Cognition-aware Knowledge Graph Reasoning for Explainable Recommendation

Qingyu Bing*
School of Information, Renmin University of China, Beijing, China
bqy@ruc.edu.cn

Qiannan Zhu*
Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
zhuqiannan@ruc.edu.cn

Zhicheng Dou†
Gaoling School of Artificial Intelligence, Renmin University of China, Beijing, China
dou@ruc.edu.cn

ABSTRACT

Knowledge graphs (KGs) have been widely used in recommendation systems to improve recommendation accuracy and interpretability effectively. Recent research usually endows KG reasoning to find the multi-hop user-item connection paths for explaining why an item is recommended. The existing path-finding process is well designed by logic-driven inference algorithms, while there exists a gap between how algorithms and users perceive the reasoning process. Factually, human thinking is a natural reasoning process that can provide more proper and convincing explanations of why particular decisions are made. Motivated by the Dual Process Theory in cognitive science, we propose a cognition-aware KG reasoning model CogER for Explainable Recommendation, which imitates the human cognition process and designs two modules, i.e., System 1 (making intuitive judgment) and System 2 (conducting explicit reasoning), to generate the actual decision-making process. At each step during the cognition-aware reasoning process, System 1 generates an intuitive estimation of the next-step entity based on the user's historical behavior, and System 2 conducts explicit reasoning and selects the most promising knowledge entities. These two modules work iteratively and are mutually complementary, enabling our model to yield high-quality recommendations and proper reasoning paths. Experiments on three real-world datasets show that our model achieves better recommendation results with explanations compared with previous methods.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → **Knowledge representation and reasoning**.

KEYWORDS

Recommendation Systems, Explainable Recommendation, Knowledge Graph Reasoning, Reinforcement Learning

*Both authors contributed equally to this research.
†Corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
WSDM '23, February 27-March 3, 2023, Singapore, Singapore
© 2023 Association for Computing Machinery.
ACM ISBN 978-1-4503-9407-9/23/02…$15.00
https://doi.org/10.1145/3539597.3570391

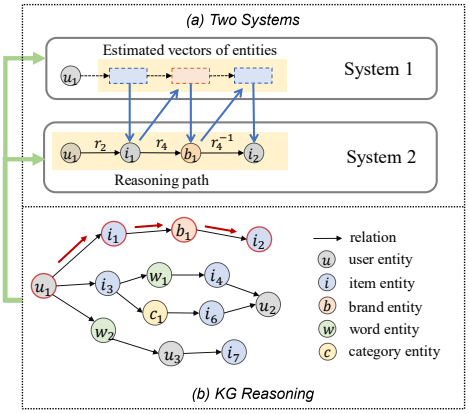


Figure 1: Illustration of the cognition-aware KG reasoning process. (a) System 1 and System 2 work iteratively. The blue arrows indicate the interactions between two systems. (b) The red circles and arrows denote a reasoning path on the KG when the recommendation is made for user u_1.

ACM Reference Format:
Qingyu Bing, Qiannan Zhu, and Zhicheng Dou. 2023. Cognition-aware Knowledge Graph Reasoning for Explainable Recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23), February 27-March 3, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3539597.3570391

1 INTRODUCTION

Recommendation systems attract increasing attention in both industry and academia [2, 7] due to their ability to generate personalized recommendations for different users. In the field of E-commerce, recommendation systems serve to predict products that users may be interested in based on their historical behaviours, such as purchases, reviews, and ratings. In the past decades, lots of excellent work has been carried out to improve the accuracy and quality of recommendations [27, 32, 43]. Traditional methods like Collaborative Filtering-based recommendation [23, 28] and content-based recommendation [26], usually leverage various information (e.g., reviews, implicit or explicit feedback) to capture users' preferences. In recent years, deep neural networks have been applied to recommendation for better representations of users and items, and consequently improving recommendation accuracy [5, 13]. Besides, recommendation with explanations (i.e. Explainable Recommendation) is drawing more and more attention from researchers [6, 36, 44]. With explanations, users can understand why the recommendation

systems recommend certain items for them, which consequently brings a better user experience.

To deal with this problem of explainable recommendation, Knowledge Graphs (KGs), which contain comprehensive structural knowledge, have been applied. These knowledge-aware explainable models usually find multi-hop paths over KGs as explanations for the recommended items [37, 42]. To take an example in Figure 1*(b)*, in the knowledge graph, the user u_1 has bought item i_1 whose brand is b_1, and there's another item i_2 under the same brand b_1. Based on their connections in the KG, the item i_2 could be recommended to user u_1 with the explanation "the user may purchase this item under the same brand as a previously purchased item". The process above is widely called **KG reasoning** [34, 38]. The reasoning path $u_1 \rightarrow i_1 \rightarrow b_1 \rightarrow i_2$ consists of each step of the reasoning process on the KG and thus serves as the explanation for recommending item i_2. The KG reasoning methods for recommendation not only obtain high-quality recommendation results but also generate corresponding explanations. These methods can be classified into embedding-based methods which adopt KG embeddings for recommendation [33, 40, 43], and path-based methods which incorporate structure information of the KG into the path-finding process [17, 37, 42]. However, the methods above are all logic-driven algorithms and have differences from actual human reasoning. In this paper, **we get inspiration from cognitive science, and propose a model for explainable recommendation that imitates the reasoning process of human beings**.

Described by Dual Process Theory [29] in cognitive science, the reasoning (thinking) process of human beings involves two processes: an intuitive, unconscious and fast process called System 1 and a logical, conscious and slow process called System 2. As specified in the study of cognitive science [19], System 1 is employed to get an intuitive judgment of the current situation and reach an experience-based conclusion. System 2 is used for deeper and more logical reasoning to obtain more reasonable results. When humans make decisions or perform reasoning tasks, System 1 is unconsciously deployed and makes an intuitive judgment. When complex reasoning is needed, System 2 is deployed and works iteratively together with System 1 [9], as shown in Figure 1*(a)*.

