Mining Implicit Entity Preference from User-Item Interaction Data for Knowledge Graph Completion via Adversarial Learning

Gaole He[1,3], Junyi Li[2], Wayne Xin Zhao[2,3]*, Peiju Liu[4] and Ji-Rong Wen[2,3]
[1]School of Information, Renmin University of China
[2]Gaoling School of Artificial Intelligence, Renmin University of China
[3]Beijing Key Laboratory of Big Data Management and Analysis Methods
[4]School of Electronics Engineering and Computer Science, Peking University
{hegaole, lijunyi, jrwen}@ruc.edu.cn, batmanfly@gmail.com, liupage2016@pku.edu.cn

ABSTRACT

The task of Knowledge Graph Completion (KGC) aims to automatically infer the missing fact information in Knowledge Graph (KG). In this paper, we take a new perspective that aims to leverage rich user-item interaction data (*user interaction data* for short) for improving the KGC task. Our work is inspired by the observation that many KG entities correspond to online items in application systems. However, the two kinds of data sources have very different intrinsic characteristics, and it is likely to hurt the original performance using simple fusion strategy.

To address this challenge, we propose a novel adversarial learning approach by leveraging user interaction data for the KGC task. Our generator is isolated from user interaction data, and serves to improve the performance of the discriminator. The discriminator takes the learned useful information from user interaction data as input, and gradually enhances the evaluation capacity in order to identify the fake samples generated by the generator. To discover implicit entity preference of users, we design an elaborate collaborative learning algorithms based on graph neural networks, which will be jointly optimized with the discriminator. Such an approach is effective to alleviate the issues about data heterogeneity and semantic complexity for the KGC task. Extensive experiments on three real-world datasets have demonstrated the effectiveness of our approach on the KGC task.

CCS CONCEPTS

• **Computing methodologies** → *Knowledge representation and reasoning*.

KEYWORDS

Knowledge Graph, Adversarial Learning, User Preference

ACM Reference Format:
Gaole He, Junyi Li, Wayne Xin Zhao, Peiju Liu and Ji-Rong Wen. 2020. Mining Implicit Entity Preference from User-Item Interaction Data for Knowledge Graph Completion via Adversarial Learning. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380155

*Corresponding author.

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380155

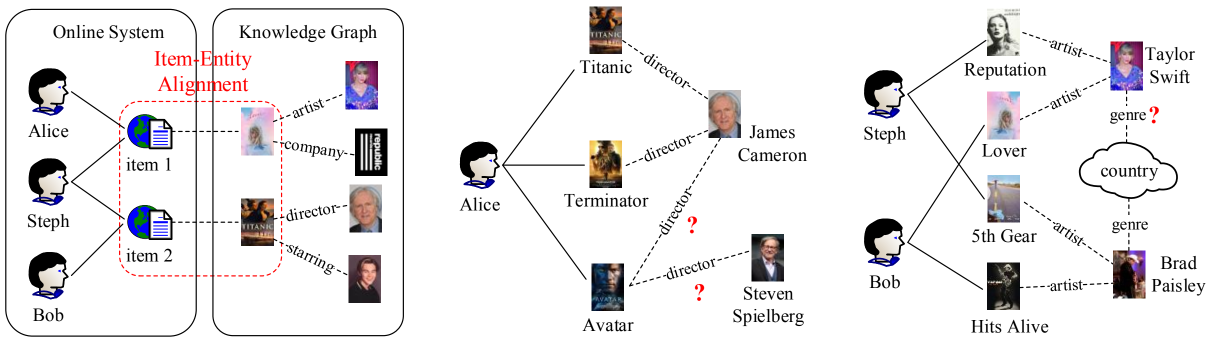
1 INTRODUCTION

Recent years have witnessed the great thrive and wide application of large-scale knowledge graph (KG). Although many existing KGs [2, 9, 26, 27] are able to provide billions of structural facts about entities, they are known to be far from complete [7]. Hence, various methods have been proposed to focus on the task of knowledge graph completion (KGC) [3, 6, 38]. Typically, KG represents a fact as a triple consisting of head entity, relation, tail entity. Based on this data form, the KGC task is usually described as predicting a missing entity in an incomplete triple.

Most of previous KGC methods aim to devise new learning algorithms to reason about underlying KG semantics using known fact information. In this work, we take a different perspective for tackling the KGC task. Since KG has been widely used in various applications, can we leverage the accumulated application data for improving the KGC task? Specially, we are inspired by the observation that many KG entities correspond to online items in application systems. As shown in [40, 41], the items (*i.e.,* movies) from MovieLens have largely overlapped with the KG entities in Freebase. For KG entities aligned to online items, we can obtain fact triples from the KG as well as rich user-item interaction data (called *user interaction data* for short) from the application platforms (See Fig. 1(a)). Based on this observation, the focus of this work is to study how user interaction data can be utilized to improve the KGC task.

User interaction data has explicitly reflected users' preference at the item level, while it is likely to contain implicit evidence about entity semantics, which is potentially useful to our task. Here, we present two illustrative examples. In Fig. 1(b), the user "*Alice*" has watched three movies of "*Terminator*", "*Titanic*" and "*Avatar*", and she is a fan for the director of "*James Cameron*". Given a query about the director of "*Avatar*" and two candidate directors "*James Cameron*" and "*Steven Allan Spielberg*", knowing the user's interaction history is useful to identify the correct director in this case. As another example in music domain (See Fig. 1(c)), the users of "*Steph*" and "*Bob*" like the songs from both singers "*Taylor Swift*" and "*Brad Paisley*" due to the similar style. Such co-occurrence patterns in user interaction data are helpful to infer whether the two singers share the same artist genre in KG. From two examples, it can be seen that user interaction data may contain useful preference information of users over KG entities.

