *(AAAI'15)*. 2181–2187.

[16] Zeming Liu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, and Ting Liu. 2020. Towards Conversational Recommendation over Multi-Type Dialogs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 1036–1049.

[17] Xu Lu, Lei Zhu, Zhiyong Cheng, Liqiang Nie, and Huaxiang Zhang. 2019. Online Multi-modal Hashing with Dynamic Query-adaption. In *SIGIR*. 715–724.

[18] Gustavo Padilha Polleti, Hugo Neri Munhoz, and Fabio Gagliardi Cozman. 2020. Explanations within Conversational Recommendation Systems: Improving Coverage through Knowledge Graph Embeddings. In *AAAI'2020*.

[19] Xuhui Ren, Hongzhi Yin, Tong Chen, Hao Wang, Nguyen Quoc Viet Hung, Zi Huang, and Xiangliang Zhang. 2020. CRSAL: Conversational Recommender Systems with Adversarial Learning. *ACM Trans. Inf. Syst.* 38, 4, Article 34 (June 2020), 40 pages.

[20] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. *UAI* (2009).

[21] Yueming Sun and Yi Zhang. 2018. Conversational Recommender System. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18)*. 235–244.

[22] Zhu Sun, Jie Yang, Jie Zhang, Alessandro Bozzon, Long-Kai Huang, and Chi Xu. 2018. Recurrent Knowledge Graph Embedding for Effective Recommendation *(RecSys '18)*. 297–305.

[23] Xiaoli Tang, Tengyun Wang, Haizhi Yang, and Hengjie Song. 2019. AKUPM: Attention-Enhanced Knowledge-Aware User Preference Model for Recommendation *(KDD '19)*. 1891–1899.

[24] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. RippleNet: Propagating User Preferences on the Knowledge Graph for Recommender Systems *(CIKM '18)*. 417–426.

[25] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation *(KDD '19)*. 950–958.

[26] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering *(SIGIR'19)*. 165–174.

[27] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable Reasoning over Knowledge Graph Paths for Recommendation *(AAAI'19)*. 5329–5336.

[28] Yuandong Wang, Hongzhi Yin, Hongxu Chen, Tianyu Wo, Jie Xu, and Kai Zheng. 2019. Origin-Destination Matrix Prediction via Graph Convolution: A New Perspective of Passenger Demand Modeling *(KDD '19)*. 1227–1235.

[29] Ze Wang, Guangyan Lin, Huobin Tan, Qinghong Chen, and Xiyang Liu. 2020. CKAN: Collaborative Knowledge-Aware Attentive Network for Recommender Systems *(SIGIR '20)*. 219–228.

[30] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes *(AAAI'14)*. 1112–1119.

[31] Yikun Xian, Zuohui Fu, S. Muthukrishnan, Gerard de Melo, and Yongfeng Zhang. 2019. Reinforcement Knowledge Graph Reasoning for Explainable Recommendation *(SIGIR'19)*. 285–294.

[32] Xin Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon Jose. 2019. Relational Collaborative Filtering: Modeling Multiple Item Relations for Recommendation *(SIGIR'19)*. 125–134.

[33] Hongzhi Yin and Bin Cui. 2016. *Spatio-temporal recommendation in social media*. Springer.

[34] Chen Zhang, Hao Wang, Feijun Jian, and Hongzhi Yin. 2021. Adapting to Context-Aware Knowledge in Natural Conversation for Multi-Turn Response Selection *(WWW)*.

[35] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems *(KDD '16)*. 353–362.

[36] Yongfeng Zhang, Xu Chen, Qingyao Ai, Liu Yang, and W. Bruce Croft. 2018. Towards Conversational Search and Recommendation: System Ask, User Respond *(CIKM '18)*. 177–186.

[37] 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)*. 1006–1014.

[38] Jie Zou, Yifan Chen, and Evangelos Kanoulas. 2020. Towards Question-Based Recommender Systems *(SIGIR '20)*. 881–890.