Intuitively, the reasoning process, which naturally satisfies the human thinking process, can provide more convincing explanations of why particular decisions are made. In this paper, we draw inspiration from Dual Process Theory and propose a **cognition-aware KG reasoning method, namely CogER, for explainable recommendation**, which is composed of System 1 (intuitive judgment) and System 2 (explicit reasoning). In our model, **System 1** is a simple entity estimator that generates the estimated representation (intuitive judgment) of the next-step entity based on users' preferences (situation), which offers guidance on the following reasoning process in System 2. **System 2** is an advanced reinforcement learning (RL) based framework that carefully selects the most important entities in the reasoning path under the guidance of the estimated embedding from System 1. The selected entity is then fed to System 1 as the clue to generate the estimated embedding of the next-step entity. As illustrated in Figure 1, **the two systems interact with each other** during the reasoning process on the KG, until the final recommendation result i_2 is obtained. In our model, knowledge graphs serve as the information source of both System 1 and System 2, which is like the acquired knowledge and prior experience of humans during their cognitive process.

Recently, RL-based recommendation methods are widely used for KG reasoning tasks due to their powerful ability in multi-step decision-making. Compared with these methods, we introduce System 1 to train relation-specific entity estimators based on users' historical behaviors, which can subsequently guide the noise-reducing path-finding process in System 2 and derive more proper and precise reasoning paths for explainable recommendations. We experiment with three real-world e-commerce datasets, and the results show that CogER can generate better recommendations with explanations compared with previous methods.

The contributions of our paper can be summarized as follows:

(1) To the best of our knowledge, this is the first approach that incorporates the Dual Process Theory of cognitive science into explainable recommendation, imitating human cognitive process as the convincing explanations.

(2) Inspired by the theory, we devise a cognition-aware KG reasoning model, consisting of an intuitive estimating module (System 1) and an explicit reasoning module (System 2), to explore the KG paths for explainable recommendation.

(3) We apply a relation-specific entity estimator to achieve the intuitive estimation in System 1 and a RL-based module to make explicit reasoning in System 2. These two modules interact with each other during the reasoning process until the target is reached.

2 RELATED WORK

2.1 Knowledge Graph-based Recommendation

In recent years, knowledge graphs have been introduced into recommendation systems and have shown their powerful ability to generate accurate and explainable recommendations [31, 32, 37]. According to the ways of using KGs, KG-based recommendation methods can be classified into embedding-based methods and path-based methods.

Embedding-based methods utilize KG embeddings in recommendation tasks. Some works capture rich information in the KG and use latent vectors to characterize KG entities and relations [33, 35, 40, 43]. These methods introduce the KG as side information and effectively alleviate the problem of cold start and data sparsity in recommendation. However, they don't take advantage of the structure information of KGs and fail to explain why certain items are recommended [14]. To deal with this, some researchers attempt to leverage the connective structure of KG for refining the representations of entities [4, 31, 32]. These methods all involve embedding propagation process, which can be considered as the reasoning process over KGs and thus serve as the explanation of the recommendations.

By contrast, path-based methods exploit the relational structure of the KG and provide interpretable recommendations for users. They employ KG reasoning to generate reasoning paths including the decision that has been made at each step during reasoning. These reasoning paths serve as explanations of recommendation. Some early works predefine the metapaths and make recommendations based on them [42]. These methods involve many human-defined hyper-parameters and have many limitations to explore new connection patterns. To address this, deep learning has been

applied to directly learn the representations of the connections between KG entities. For example, Wang et al. encode each path with an LSTM layer [37]. Xian et al. and Zhao et al. adopt reinforcement learning for KG reasoning [38, 45]. In this work, we focus on the path-based approach to yield recommendations with explanations.

2.2 Cognitive Science in AI

In recent years, deep learning models achieved excellent performance on many machine learning tasks [1, 15, 21, 41]. However, due to their insufficiency in interpretability, there may be problems with their actual industrial application. Improving the interpretability and reliability of AI systems is the trend of AI development in the coming years [12, 16].

Since artificial intelligence is the imitation of human intelligence, theories of brain science and cognitive science has been incorporated in the design and development of AI systems [11]. For example, Ding et al. get inspired by the Dual Process Theory [29] in cognitive science and propose a two-system framework for the multi-hop question answering (QA) task [9]. Du et al. propose the cognitive model cogKR for one-shot KG reasoning [10]. In this paper, we do not consider these models as baselines because of the characteristics of the different task scenarios. Here we draw inspiration from the two-module mechanism of human cognitive process [29] and apply it to explainable recommendation.

3 PROBLEM FORMULATION

3.1 Definitions and Notations

Definition 1 (Knowledge Graph). A knowledge graph $\mathcal{G}$ is denoted as a set of triplets $\mathcal{G} = \{(e, r, e') \mid e, e' \in \mathcal{E}, r \in \mathcal{R}\}$ where $\mathcal{E}$ is the entity set and $\mathcal{R}$ is relation set. Each triplet (e, r, e') represents a fact that head entity e has relation r to entity e'. In an E-commerce recommendation scenario, the entity set $\mathcal{E}$ consists of user set $\mathcal{U}$, item set $\mathcal{I}$, words, and items' attributes (e.g. brand, category, etc.). The relation set $\mathcal{R}$ includes the relationships between entities. Correspondingly, the triplets in the KG can be organized as follows: (1) (u, *purchase*, i): a user u purchased an item i previously; (2) (u, *mention*, w): a user u mentioned a word w in his reviews; (3) (i, *described_by*, w): an item i is described by a word w in the item's reviews; (4) (i, *belongs_to*, c): an item i belongs to a category c; (5) (i, *produced_by*, b): an item i is produced by a brand b; (6) (i_1, *also_bought*, i_2): users that purchased this item i_1 has bought another item i_2; (7) (i_1, *also_viewed*, i_2): users that viewed this item i_1 has viewed another item i_2; (8) (i_1, *bought_together*, i_2): two items i_1 and i_2 are bought together by users.

Definition 2 (Metapath). In KG reasoning, a **metapath** is a sequence of relations between entity types, which naturally describes a specific user behavior towards a product via some actions (relations) on the e-commerce platform [45]. For example, the metapath for the path in Fig.1(b) can be as $user$—($purchase$)→ $item$—($produced\_by$)→ $brand$—($produced\_by^{-1}$)→ $item$. We adopt manually predefined metapaths to filter out less useful entities in KG reasoning (discussed in Section 4.1). For the path generation, we add inverse and "no_operation" relations in KG, i.e., if $(e, r, e') \in \mathcal{G}$, then $(e', r^{-1}, e) \in \mathcal{G}$, and if $e \in E$, then $(e, r_{noop}, e) \in \mathcal{G}$. Then a reasoning path is denoted as $P_{e_0 \to e_x} = \{e_0, r_1, e_1, ..., r_x, e_x\}$, where $e_0, ..., e_x \in \mathcal{E}, r_1, ..., r_x \in \mathcal{R}$ and $(e_i, r_{i+1}, e_{i+1}) \in \mathcal{G}$, $0 \le i \le x - 1$.