Indeed, several recent efforts have attempted to leverage both KG data and user interaction data for jointly improving the KGC task and related recommendation tasks, including path-based methods [28], regularization-based methods [1, 20] and graph neural network methods [36]. These studies mainly focus on developing



(a) Item-entity alignment **(b) Query = < Avatar, isDirectedBy, ? >** **(c) Query = < Taylor_Swift, genre, ? >**

Figure 1: Illustrative examples for our work: (a) item-entity alignment across online systems and KG entities in movie and music domain; (b) inferring the director for the movie "*Avatar*"; and (c) inferring the artist genre for "*Taylor Swift*".

data fusion models for integrating the two kinds of data sources, *e.g.,* learning representations in the same space or share the same information representation across different sources. However, the two kinds of data sources have very different intrinsic characteristics, and it is likely to hurt the original representation performance using simple fusion strategy. In addition, user interaction data is usually very noisy since user behaviors will be affected by external events (*e.g.,* on sale) or other influencing factors (*e.g.,* popularity). It may be problematic to directly incorporate the learned information (*e.g.,* user preference) for inferring KG facts. To solve our task, we have to consider the effect of data heterogeneity and semantic complexity on model design. The major challenge can be summarized as: (1) how to learn useful information from user interaction data for improving KGC task and (2) how to integrate or utilize the learned information in KGC methods.

As shown in Fig. 1, we can see that implicit entity preference of users is helpful to infer the plausibility of KG facts. Based on this motivation, our idea is to develop a specific evaluation component that incorporates and learns user preference information about entities for evaluating a candidate entity given a query (*i.e.,* the head entity and relation). Meanwhile, we keep a prediction component to produce the candidate entity without using user preference information. Since the prediction component tries to pass the check of the evaluation component by producing high-quality answers, it will tune and improve itself according to the feedback of the evaluation component. The two components will be improved via a mutual reinforcement process. By mapping the two components to discriminator and generator respectively, our idea naturally fits into the successful framework of generative adversarial nets (GAN) [8]. In our setting, the discriminator is expected to effectively integrate the two kinds of heterogeneous data signals for the KGC task. While, the generator is employed to improve the discriminator by modeling a pure KG semantic space.

To this end, we propose a novel adversarial learning approach for leveraging user interaction data for the KGC task, named as *UPGAN* (User Preference enhanced GAN). The proposed approach contains three major technical extensions. First, to learn useful evidence from user interaction data, we integrate the two kinds of data sources and construct an interaction-augmented KG. Based on this graph, we design a two-stage representation learning algorithm for collaboratively learning entity-oriented user preference and preference-enhanced entity representation. The obtained entity representation is able to encode implicit entity preference of related users with high-order connectivity on the KG. Second, we design a user preference guided discriminator for evaluating the plausibility of a candidate entity given a query. Besides original KG data, our discriminator is able to utilize the learned preference-enhanced entity representations. Third, we design a query-specific entity generator for producing hard negative entities. Its major role is to improve the discriminator by learning to sample negative samples from the candidate pool.

Our approach adopts a "safer and more careful" way to utilize user interaction data for the KGC task. We design an elaborate collaborative learning algorithms for learning implicit entity preference of users from their interaction data. Our generator is relatively isolated from user interaction data, and improves itself according to the feedback from the discriminator. The discriminator takes entity-oriented user preference as input, and gradually enhances the evaluation capacity in order to defend the increasingly hard fake samples generated by the generator. Such an approach is effective to alleviate the issues about data heterogeneity and semantic complexity that were raised earlier. To evaluate our approach, we construct extensive experiments on three real-world datasets. Extensive experiments have demonstrated the effectiveness of our approach on the KGC task, especially for entities with relatively sparse triples.

The rest of this paper is organized as follows. We first introduce the related work in Section 2. Then, the preliminary and the proposed approach are presented in Section 3 and 4, respectively. The experimental results are summarized in section 5, and we conclude the paper in section 6.

2 RELATED WORK

Our work is closely related to the studies on knowledge graph completion (KGC), collaborative recommendation and KGC models, and generative adversarial networks (GAN).

Knowledge Graph Completion. For the KGC task, various methods have been developed in the literature by adopting different technical approaches. Translation-based embedding methods, *e.g.,* TransE [3] and its variants [17, 37], model relational fact as directed translation from head entity to tail entity. Semantic matching based methods [6, 18, 31, 38] serve as another line of research, which try to learn triple plausibility in relational semantic space with bilinear semantic matching. More recently, Graph Neural Network (GNN) [15, 32] has received much attention as an effective technique to learn node embeddings over graph-structured data. Several studies try to utilize GNN to capture semantic relations on the KG, such as relational convolution [24] and structural convolution [25]. However, these methods mainly focus on modeling KG graph structure, which cannot effectively integrate user interaction data.

Collaborative Recommendation and KGC Models. Recently, several studies try to develop collaborative models for the two tasks of item recommendation and KGC, including co-factorization model [20], relation transfer [5], multi-task learning [34] and graph neural networks [36]. In these studies, either shared information is modeled or the same representation space is adopted. As we discussed, user interaction data is very noisy, and it may be problematic to simply combine the two kinds of data sources. Especially, most of these works have set up two optimization objective considering improving both recommendation and KGC. As a comparison, we only consider the KGC task, and user interaction data is only utilized as an auxiliary source. Besides, a series of works [14, 28, 39] have been proposed to incorporate knowledge graph to improve the quality and explainability of recommendation.

Generative Adversarial Networks. GANs [8, 19] have been one of the most breakthrough learning techique in recent years. The GAN framework provides a general, effective way to estimate generative models via an adversarial process, in which we simultaneously train two models namely generator and discriminator. The original GAN [8] aims to generate realistic simulation pictures with continuous data representation. Recently, there are quite a few studies that adapt GAN to model data with discrete graph structure, such as graph data [33] and heterogenous information network [13]. These works mainly focus on general graph based tasks (*e.g.,* node classification), which are not directly applicable to our task. Especially, GAN has also been used in knowledge graph completion [4, 35]. Their core idea is to enhance the training of existing KGC methods by generating high-quality negative samples, which do not consider other external signals.