3.2 Task Description

Given a knowledge graph $\mathcal{G}$ and a user $u \in \mathcal{U}$, our model is expected to recommend a set of products $\{i_n\}$ for the user u and generate corresponding reasoning paths $\{P_{u \to i_n}\}$, where $i_n \in \mathcal{I}$ and $(u, r, i_n) \notin \mathcal{G}$.

4 OUR PROPOSED MODEL COGER

In this section, we introduce our cognition-aware KG reasoning model for explainable recommendation, which imitates human cognitive process and consists of two interactive modules, i.e. System 1 and System 2. At each step, System 1 is a relation-specific entity estimator derived from users' historical behaviors to generate an estimated representation of the next-step entity, which guides the reasoning process in System 2. System 2 is a RL-based framework to select the most promising entity based on the estimated vector from System 1 and feed the entity back to System 1. The two modules work iteratively and interact with each other until the target item is reached. The architecture of our model is illustrated in Figure 2.

4.1 System 1: Relation-specific Entity Estimator

According to the theories in cognitive science, System 1 in the human cognitive process is to make a fast and intuitive judgment based on experience. Inspired by this, System 1 of our model aims at generating an intuitive estimation of the action at each step during reasoning, which can provide valuable guidance for System 2.

The whole reasoning process, namely the path-finding process, involves multiple steps on the KG. In this paper, at each step during reasoning, the relation-specific entity estimator takes as input the user embedding, relation embedding and the embedding of the current-step entity, and outputs the estimated vector of the next-step entity, which is then passed on to System 2 for explicit reasoning.

4.1.1 Metapath based Data Filtering. Since the KG contains comprehensive information about items and user-item interactions, it could be really large and contains a great number of entities and relations. However, not all the entities and relations are useful when the recommendation is made for users. Thus, we need to filter out some entities and relations that are relatively less useful for each user. We achieve this by capturing personalized path-finding patterns (i.e. metapaths) and following these patterns to perform reasoning. For each user u, we construct a personalized metapath set $\mathbb{M}_u$ based on the user's historical behaviors. To build the personalized metapath set, following previous work [39], we pre-define a set of metapaths $\mathbb{M}$ that are no longer than a pre-defined maximum length. For each user-item pair of user u, i.e., the user's historical purchase, we generate connecting paths using the stochastic bi-directional breadth-first search (BFS) under each metapath in $\mathbb{M}$. Then, we aggregate these paths of all user-item pairs and keep the top-K metapaths as the personalized metapath set $\mathbb{M}_u$ according to the frequency of metapaths. Since personalized metapaths capture users' preferences, paths under these metapaths are more likely to arrive at items of user interest. Also, these user-centric paths consist of multiple one-step triplets $(e, r, e') \in \mathcal{G}$. Therefore, they can be regarded as valuable supervision signals to learn the entity estimators of System 1.

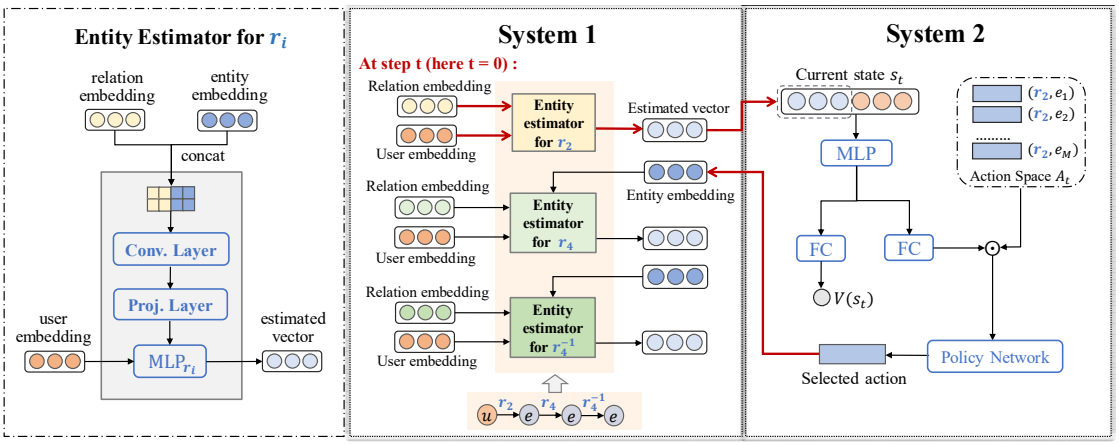


Figure 2: Architecture of our model. System 1 generates personalized metapath sets for users before reasoning. At each step during reasoning, System 1 is a relation-specific entity estimator for intuitive estimating and System 2 is a reinforcement learning framework for explicit reasoning. The red arrows indicate the workflow at the first reasoning step (t=0).

4.1.2 Entity Estimator. Having established the personalized metapaths that can distinctively characterize user behaviors, we sample a metapath m from the personalized metapath set $\mathbb{M}_u$ and follow the metapath m to conduct reasoning. Since the metapath m specifies the relation type and the entity type at each step of the reasoning process, now we just need to focus on which entity should be selected at each step. To find the most proper entity at each step, we train a set of relation-specific entity estimators which generate an estimated vector of the next-step entity based on the user and the current-step entity. For each relation r, there's a corresponding entity estimator E_r, illustrated in the left side of Figure 2. Each relation-specific entity estimator takes as input the relation embedding for r, the embedding of the current-step entity and the embedding of user. To better capture the features of relations and their connected entities in the KG, we follow previous work [8] and employ a multi-layer convolutional network here. Firstly, the input relation embedding and the entity embedding are reshaped and concatenated to form a matrix. Then a convolutional layer is applied on the matrix to generate a feature tensor. The tensor is finally fed to a projection layer and mapped into a l-dimensional vector. To avoid overfitting, dropout layers are adopted between each feature capturing layer above. Additionally, we add a relation-specific MLP φ_r after the projection layer to make better use of relation-related information. φ_r takes as input the user embedding and the l-dimensional vector above, and outputs the estimated vector of the next-step entity. The entity estimator E_r is defined as:

$$E_r(u, r, e_{t-1}) = \varphi_r\left([\boldsymbol{u}, \tau(r, e_{t-1})]\right), \tag{1}$$

$$\tau(r, e_{t-1}) = P(Conv\left([\overline{\boldsymbol{r}}, \overline{\boldsymbol{e_{t-1}}}]\right)), \tag{2}$$

where $[\cdot,\cdot]$ denotes concatenation, $\varphi_r(\cdot)$ is the relation-specific MLP with ReLU [24] as its activation function, and $\overline{\boldsymbol{r}}$ and $\overline{\boldsymbol{e_{t-1}}}$ are the reshaped embeddings of r and e_{t-1}. $P(\cdot)$ and $Conv(\cdot)$ represent the projection layer and the convolutional layer respectively, which will be elaborated in Section 5.3.

4.1.3 Training of System 1. At each step of the path, we calculate the similarity between the estimated vector with the embedding of every possible entity e_t, formulated as:

$$P(e_t \mid u, r_t, e_{t-1}) = \frac{\exp\left(\langle E_{r_t}(u, r_t, e_{t-1}), e_t\rangle\right)}{\sum_{e\in\mathcal{E}_{r_t}} \exp\left(\langle E_{r_t}(u, r_t, e_{t-1}), e\rangle\right)}, \tag{3}$$

where $\langle\cdot,\cdot\rangle$ represents the dot product operation, and $\mathcal{E}_{r_t}$ is the set of all the entities connecting with e_{t-1} by relation r_t in the KG.

Assume that e_T is an item that user u has purchased before, and $\mathbb{P} = \{u, r_1, e_1, ..., r_T, e_T\}$ is a path connecting u and e_T. To enable the estimators to find proper entities and obtain correct paths, the training goal of System 1 is to minimize the following loss:

$$\mathcal{L}_{sys1} = -\sum_{u\in\mathcal{U}}\sum_{\mathbb{P}\in\mathbb{P}_u}\sum_{t=1}^{T}\log P(e_t \mid u, r_t, e_{t-1}), \tag{4}$$

where $\mathbb{P}_u$ denotes the set of all the positive paths for user u, defined as $\mathbb{P}_u = \{\mathbb{P}_{u\to i} \mid (u, purchase, i) \in \mathcal{G},\ metapath(P_{u\to i}) \in \mathbb{M}_u\}$.

4.2 System 2: RL-based Reasoner

In cognitive science theories, System 2 aims to conduct slow, logical and reasoning thinking. To simulate this, in this paper, we design System 2 which serves to conduct logical and explicit reasoning under the guidance of signals from System 1. We use a reinforcement learning (RL) framework to implement System 2.

4.2.1 RL Framework. Reinforcement learning involves an agent, an environment, and the interactions between them. These related concepts are defined as follows.

State. During the reasoning process, the state s_t at step t is defined as:

$$s_t = [\boldsymbol{u},\ TYPE_{user},\ \boldsymbol{e_t},\ TYPE_{e_t},\ e'_{t+1}], \tag{5}$$

where $[,]$ denotes the concatenating operation, $\boldsymbol{u}$ and $\boldsymbol{e_t}$ are the embeddings of the user u and the current-step entity e_t, and e'_{t+1} represents the **estimated embedding of e_{t+1} derived from System 1**. $TYPE_{user}$ and $TYPE_{e_t}$ denote the trainable type embeddings of these two entities, which will be specified in Section 4.3. The initial state is denoted as $[\boldsymbol{u}, TYPE_{user}, \boldsymbol{u}, TYPE_{user}, \varnothing]$.

Action. At step t, the action is defined as $a_t = (r_{t+1}, e_{t+1}) \in A_t$, where e_{t+1} denotes the next-step entity and r_{t+1} denotes the relation between e_t and e_{t+1}. The full action space A_t can be formulated as $A_t = \{(r, e) \mid (e_t, r, e) \in \mathcal{G}, e \notin \{u, e_1, ..., e_{t-1}\}\}$, i.e. all the unexplored outgoing edges of the current entity e_t. Note that some entities in the KG have much more outgoing edges than the others. Reserving all the outgoing edges of every entity will lead to a waste of space. Thus, we follow previous work [38] and adopt a user-conditional action pruning strategy, which screens out the potential actions and maintains a proper size of action space. For each action $a_t = (r, e) \in A_t$, we calculate a score for it using a score function $f((r, e) \mid u)$, defined as follows:

$$f((r, e) \mid u) = \langle \boldsymbol{u} + T(u, e), \boldsymbol{e} \rangle, \tag{6}$$

where $\langle \cdot, \cdot \rangle$ represents the dot product operation. $T(u, e)$ is the aggregation of relation embeddings in the 1-reverse path[1] between the user u and the entity e. Then, all the actions in the action space are ranked by the scores and top k of them are kept in the pruned action space $A_t{}'$, which is defined as:

$$A_t{}' = \{(r, e) \mid \text{rank}(f((r, e) \mid u)) \leq k, (r, e) \in A_t\}. \tag{7}$$

Transition. Since our model imitates the cognitive process of human beings, System 1 and System 2 works iteratively and interacts with each other at each step. Given a state s_t, System 2 works and the agent selects an action $a_t = (r_{t+1}, e_{t+1})$. The action is then **passed on to System 1** to get the estimated representation of the next-step entity:

$$e'_{t+2} = E_{r_{t+2}}(u, r_{t+2}, e_{t+1}). \tag{8}$$

Then e'_{t+2} is transferred back to System 2 as part of the RL environment. The agent transits to the next state s_{t+1}:

$$s_{t+1} = \delta(s_t, a_t) = [\boldsymbol{u}, TYPE_{user}, \boldsymbol{e_{t+1}}, TYPE_{e_{t+1}}, e'_{t+2}]. \tag{9}$$

Reward. We design a threefold terminal reward to indicate the correctness of both the terminal entity and its entity type, formulated as:

$$R_T = \psi(s_T, a_T) = \begin{cases} 1, & e_T \in \mathcal{I}_u \\ 0, & e_T \in \mathcal{I} \text{ and } e_T \notin \mathcal{I}_u \\ -1, & e_T \notin \mathcal{I}, \end{cases} \tag{10}$$

where e_T denotes the terminal entity of the reasoning path, I_u is the set of items that the user has purchased before and I is the set of items in $\mathcal{G}$.