Compared with these studies, our focus is to leverage user interaction data for the KGC task with an adversarial learning approach. We design an elaborate model architecture to effectively fuse user interaction data in the discriminator, and utilize a separate generator to produce high-quality "fake samples" to help improve the discriminator.

3 PRELIMINARY

In this section, we first introduce the KGC task, then describe the construction details of interaction-augumented knowledge graph based on entity-to-item alignment, and finally present our task.

Knowledge Graph Completion (KGC). A knowledge graph typically organizes fact information as a set of triples, denoted by $\mathcal{T}_{KG} = \{\langle h, r, t\rangle | h, t \in \mathcal{E}, r \in \mathcal{R}\}$, where $\mathcal{E}$ and $\mathcal{R}$ denote the entity set and relation set, respectively. A triple $\langle h, r, t\rangle$ describes that there is a relation r between head entity h and tail entity t regarding to some fact. For example, a triple $\langle$AVATAR, DIRECTEDBY, JAMESCAMERON$\rangle$ describes that the movie of *"Avatar"* is directed by *"James Cameron"*. Since not all the facts have corresponding triples in KG, the KGC task aims to automatically predict triples with missing entities, either a tail entity $\langle h, r, ?\rangle$ or a head entity $\langle ?, r, t\rangle$. Without loss of generality, in this paper, we only discuss the case with a missing tail entity, *i.e.,* $\langle h, r, ?\rangle$. For convenience, we call a KG triple with a missing entity *a query*, denoted by $q = \langle h, r, ?\rangle$. A commonly adopted way by KGC methods is to embed entities and relations into low-dimensional latent space [3, 38], and then develop a scoring function for predicting the plausibility of a triple. Hence, we introduce $\boldsymbol{v}_h \in \mathbb{R}^K$, $\boldsymbol{v}_r \in \mathbb{R}^K$ and $\boldsymbol{v}_t \in \mathbb{R}^K$ to denote the embeddings for head entities, relations and tail entities, respectively.

User Interaction. In online systems, we can obtain rich use interaction data with items. Formally, user-item interaction data can be characterized as a set of triples $\mathcal{T}_{UI} = \{\langle u, r_{int}, i\rangle, | u \in \mathcal{U}, i \in \mathcal{I}\}$, where $\mathcal{U}$ and $\mathcal{I}$ denote the user set and item set respectively, and the triple $\langle u, r_{int}, i\rangle$ indicates that there is an observed interaction r_{int} (*e.g.,* purchases and clicks) between user u and item i. According to specific tasks or datasets, we can define multiple kinds of user-item interaction relations. Here, for simplicity, we only consider a single interaction relation r_{int}. An interesting observation is that a KG entity usually corresponds to an online item in user-oriented application systems [41]. For instance, the Freebase movie entity "*Avatar*" (with the Freebase ID *m.0bth54*) has an entry of a movie item in IMDb (with the IMDb ID *tt0499549*). Such a correspondence is called *entity-to-item alignment* across KG and online application systems.

Interaction-Augmented Knowledge Graph. Considering the overlap between KG entities and online items, we introduce an extended entity graph to unify the KG information and user interaction data. The extended knowledge graph consists of a union set of triples based on KG and online systems: $\mathcal{G} = \{\langle h, r, t\rangle | h, t \in \widetilde{\mathcal{E}}, r \in \widetilde{\mathcal{R}}\}$, where $\widetilde{\mathcal{E}} = \mathcal{E} \cup \mathcal{U} \cup \mathcal{I}$, $\widetilde{\mathcal{R}} = \mathcal{R} \cup \{r_{int}\}$ and $\mathcal{G} = \mathcal{T}_{KG} \cup \mathcal{T}_{UI}$. A major difference with traditional KG is the incorporation of user nodes and user interaction with items into the graph. We introduce a general placeholder n (n_j and n_k) to denote any node on the graph. Note that although a KG entity has a corresponding item, we only keep a single node for a KG entity in the graph. Since our task is to leverage user interaction data for learning useful evidence to the KGC task, we organize the entity graph in a user-oriented layer-wise structure. Specially, user nodes are placed on the first layer, then the aligned entities (which correspond to online items) are placed on the second layer. The other nodes are organized in layers according to their shortest distance (*i.e.,* minimum hop number) for arriving at any user node. Let d_n denote the minimum hop number from a node n to user nodes. We can see that $d_n = 0, \forall n \in \mathcal{U}$, and $d_n = 1, \forall n \in \mathcal{I}$, and $d_n > 1, \forall n \in \mathcal{E} \setminus \mathcal{I}$. In this way, entities with the same distance will be placed at the same layer.

Task Description. Given a query triple $\langle h, r, ?\rangle$ or $\langle ?, r, t\rangle$, we aim to predict the missing entity given both the KG information and user interaction data. In what follows, we will focus on the former query case for describing our approach. While, our experiments will consider both cases for evaluation.

4 THE PROPOSED APPROACH

In this section, we present the proposed approach, *UPGAN (User Preference enhanced GAN)*, for the KGC task by leveraging user interaction data based on adversarial learning.

4.1 Overview

As discussed earlier, user interaction data is quite different from KG data in intrinsic characteristics. It is likely to bring irrelevant information or even noise if simply integrating it into the KGC method. Considering data heterogeneity and semantic complexity, we design an adversarial learning approach to utilizing useful information from user interaction data for the KGC task.