4.2.2 Actor-Critic. We adopt the actor-critic algorithm of Reinforcement Learning, which involves an actor and a critic [30].

In our model, given a state s_t at step t, the policy network π_θ generates the probability distribution of the actions in the action space A_t based on s_t. The probability of each action can be computed by:

$$\pi_\theta(a_t, s_t, A_t) = p(a_t | s_t, A_t) = \begin{cases} 0, & a_t \notin A_t{}' \\ \phi(s_t, A_t{}'), & a_t \in A_t{}', \end{cases} \tag{11}$$

where θ is the parameter of the policy network, $A_t{}'$ is the pruned action space and the function $\phi(\cdot)$ is defined as follows.

$$\phi(s_t, A_t{}') = \text{Softmax}\left(F(s_t)W_p \odot Mask_{A_t{}'}\right), \tag{12}$$

[1]A 1-reverse path has the form of $u \xrightarrow{r1} ... \xrightarrow{r_j} e_j \xleftarrow{r_{j+1}} e_{j+1} ... \xleftarrow{r_k} e_k$, which is defined and proved in [38].

Table 1: Statistics of three Amazon e-commerce datasets.

	Beauty	Cell Phones	Clothing
#Users	22,363	27,879	39,387
#Products	12,101	10,429	23,033
#Entities	224,080	163,249	425,528
#Entity Types	6	6	6
#Relation Types	16	16	16

where $\odot$ denotes the element-wise product, $W_p \in \mathbb{R}^d \times \mathbb{R}^{|A'_t|}$ represents a trainable matrix, and $F(\cdot)$ is a multi-layer perceptron (MLP) with ReLU [24] activation layers and dropout layers. $Mask_{A_t{}'} \in \mathbb{R}^{|A_t{}'|}$ is a binary vector which mask the actions in $A_t{}'$ with a rate of α.

The value network is defined as:

$$V(s_t) = F(s_t)W_V, \tag{13}$$

where $W_V \in \mathbb{R}^{d \times 1}$ is a trainable parameter. For convenience, we use Θ as the parameters of System 2, which includes the parameters of actor-critic network and the embeddings of entity types $TYPE$. The training goal is to maximize the expected cumulative reward (denoted as U_T), and the policy gradient is defined as follows:

$$\nabla J(\Theta) \approx E_\Theta \left[\sum_{t=0}^{T-1} \nabla_\Theta \log \pi_\theta(a_t, s_t) (U_T - V(s_t)) \right]. \tag{14}$$

4.3 Training

In cognitive science theories, intuitive thinking derived from the prior knowledge in System 1 is very difficult to change or manipulate [18]. In this case, we train System 1 in advance based on user-centric paths derived from user's purchase history to obtain valuable intuitive information. Overall, the parameters of our model that need to be learned during training are the parameters of the System 2, namely Θ. For each entity type in the KG, we train a type embedding $TYPE_e$ to capture the type-related features. Experimental results in Section 6.4 will show the effectiveness of these type embeddings.

5 EXPERIMENTAL SETUP

5.1 Dataset

We perform experiments on three categories of the Amazon e-commerce datasets [25], including *Clothing*, *Cell Phones* and *Beauty*. These datasets consist of product reviews, product metadata and links between products from Amazon. We construct three individual knowledge graphs based on three datasets respectively. Following previous work [3, 38], we randomly select 70% from the interaction of each user as the training set and the rest 30% as the test set. The statistics of three datasets are presented in Table 1.

5.2 Baselines and Metrics

5.2.1 Baselines. The following state-of-the-art recommendation models are considered as baselines.

BPR [27]: BPR is a pairwise ranking method for top-N recommendation, based on users' implicit feedback.

Table 2: Comparison of overall recommendation performance on three datasets. The results are calculated based on top-10 recommendations in the test set and are presented as percentages (%). Our model outperforms all baselines with paired t-test at p < 0.01 level. The best results are highlighted in bold font.

Dataset	Beauty				Cell Phones				Clothing			
Metrics	NDCG	Recall	HR	Precision	NDCG	Recall	HR	Precision	NDCG	Recall	HR	Precision
BPR	2.704	4.927	9.113	1.066	1.892	3.363	5.323	0.624	0.598	1.086	1.801	0.196
DeepCoNN	3.359	5.429	9.807	1.200	3.636	6.353	9.913	0.999	1.310	2.332	3.286	0.229
CKE	3.717	5.938	11.043	1.371	3.995	7.005	10.809	1.070	1.502	2.509	4.275	0.388
RippleNet	5.251	8.127	14.681	1.699	4.837	7.716	11.454	1.101	2.195	3.892	6.032	0.603
PGPR	5.449	8.324	14.401	1.707	5.042	8.416	11.904	1.274	2.858	4.834	7.020	0.728
ADAC	6.080	9.424	16.036	1.991	5.220	8.943	12.537	1.358	3.048	5.152	7.502	0.783
CogER-sys1	5.652	9.503	14.981	1.712	5.426	9.509	12.230	1.308	3.013	5.159	7.404	0.779
CogER-sys2	5.513	8.452	14.873	1.801	5.175	8.592	12.228	1.316	2.956	4.971	7.372	0.750
CogER (Ours)	**6.254**	**9.671**	**16.309**	**2.062**	**5.723**	**9.613**	**13.587**	**1.478**	**3.201**	**5.268**	**7.693**	**0.802**

DeepCoNN [46]: DeepCoNN leverages information in the user reviews to learn better representations of users and items.

CKE [43]: CKE is an embedding-based recommendation method that captures item representations based on heterogeneous information in the knowledge base.

RippleNet [32]: RippleNet is an end-to-end recommendation model which propogates user preferences over the entities in the knowledge graph.

PGPR [38]: PGPR is a policy-guided path reasoning algorithm that adopts reinforcement learning and an innovative reward strategy to conduct reasoning on the knowledge graphs.

ADAC [45]: ADAC is a demonstration-guided path reasoning model which extracts and leverages path demonstrations with minimal labeling efforts.

CogER-sys1: CogER-sys1 only keeps System 1 of CogER. At each step, System 1 yields the estimated representation of the next-step entity and feeds it to the following entity estimator without the reasoning in System 2.

CogER-sys2: CogER-sys2 only keeps System 2 of CogER and only contains the reinforcement learning framework.

5.2.2 Evaluation Metrics. Following [38], we apply the following four metrics to evaluate the performance of our model, including Normalized Discounted Cumulation Gain (**NDCG**), **Recall**, Hit Rate (**HR**) and **Precision**. The metrics are calculated using the top-10 recommendation results for each user in the test set.

5.3 Model Settings

For all datasets, we employ the Adam optimizer [20] with a learning rate of 10^{-4} and train our model for 60 epochs. The dimension of KG entity embedding and relation embedding is set to 100. The $Conv(\cdot)$ in Equation 2 contains a convolutional network with 32 filters of 3×3 and a dropout layer with a dropout rate of 0.2. The $P(\cdot)$ in Equation 2 contains a fully connected layer, a dropout layer (dropout rate = 0.3), and an activation ReLU layer. The type embeddings are 10 dimensional, and the estimated vectors are 100 dimensional. Previous work indicates that concise explanations with high quality are sufficient to justify recommendation results and are preferable for users [22, 38, 45], hence we limit the maximum length of reasoning paths to 3. The batch size is set to 128, and the size of the pruned action space is set to 20. The dropout layer of MLP in Equation 12 has a dropout rate of 0.5. The binary vector masks the pruned action space with a rate of 0.5.

6 RESULTS AND ANALYSIS

6.1 Overall Performance

Based on the experimental settings in Section 5.3, we evaluate the recommendation performance of our model compared with the baselines. The overall performance is shown in Table 2.

The results show that our model CogER outperforms other recommendation methods on all three datasets. For example, on the *Cell Phones* dataset, CogER achieves a 9.64% NDCG improvement, a 7.49% recall improvement, a 8.38% HR improvement, and a 8.84% precision improvement over the best baseline ADAC. The advantage of CogER can also be observed on the rest two datasets. The results indicate that CogER has the ability to conduct KG reasoning effectively and generate high-quality recommendations, with System 1 and System 2 working jointly in a mutually supplementary way. Besides, from Table 2, we observe that: (1) Compared with RL-based methods (PGPR, ADAC), our model achieves better recommendation results, which proves that System 1 of our model provides useful guidance for the RL reasoner of System 2. Namely, the interactions between System 1 and System 2 bring positive impacts to the reasoning process. (2) The embedding-based models (DeepCoNN, CKE, RippleNet) are inferior to path-based models (PGPR, ADAC). The reason is that path-based models can make better use of structural and relational information of KG.

6.2 Ablation Study

As shown in Table 2, we evaluate the CogER-sys1 and CogER-sys2 to observe the effectiveness of two systems separately.

6.2.1 Effectiveness of System 1. In the original CogER model, System 1 is responsible for making preliminary estimations and guiding the explicit reasoning of System 2. CogER-sys2 removes System 1 from our model and only keeps System 2. Table 2 shows that CogER-sys2 is inferior to CogER in recommendation performance under all four metrics on three datasets. The results prove that System 1 is effective in providing valuable information for System 2 to help it find the most proper entities at each step.

Table 3: Performance of our model w/ and w/o personalized metapath sets on the *Beauty* and the *Cell Phones* datasets.

	NDCG	Recall	HR	Precision
Beauty				
CogER	6.254	9.671	16.309	2.062
CogER-RM	6.121	9.435	16.041	2.003
Cell Phones				
CogER	5.723	9.613	13.587	1.478
CogER-RM	4.727	7.622	10.761	1.165

6.2.2 Effectiveness of System 2. System 2 considers information of the next-step entity provided by System 1 and selects the most promising action, which is sent back to System 1 for the following steps. CogER-sys1 removes System 2 from our model and only keeps System 1. As shown in Table 2, CogER outperforms CogER-sys1 on all three datasets. It's because System 2 refines the estimated entity representations from System 1 into specific entities, enabling System 1 to make better judgments at the following steps.

6.3 Effectiveness of Personalized Metapaths

To study the effectiveness of the personalized metapath set, we develop and evaluate a variant CogER-RM that randomly samples a metapath from the general metapath set $\mathbb{M}$ instead of the user-centric metapaths. The results are shown in Table 3.

We notice that on both datasets, CogER outperforms CogER-RM under all four metrics. This shows that personalized metapath sets filter out the less helpful metapaths for different users. The path-searching range is narrowed down from a large KG to a user-relevant and informative subgraph, which enables more adequate searching of our model on the KG.

Note that on the *Beauty* dataset, the recommendation performance of CogER is slightly better than CogER-RM, while on the *Cell Phones* dataset, the advantage of CogER is much larger. One possible reason is that the personalized metapaths in *Cell Phones* follow a more uneven distribution of users than those in *Beauty*. To verify the assumption, we conduct the auxiliary experiment below. Based on the personalized metapaths generated in Section 4.1, we yield the distribution of users over different metapaths in Figure 3. Figure 3 shows that the distribution on *Cell Phones* is highly uneven, with the most popular metapath (id = 8) close to 1 and the least popular metapaths (id = 3, 4, 5, etc.) equal to 0. This means that the metapath with id 8 exists in most users' personalized metapath set, while the metapath with id 3 is not in any user's personalized metapath set. Since CogER-RM randomly selects the metapath for reasoning, it is possible to sample a metapath that does not leads the agent to reach the correct target items, which further results in a drop in recommendation performance.

6.4 Effectiveness of Type Embeddings

To study the effectiveness of the type embeddings, we evaluate our model with and without type embeddings on the *Beauty* and the *Cell Phones* datasets. Besides, we use type embeddings with different dimensions to explore how the dimension influences the recommendation performance of our model. We set the range of the dimension from 10 to 40 with an interval of 10. The experimental results are presented in Figure 4.

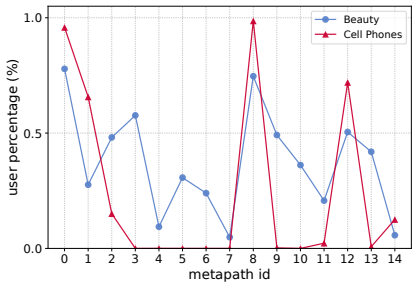
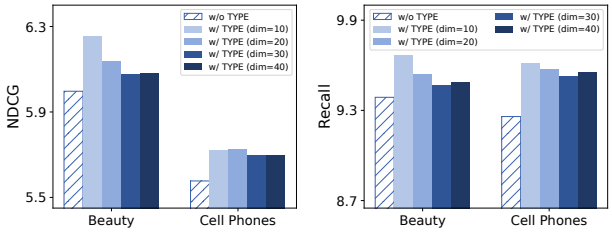


Figure 3: Distribution of users over different metapaths on the *Beauty* and the *Cell Phones* datasets.