We set up two components with different purposes for the KGC task, namely prediction component (*i.e.,* generator G) and evaluation component (*i.e.,* discriminator D). The generator G produces a candidate answer for the missing entity, and the discriminator D evaluates the plausibility of the generated answer by G. The two components force each other to improve in a mutual reinforcement way. Our focus is to train a capable discriminator that is able to leverage KG information for the KGC task, and the role of the generator is to improve the discriminator and help the fusion of user interaction data. In this way, we can fully utilize useful evidence from user interaction data in the discriminator, and meanwhile avoid direct influence of user interaction data on the KG semantic space modeled by the generator.

Following GANs [4, 8, 33], we cast our problem as a minimax game between two players, namely generator G (parameterized by θ^G) and discriminator D (parameterized by θ^D), for our KGC task:

$$\begin{aligned}\min_{\theta^G}\max_{\theta^D} \quad & \mathbb{E}_{\langle h,r,t\rangle\sim P_{\mathcal{T}_{KG}}} \log D(t|h,r;\mathcal{G},\theta^D) \\ + \quad & \mathbb{E}_{\langle h,r,?\rangle\sim P_{\mathcal{T}_{KG}}, a\sim G} \log\left(1 - D(a|h,r;\mathcal{G},\theta^D)\right).\end{aligned} \tag{1}$$

where $a \sim G(h, r; \theta^G)$ denotes a generated entity by the generator. The discriminator would drive the generator to produce more better candidates, and the generator would improve the discriminator by providing more hard fake samples. By repeating such a mutual improvement process, we expect a more effective KGC solution can be derived. Note $\mathcal{G} = \mathcal{T}_{KG} \cup \mathcal{T}_{UI}$, consisting of KG triples and user-item interaction triples, has been incorporated into the discriminator D. To model the information on the heterogeneous graph $\mathcal{G}$, we develop a collaborative representation learning algorithm based on graph neural networks for extracting useful user preference information from user interaction data.

We present an overall sketch of the proposed approach in Fig. 2. In what follows, we first introduce how to learn suitable representations from $\mathcal{G}$, and then describe the discriminator and generator.

4.2 Collaborative Representation Learning over Interaction Augmented KG

As shown in Fig. 2, user interaction data explicitly reflects user preference at the item level, and we would like to learn and utilize implicit entity-oriented preference of users in the semantic space of KG. Our solution is to learn effective node embeddings over the interaction-augmented KG $\mathcal{G}$, which is expected to encode useful preference evidence for enhancing KG entity representations. A straightforward method is to treat all the graph nodes equally

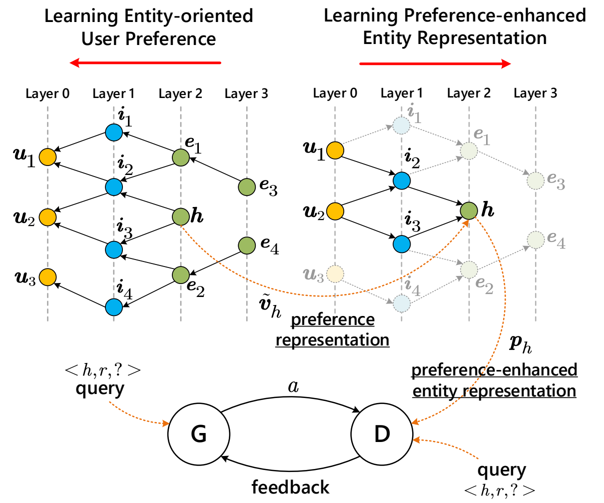


Figure 2: The overview of the proposed UPGAN model. The orange, blue and green nodes represent the users, items interacted with users, and entities in KG, respectively.

and employ a standard graph neural network model to learn node embeddings. However, it may incorporate irrelevant information or noise into node representations due to node heterogeneity. To address this issue, we design an elaborative two-stage collabarative learning algorithm based on user-oriented graph neural networks.

4.2.1 *Learning Entity-oriented User Preference.*

Recall that user nodes are placed at the bottom layer, and other entity nodes are at a higher layer. In the first stage, we preform the information propagation from KG entities to users. The update strategy is a combination between the original embedding and the received embeddings from forward triples:

$$\tilde{\boldsymbol{v}}_{n_j} = \sigma(\boldsymbol{W}_0^D \boldsymbol{v}_{n_j} + \sum_{\langle n_j,r,n_k\rangle\in\mathcal{F}_{n_j}} \frac{1}{|\mathcal{F}_{n_j}|} \boldsymbol{W}_r^D \tilde{\boldsymbol{v}}_{n_k}). \tag{2}$$

where $\boldsymbol{v}_{n_j}$ is the original learned or initialized node representation, n_j/n_k denotes a node on the graph (can be a user, item or entity), $\mathcal{F}_{n_j} = \{\langle n_j, r, n_k\rangle | d_{n_j} = d_{n_k} - 1, \langle n_j, r, n_k\rangle \in \mathcal{G}\}$ denotes the set of forward triples (an entity links to another connected entity at the next layer) for entity n_j, and $\boldsymbol{W}_0^D$ and $\boldsymbol{W}_r^D$ denote the transformation matrices for the original representation and relation r, respectively. With this update formula, a node on the graph can collect related entity semantics from its upstream neighbors. By organizing nodes in layers, the entities closer to users have a greater impact on user preference. The update for user embeddings is performed at the last step, which alleviates the influence of noisy interaction data. Another merit is that the propagation implicitly encodes path semantics into the node representations, which has been shown important to consider in the KGC task [10, 16]. When this stage ends, each user node will be learned with a preference representation $\tilde{\boldsymbol{v}}_u$ based on Eq. 2, encoding her/his preference over entity-level semantics.