(a) NDCG (b) Recall

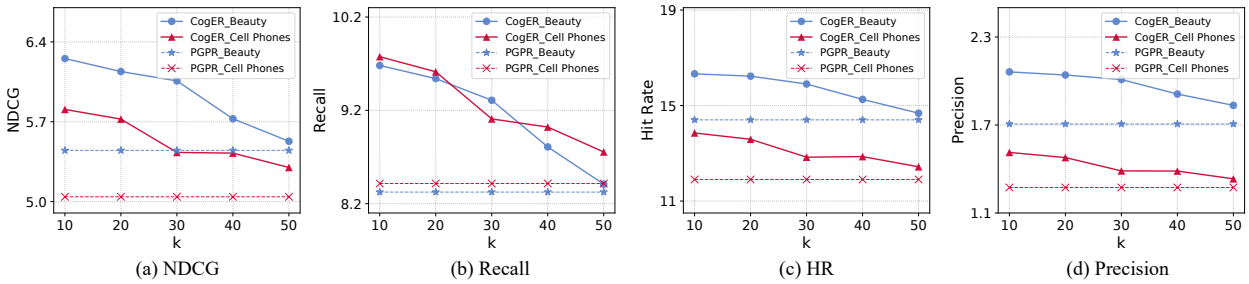
Figure 4: Performance of our model w/ and w/o type embeddings on the *Beauty* and the *Cell Phones* datasets.

There're two major discoveries in the results. (1) On both datasets, the performance of CogER with type embeddings is superior to CogER without type embeddings regardless of the dimension of type embeddings. This indicates that the type embeddings capture the type-related features of KG entities, and help improve the accuracy of our model. (2) As the dimension of type embeddings increases, there's a declining tendency of the recommendation performance of our model. One possible reason is that low-dimensional vectors are adequate to capture the type-related features of KG entities. Higher-dimensional vectors might introduce unwanted noise during training, which results in a decrease in recommendation accuracy.

6.5 Influence of the Size of Pruned Action Space

In this section, we study how the size of pruned action space influences the recommendation performance of our model. We use k as the size of pruned action space, and evaluate our model as well as a baseline PGPR on two datasets (*Beauty* and *Cell Phones*) under different k. In this experiment, we set the range of k from 10 to 50 with an interval of 10, and measure PGPR using its default parameter settings. The results are illustrated in Figure 5, which show that as the size of pruned action space k varies, our model outperforms PGPR consistently on both datasets.

Besides, Figure 5 also demonstrates that the recommendation performance of our model declines as the size of pruned action space k increases in the range of 10 to 50. Namely, the smaller



(a) NDCG (b) Recall (c) HR (d) Precision

Figure 5: Performance of our model with different sizes of pruned action space on the *Beauty* and the *Cell Phones* datasets.

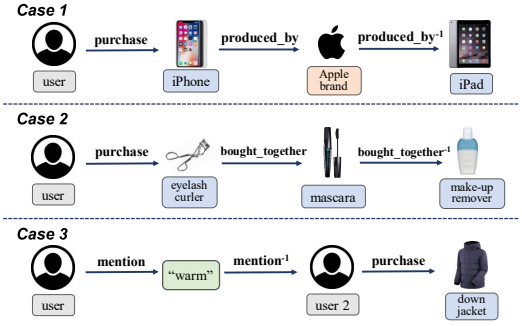


Figure 6: Three real-world cases of KG reasoning paths.

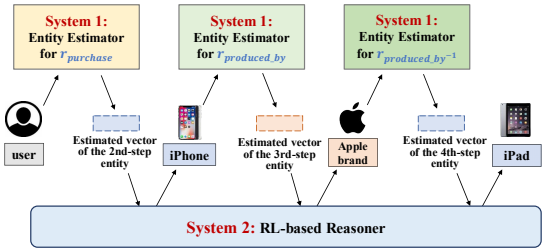


Figure 7: The cognition-aware reasoning process of a real-world case (Case 1).

the pruned action space is, the better accuracy our model can get. One possible explanation is that the scoring function used in the pruning strategy effectively filters out irrelevant actions in the full action space. Promising actions are kept and selected by the policy network with higher probability for further correct reasoning.

6.6 Case Study

As shown in Figure 6, we provide three real-world cases of KG reasoning paths derived from our model. In Case 1, our model finds the reasoning path in Fig. 6 (Case 1). The path shows, the user purchased an item "iPhone" produced by the "Apple" brand and there's another item "iPad" under the "Apple" brand. Hence, the item "iPad" is recommended to the user. According to the reasoning path, the explanation for this recommendation is that "the user might purchase this item under the same brand as a previously purchased item". Case 2 and Case 3 are another two examples. These real cases demonstrate that our model can perform KG reasoning effectively and generate good recommendations with explanations.

Further, to demonstrate the reasoning process, we provide an illustration for Case 1 in Figure 7. At the first step, the user embedding is fed to the entity estimator for relation $r_{purchase}$, which outputs the estimated vector of the next-step entity. The vector is then inputted into the RL-based reasoner of System 2 for explicit reasoning. The reasoner selects the entity "iPhone" and sends it back to System 1. The interactions continue until the reasoning path attains a certain length. As the reasoning process is finished, our model yields a reasoning path $user$—($purchase$)→ $iPhone$—($produced\_by$)→ $Apple\ brand$—($produced\_by^{-1}$)→ $iPad$, and the item "iPad" is recommended to the user.

7 CONCLUSION

In this paper, we draw inspiration from Dual Process Theory in cognitive science and propose a cognition-aware KG reasoning method CogER for explainable recommendation. CogER involves two modules: System 1 (making intuitive estimation) and System 2 (conducting explicit reasoning). At each step of reasoning, System 1 makes a preliminary estimation for the next-step action, which guides the explicit reasoning of System 2. System 2 selects a promising action using a RL framework and passes the action back to System 1. Two modules work and interact with each other iteratively until the target entity is reached. Experiments on three real-world datasets show that our model is effective in yielding high-quality recommendations with explanations.