4.2.2 *Learning Preference-enhanced Entity Representation.*

In the second stage, given a query triple $\langle h, r, ?\rangle$, we would like to collect user preference information over entity semantics on the graph regarding to the target entity h. For example, in Fig. 1(b), knowing the preference of user "*Alice*" is helpful to answer the query regarding to the director for entity "*Avatar*". For this purpose, we perform an inverse aggregation from user nodes to the target entity as follows:

$$\boldsymbol{p}_{n_j} = \sum_{\langle n_j, r, n_k \rangle \in \mathcal{B}_{n_j}} \alpha_{n_j,r,n_k} \boldsymbol{p}_{n_k}, \tag{3}$$

where $\mathcal{B}_{n_j} = \{\langle n_j, r, n_k \rangle | d_{n_j} = d_{n_k}+1, \langle n_j, r, n_k \rangle \in \mathcal{G}\}$ denotes the set of backward triples (an entity links to another connected entity at the previous layer) for entity n_j, and α_{n_j,r,n_k} is the attention coefficient for aggregation defined as

$$\begin{aligned} \alpha_{n_j,r,n_k} &= \frac{\exp(\pi(n_j, r, n_k))}{\sum_{\langle n_j, r', n_{k'} \rangle \in \mathcal{B}_{n_j}} \exp(\pi(n_j, r', n_{k'}))}, \\ \pi(n_j, r, n_k) &= \text{LeakyReLU}(\boldsymbol{w}^\top [\boldsymbol{W}_r^D \tilde{\boldsymbol{v}}_{n_j}; \boldsymbol{W}_r^D \boldsymbol{p}_{n_k}]). \end{aligned} \tag{4}$$

where $\tilde{\boldsymbol{v}}_{n_j}$ is the learned representation in Section 4.2.1. Before running the aggregation procedure, we first initialize $\boldsymbol{p}_n$ as $\tilde{\boldsymbol{v}}_n$. Given a target entity, our aggregation update indeed spans a tree-like structure (See Fig. 2), and only reachable nodes on the graph are activated in this process. When this stage ends, we can derive an updated representation for the target entity h, which encodes the preference information passed from activated user nodes, denoted by $\boldsymbol{p}_h$.

4.2.3 Discussion. We have designed an elaborate two-stage learning algorithm over the interaction-augmented KG. The update in both stages is directed. The first stage propagates entity semantics to user nodes, which aims to learn entity-oriented user preference; the second stage collects the learned user preference at the target entity, which aims to learn preference-enhanced entity representations. When the involved weight parameters are fixed, it can be proved that $\boldsymbol{p}_h$ is indeed a linear combination of user preference representations (learned in the first stage), given the fact that we aggregate the information by layer and start from the first layer of user nodes. It can be formally given as:

$$\boldsymbol{p}_h = \sum_{u \in \mathcal{U}} w_{h,u} \tilde{\boldsymbol{v}}_u, \tag{5}$$

where $\tilde{\boldsymbol{v}}_u$ is the user embeddings learned in the first stage (Eq. 2), and $w_{h,u}$ (set to zero for unactivated users) can be computed according to the accumulative attention coefficients along the paths from user u to target entity h. Indeed, these activated users are high-order connectable nodes to the target entity. Besides the learned semantic representation $\boldsymbol{v}_h$, we enhance the entity representation using the entity-level preference of the users with high-order connectivity.

4.3 User Preference Guided Discriminator

In our approach, the major function of the discriminator is to distinguish between real and fake answers given the query. Compared with previous GAN-based KGC methods [4, 35], a major difference is that we would incorporate the learned preference-enhanced entity representations for improving the discriminator.

4.3.1 Discriminator Formulation. Our discriminator $D(t|h, r; \mathcal{G}, \theta^D)$ evaluates whether the entity t can be the answer to a given query $\langle h, r, ? \rangle$ by computing the following probability:

$$D(t|h, r; \mathcal{G}, \theta^D) = \frac{1}{1 + \exp(-s(h, r, t; \mathcal{G}, \theta^D))}, \tag{6}$$

where $s(\cdot)$ is the score function measuring the plausibility of the triple $\langle h, r, t \rangle$. Here, we give a general form for $s(\cdot)$, and many previous methods can be used to instantiate it, such as TransE [3] and DistMult [38]. We incorporate the preference-enhanced entity representation $\boldsymbol{p}_h$ for improving the evaluation capacity of the discriminator as follows:

$$s(h, r, t; \mathcal{G}, \theta^D) = (\boldsymbol{W}_2 \boldsymbol{v}_t + \boldsymbol{b}_2)^\top \cdot \tanh(\boldsymbol{W}_1 \boldsymbol{x}_q + \boldsymbol{b}_1), \tag{7}$$

where $\boldsymbol{W}_1, \boldsymbol{W}_2$ and $\boldsymbol{b}_1, \boldsymbol{b}_2$ are parameter matrices or vectors, $s(\cdot)$ takes as input the query embedding $\boldsymbol{x}_q$ and candidate entity embedding $\boldsymbol{v}_t$, and $\tanh(\cdot)$ is incorporated as a non-linear transformation function that can be replaced by other functions. $\boldsymbol{x}_q$ is composed of two parts: the learned entity embeddings using KG information and the enhanced entity representation from user interaction data, formally given as

$$\boldsymbol{x}_q = [\underbrace{\boldsymbol{v}_h \odot \boldsymbol{v}_r}_{\text{KG information}}; \underbrace{\boldsymbol{p}_h \odot \boldsymbol{v}_r}_{\text{preference information}}], \tag{8}$$

where $\boldsymbol{p}_h$ is defined in Eq. 5 reflecting the related user preference regarding to entity h. In this way, user preference over KG entities on the graph $\mathcal{G}$ has been considered into the discriminator. A good candidate answer should not only match the query well in the KG, but also meet the semantic requirement of the entity preference of users with high-order connectivity.

4.3.2 Discriminator Loss. To optimize the discriminator, we consider two cases for computing the loss. First, the real answer entity t to the query $\langle h, r, ? \rangle$ on the knowledge graph $\mathcal{T}_{KG}$ should be recognized as positive by the discriminator. Second, the discriminator tries to identify the generated answer by the generator $G(h, r; \theta^G)$ as negative. The loss of the two cases can be given as follows:

$$\begin{aligned} \mathcal{L}^D = &\mathbb{E}_{\langle h,r,t \rangle \sim P_{\mathcal{T}_{KG}}} - \log D(t|h, r) + \\ &\mathbb{E}_{\langle h,r,? \rangle \sim P_{\mathcal{T}_{KG}}, t' \sim G(h,r;\theta^G)} - \log(1 - D(t'|h, r)) + \lambda^D ||\theta^D||_2^2, \end{aligned} \tag{9}$$

where $\lambda^D > 0$ controls the regularization term to avoid overfitting. Given a query, the real answers from the KG population is considered as the positive cases, and the generated entities from the generator G as the negative cases. The parameter θ^D of the discriminator can be optimized by minimizing $\mathcal{L}^D$. Note that although we describe the learning of $\boldsymbol{p}_h$ and the discriminator in different sections, they are bound through the discriminator objective and will be learned jointly. With increasingly hard samples from the generator, the discriminator jointly optimizes its own parameters and the involved parameters in Section 4.2. In this way, the entity-oriented user preference $\tilde{\boldsymbol{v}}_u$ and enhanced entity representation $\boldsymbol{p}_h$ are gradually transformed into a suitable representation for the KGC task.

4.4 Query-specific Entity Generator

In our approach, the major function of the generator is to provide high-quality negative entities to improve the discriminator. We design a query-specific entity generator by sampling from the candidate entity pool. Since user interaction data itself is likely to contain noise, the generator would not utilize any user interaction data and model a pure KG semantic space.

4.4.1 Generator Formulation. For each query $\langle h, r, ?\rangle$, we assume that a candidate entity set $C^{h,r} \subset \mathcal{E}$ can be first constructed, *e.g.,* using existing KGC methods or random sampling. Then, our generator defines a distribution over the candidate set and samples from it. Given a query $q = \langle h, r, ?\rangle$, we compute the query representation $\boldsymbol{v}_q^G$ as

$$\boldsymbol{v}_q^G = \boldsymbol{v}_h \odot \boldsymbol{v}_r. \tag{10}$$

We can implement $\boldsymbol{v}_q^G$ in other ways as needed. Note that KG embeddings $\boldsymbol{v}_h$ and $\boldsymbol{v}_r$ are not necessarily the same as those in the discriminator. To enhance the robustness of our generator, we concatenate the query representation with a noise $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I})$, which is a Gaussian distribution with zero mean and covariance $\sigma^2\boldsymbol{I}$:

$$\boldsymbol{e}_q^G = [\boldsymbol{v}_q^G; \boldsymbol{z}]. \tag{11}$$

Finally, the concatenated vector is fed into a Multi-Layer Perceptron (MLP), which is activated with non-linear function LeakyReLU. The probability distribution to sample a candidate entity from $C^{h,r}$ is defined as:

$$G(a|h, r; \theta^G) = \frac{\exp(\mathrm{MLP}(\boldsymbol{e}_q^G) \cdot \boldsymbol{v}_a)}{\sum_{t' \in C^{h,r}} \exp(\mathrm{MLP}(\boldsymbol{e}_q^G) \cdot \boldsymbol{v}_{t'})}. \tag{12}$$

With this distribution, we sample n_G entities from the candidate set, which are taken as input for the discriminator as negative samples.

4.4.2 Policy Gradient. Since sampling an entity from the candidate set is a discrete process, we do not directly optimize the loss for the generator. Here, we follow KBGAN [4] to adopt policy gradient [29] for parameter learning. A key point is how to set the reward function appropriately. Here, we utilize the feedback of the discriminator as the reward signal to guide the learning of the generator.

$$R(a|h, r) = \frac{\exp(s(a|h, r; \mathcal{G}, \theta^D))}{\sum_{t' \in C^{h,r}} \exp(s(t'|h, r; \mathcal{G}, \theta^D))} - b, \tag{13}$$

where the score function $s(\cdot)$ is defined in Eq. 7 and we set $b = \frac{1}{|C^{h,r}|}$ as the bias. Here, we incorporate the bias b by considering uniform sampling as a reference. When a sample receives a larger probability by the discriminator than the average, it would be assigned with a positive reward by our approach. Formally, we optimize the following loss for the generator:

$$\mathcal{L}^G = \mathbb{E}_{\langle h, r, ?\rangle \sim P_{\mathcal{T}_{KG}}, a \sim G(h, r; \theta^G)} R(a|h, r) + \lambda^G ||\theta^G||_2^2, \tag{14}$$

where $\lambda^G > 0$ controls the regularization term to avoid overfitting. To optimize the above loss, the policy used by the generator would punish the trivial negative entities by lowering down their corresponding probability, and encourage the network to assign a larger probability to the entities that can bring higher reward.

4.5 Optimization and Discussion

In this part, we discuss the model optimization and comparison with previous works.

To learn our model, we first pretrain the discriminator component with training data. Then, we follow the standard training algorithms for GAN-based models [8] by alternating between the G-step and D-step at an iteration. We adopt a mini-batch update strategy. For each training triple $\langle h, r, t\rangle$ in a batch, the generator will first randomly sample n_C entities from the entire entity set (excluding observed true answers) as the candidate pool $C^{h,r}$. Since the entire entity set is likely to contain false negatives (*i.e.,* true answers), we empirically find that n_C should not be set to a very large number. After that, the generator G samples n_G entities from the n_C candidates as negative samples using Eq. 12. Then, we update the parameters of G according to the loss in Eq. 14 using policy gradient [29]. For the discriminator, given a query from the training set, it minimizes the loss in Eq. 9 over the real answer and the n_G fake samples from the generator.