ACKNOWLEDGMENTS

Zhicheng Dou is the corresponding author. This work was supported by the National Natural Science Foundation of China No. 62102421, No. 62272467 and No. 61872370, China Postdoctoral Science Foundation (2022T150717), Beijing Outstanding Young Scientist Program NO. BJJWZYJH012019100020098, the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China NO. 22XNKJ34, Public Computing Cloud, Renmin University of China, and Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "Double-First Class" Initiative, Renmin University of China. The work was partially done at Engineering Research Center of Next-Generation Intelligent Search and Recommendation, MOE, and Beijing Key Laboratory of Big Data Management and Analysis Methods.

REFERENCES

[1] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. 2014. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on audio, speech, and language processing* 22, 10 (2014), 1533–1545.

[2] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE transactions on knowledge and data engineering* 17, 6 (2005), 734–749.

[3] Qingyao Ai, Vahid Azizi, Xu Chen, and Yongfeng Zhang. 2018. Learning heterogeneous knowledge base embeddings for explainable recommendation. *Algorithms* 11, 9 (2018), 137.

[4] Hanxiong Chen, Yunqi Li, Shaoyun Shi, Shuchang Liu, He Zhu, and Yongfeng Zhang. 2022. Graph Collaborative Reasoning. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 75–84.

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.

[6] Zhiyong Cheng, Xiaojun Chang, Lei Zhu, Rose C Kanjirathinkal, and Mohan Kankanhalli. 2019. MMALFM: Explainable recommendation by leveraging reviews and images. *ACM Transactions on Information Systems (TOIS)* 37, 2 (2019), 1–28.

[7] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.

[8] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D knowledge graph embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 32.

[9] Ming Ding, Chang Zhou, Qibin Chen, Hongxia Yang, and Jie Tang. 2019. Cognitive graph for multi-hop reading comprehension at scale. *arXiv preprint arXiv:1905.05460* (2019).

[10] Zhengxiao Du, Chang Zhou, Ming Ding, Hongxia Yang, and Jie Tang. 2019. Cognitive knowledge graph reasoning for one-shot relational learning. *arXiv preprint arXiv:1906.05489* (2019).

[11] Kenneth D Forbus. 2010. AI and cognitive science: The past and next 30 years. *Topics in Cognitive Science* 2, 3 (2010), 345–356.

[12] Randy Goebel, Ajay Chander, Katharina Holzinger, Freddy Lecue, Zeynep Akata, Simone Stumpf, Peter Kieseberg, and Andreas Holzinger. 2018. Explainable AI: the new 42?. In *International cross-domain conference for machine learning and knowledge extraction*. Springer, 295–303.

[13] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).

[14] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering* (2020).

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[16] Andreas Holzinger. 2018. From machine learning to explainable AI. In *2018 world symposium on digital intelligence for systems and machines (DISA)*. IEEE, 55–66.

[17] Xiaowen Huang, Quan Fang, Shengsheng Qian, Jitao Sang, Yan Li, and Changsheng Xu. 2019. Explainable interaction-driven user modeling over knowledge graph for sequential recommendation. In *Proceedings of the 27th ACM International Conference on Multimedia*. 548–556.

[18] Daniel Kahneman. 2003. A perspective on judgment and choice: mapping bounded rationality. *American psychologist* 58, 9 (2003), 697.

[19] Daniel Kahneman. 2011. *Thinking, fast and slow*. Macmillan.

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[21] Siwei Lai, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Recurrent convolutional neural networks for text classification. In *Twenty-ninth AAAI conference on artificial intelligence*.

[22] Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. *arXiv preprint arXiv:1606.04155* (2016).

[23] Greg Linden, Brent Smith, and Jeremy York. 2003. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing* 7, 1 (2003), 76–80.

[24] Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Icml*.

[25] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. 188–197.

[26] Michael J Pazzani and Daniel Billsus. 2007. Content-based recommendation systems. In *The adaptive web*. Springer, 325–341.

[27] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian Personalized Ranking from Implicit Feedback. *AUAI Press* (2012).

[28] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*. 285–295.

[29] Steven A Sloman. 1996. The empirical case for two systems of reasoning. *Psychological bulletin* 119, 1 (1996), 3.

[30] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems* 12 (1999).

[31] Xiaoli Tang, Tengyun Wang, Haizhi Yang, and Hengjie Song. 2019. AKUPM: Attention-enhanced knowledge-aware user preference model for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1891–1899.

[32] 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. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 417–426.

[33] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep knowledge-aware network for news recommendation. In *Proceedings of the 2018 world wide web conference*. 1835–1844.

[34] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. 2019. Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 968–977.

[35] Hongwei Wang, Fuzheng Zhang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019. Multi-task feature learning for knowledge graph enhanced recommendation. In *The World Wide Web Conference*. 2000–2010.

[36] Xiang Wang, Xiangnan He, Fuli Feng, Liqiang Nie, and Tat-Seng Chua. 2018. TEM: Tree-enhanced embedding model for explainable recommendation. In *Proceedings of the 2018 World Wide Web Conference*. 1543–1552.

[37] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable reasoning over knowledge graphs for recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 5329–5336.

[38] Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. 2019. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. 285–294.

[39] Yikun Xian, Zuohui Fu, Handong Zhao, Yingqiang Ge, Xu Chen, Qiaoying Huang, Shijie Geng, Zhou Qin, Gerard De Melo, Shan Muthukrishnan, et al. 2020. CAFE: Coarse-to-fine neural symbolic reasoning for explainable recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1645–1654.

[40] Xin Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon Jose. 2019. Relational collaborative filtering: Modeling multiple item relations for recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 125–134.

[41] Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP*.

[42] Xiao Yu, Xiang Ren, Yizhou Sun, Bradley Sturt, Urvashi Khandelwal, Quanquan Gu, Brandon Norick, and Jiawei Han. 2013. Recommendation in heterogeneous information networks with implicit user feedback. In *Proceedings of the 7th ACM conference on Recommender systems*. 347–350.

[43] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 353–362.

[44] Yongfeng Zhang and Xu Chen. 2018. Explainable recommendation: A survey and new perspectives. *arXiv preprint arXiv:1804.11192* (2018).

[45] Kangzhi Zhao, Xiting Wang, Yuren Zhang, Li Zhao, Zheng Liu, Chunxiao Xing, and Xing Xie. 2020. Leveraging demonstrations for reinforcement recommendation reasoning over knowledge graphs. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 239–248.

[46] Lei Zheng, Vahid Noroozi, and Philip S Yu. 2017. Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the tenth ACM international conference on web search and data mining*. 425–434.