Note that the parameters involved in the graph neural networks in Section 4.2 will be also optimized in the learning process of D, since D directly uses the learned node embeddings from it. We first identify the users that are activated (*i.e.,* reachable) by the entities from a batch. After that, we employ these users as seeds to construct a subgraph for local parameter update. Based on the subgraph, we perform an entity-to-user information propagation according to Section 4.2.1, and then learn the preference-enhanced entity representations only for the query entities in the sampled batch according to Section 4.2.2. Since we span a tree-like structure for this procedure, it can be efficiently implemented with tree traverse algorithms. To encourage the discriminator to estimate soft probabilities, we adopt the label smoothing trick [22] to train our UPGAN.

Although there have been a few studies which either adopt GAN or utilize user interaction data for improving the KGC task, our approach has two major differences. First, our adversarial approach is developed based on an effective two-stage learning algorithm for integrating both entity semantics and user preference. As a comparison, user interaction information has been seldom considered in previous GAN based methods for the KGC task. Second, we do not directly incorporate the learned information from user interaction into the generator. Its major role is to improve the discriminator by producing high-quality fake samples. To our knowledge, it is the first time that user interaction data has been utilized for the KGC task in an adversarial learning approach.

5 EXPERIMENT

In this section, we perform the evaluation experiments for our approach on the KGC task. We first introduce the experimental setup, and then report the results and detailed analysis.

5.1 Dataset Construction

In our setting, we need an aligned linkage between KG data and user interaction data. Here, we adopt the KB4Rec dataset [41] to construct the evaluation datasets, containing the alignment records between Freebase entities [9] and online items from three domains.

Freebase stores facts by triples of the form $\langle$*head*, *relation*, *tail*$\rangle$, and we use the last public version released on March 2015. The three user interaction datasets are MovieLens movie [11], LFM-1b music [23] and Amazon book [12]. For all datasets, we only keep the interactions related to the linked items. The *LFM-1b* music dataset is very large, and we take the subset from year 2012; while for the *MovieLens* 20m dataset, we take the subset from year 2005 to 2015. Following [21], we only keep the k-core dataset, and filter out unpopular items and inactive users with fewer than k interaction records, which is set to 10 for the music dataset and 5 for the other two datasets.

After preprocessing the three user interaction datasets, we take the left aligned entities as seeds, and generate the KG subgraph by performing breadth-first-search in each domain. We aim to examine the performance improvement of queries about both aligned entities and their reachable entities via a few hops on the KG. In our experiments, we set the maximum BFS hop to be four. Following [3, 30], we removed relations like *<book.author.works_written>* which just reverses the head and tail compared to the relations *<book.written_work.author>*. We also removed relations that end up with non-freebase string, *e.g.*, like *<film.film.rottentomatoes_id>*. To ensure the KG quality, we filter infrequent entities with fewer than k KG triples, which is set to 3 for the book dataset and 10 for the other two datasets. We summarize the statistics of three datasets after preprocessing in Table 1. Overall, the user interaction data in the book domain is sparser than the other two domains. Furthermore, for each domain, we randomly split it into training set, validation set and test set with a ratio of 8:1:1.

Table 1: Statistics of our datasets after preprocessing.

Dataset		Movie	Music	Book
User Interaction	#Users	61,859	57,976	75,639
	#Items	17,568	55,431	22,072
	#Interactions	9,908,778	2,605,262	831,130
Knowledge Graph	#Entities	56,789	108,930	79,682
	#Relations	47	45	38
	#Triplets	953,598	914,842	400,787

5.2 Experimental Setting

This part presents the basic experimental settings.

5.2.1 Evaluation Protocol. We follow [3] to cast the KGC task as a ranking task for evaluation. For each test triple $\langle h, r, t\rangle$ in a dataset, two queries, $\langle h, r, ?\rangle$ and $\langle ?, r, t\rangle$, were issued in the following way. Each missing entity (*i.e.,* ground truth) will be combined with the rest entities as a candidate pool (excluding other valid entities). Given a query, a method is required to rank the order of the entities in the candidate list, and a good method tends to rank the correct entity in top positions. To evaluate the performance, we adopt a variety of evaluation metrics widely used in previous works, the Mean Rank (MR) [3], top-k hit ratio (H@k) [3], and Mean Reciprocal Rank (MRR) [38]. Specifically, MR refers to the average rank of all testing cases, H@k is defined as the percentage of the testing triples that have a rank value no greater than k, and MRR is the average of the multiplicative inverse of the rank value for all testing triples. For all the comparison methods, we learn the models using the training set, and optimize the parameters using the validation set and compare their performance on the test set.

5.2.2 Methods to Compare. We consider the following methods for performance comparison:

- **TransE** [3]: TransE model introduces translation-based embedding, modeling relations as the translations operating on entities.
- **DistMult** [38]: It is based on the bilinear model where each relation is represented by a diagonal rather than a full matrix.
- **ConvE** [6]: It is a link prediction model that uses 2D convolution over embeddings and multiple layers of non-linear features.
- **ConvTransE**: [25]: ConvTransE enable the state-of-the-art ConvE to be translational between entities and relations while keeps the same link prediction performance as ConvE.
- **KBGAN** [4]: It utilizes pretrained KG embedding models as generator to selectively generate hard negative samples, and improves the performances of target embedding models.
- **R-GCN** [24]: It is related to a recent class of neural networks operating on graphs, and is developed specifically to handle the highly multi-relational data characteristic of realistic KGs.
- **KTUP** [5]: It jointly solve recommendation and KGC tasks, transfering the relation information in KG, so as to understand the reasons that a user likes an item.
- **CoFM** [20]: It is a multi-task co-factorization model which optimizes both item recommendation and KGC task jointly.
- **KGAT** [36]: Built upon the graph neural network framework, KGAT explicitly models the high-order relations in collaborative knowledge graph with item side information.
- **UPGAN**: It is our approach.

Our baselines have a comprehensive coverage of the related models. To summarize, we categorize the baselines into several groups shown in Table 2, according to the *technical approaches* and *utilization of user interaction data.* All the models have some parameters to tune. We either follow the reported optimal parameters or optimize each model separately using validation set. Following [6, 25], we equip semantic-matching based methods with $1 - N$ scoring strategy, including DistMult that previously adopted a simple binary entropy cross loss.

Table 2: The categorization of the comparison methods. "UI" is the abbreviation for user interaction.

Category	Translation	Semantic match	GNN
KG	TransE	DistMult,ConvE, ConvTransE	R-GCN
KG+GAN	KBGAN		
KG+UI	KTUP,CoFM	–	KGAT
KG+UI+GAN	UPGAN (our approach)		

5.2.3 Implementation Details. For our approach, we adopt the DistMult [38] model to initialize the KG related parameters, and train each individual component to converge for (at most) 1000 epochs. To avoid overfitting, we adopt early stopping by evaluating MRR on the validation set every 20 epochs. We optimize all models with Adam optimizer, where the batch size is set to 4096. The coefficient of L2 normalization is set to 10^{-5}, and the embedding size is set to 100, and the learning rate is tuned amongst {0.01, 0.005, 0.001,

Table 3: Performance comparison of different methods for KGC task on three datasets. We use bold and underline fonts to denote the best and second best performance in each metric respectively. Besides MR, the results are given in precent (%).

Models	Movie					Music					Book				
	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10	MR	MRR	H@1	H@3	H@10
TransE	1941	18.7	12.3	20.5	32.2	864	61.7	53.7	66.6	76.9	5694	31.7	25.3	34.9	44.1
DistMult	**1218**	25.2	18.4	27.8	38.5	2153	68.4	62.3	72.5	79.3	6676	34.9	29.3	37.8	45.7
ConvE	1671	24.6	18.3	27.0	36.9	1620	69.3	63.7	73.0	79.4	4858	33.0	27.0	36.0	44.3
ConvTransE	1450	25.0	18.5	27.5	37.8	1203	69.9	63.9	73.8	80.6	3995	33.4	27.0	36.8	45.4
R-GCN	1261	24.4	18.0	26.6	37.0	1565	68.4	62.6	72.0	78.9	6438	32.8	27.6	35.2	42.1
KBGAN	2324	20.9	14.8	23.2	33.3	995	63.2	55.8	67.7	77.1	6539	32.3	26.2	35.3	44.4
CoFM	1936	18.8	12.3	20.6	32.2	2204	62.4	54.5	67.1	77.4	5695	31.7	25.3	35.0	44.1
KTUP	1960	19.3	12.7	21.2	32.8	851	62.0	54.1	66.8	77.0	5456	32.1	25.7	35.3	44.5
KGAT	1347	20.1	13.8	22.2	32.3	**593**	62.5	53.6	68.2	78.4	**2670**	34.1	27.6	37.1	46.0
UPGAN	1666	**25.9**	**18.8**	**28.9**	**39.4**	1050	**71.8**	**65.8**	**75.9**	**82.1**	3463	**37.0**	**30.6**	**40.5**	**48.8**

0.0005, 0.0001}. The entity embeddings are constrained to have a length no smaller than 1. In each iteration, we set n_G as 200 and n_C as 1024. For the generator, the MLP components contain two hidden layers with the LeakyReLU activation function.

5.3 Results and Analysis

The results of different methods for knowledge graph completion task are presented in Table 3. It can be observed that:

(1) Among baselines which only use KG data, TransE performs worst since it usually adopts very simple distance function for fitting training triples. Three semantic match based methods DistMult, ConvE and ConvTransE give better results than TransE, which have used a more powerful match function for modeling the semantics of a triple. The GNN based method R-GCN shows a more competitive performance than TransE, while it performs worse than semantic match based methods. Overall, DistMult and ConvTransE are the best baseline methods.

(2) KBGAN is the only GAN based baseline, which mainly aims to produce high-quality negative samples than random sampling. As we can see that, it substantially improves over TransE on all datasets, which indicates the usefulness of adversarial learning. However, KBGAN only utilizes the information from the KG triples, and its improvement is relatively limited, and cannot perform better than the competitive baselines DistMult and ConvTransE. Besides, for a query, DistMult and ConvTransE adopt a new $1-N$ scoring function [6] as the enhanced loss by iterating over all the candidate entities. We speculate that the usefulness of $1-N$ scoring strategy is mainly due to *candidate exposure* by simply treating all the entities from the entire candidate set to be negative.

(3) Overall, the three methods that jointly utilize KG data and user interaction data seem to give slightly better results than TransE. Among these methods, CoFM and KTUP are indeed constructed based on translation based methods. KGAT has developed a collaborative graph neural network for learning the embeddings over the heterogeneous nodes. It achieves a better performance on book dataset than the other two datasets.

(4) Finally, we compare the proposed approach UPGAN with the baseline methods. It is clear to see that UPGAN is consistently better than these baselines by a large margin. As shown in Table 2, our method jointly utilizes the KG and user interaction data using a GAN-based approach. Different from the above joint models, we optimize the performance of the KGC task as the only objective. Especially, we adopt an elaborative way to incorporate the learned user preference. We only utilize the user interaction data in the discriminator, while the major role of the generator models is to improve the discriminator. The generator is improved according to the feedback of the discriminator, which can be considered as indirect signal from user interaction data.

5.4 Detailed Analysis of Performance Improvement

As shown in Table 3, our proposed approach UPGAN shows a better overall performance than the baselines. Here, we zoom into the results and check whether UPGAN is indeed better than baselines in specific cases. For ease of visualization, we only incorporate the results of DistMult and ConvTransE as the reference, since they perform generally well among all the baselines.

Table 4: Performance (H@3 in precent) comparison *w.r.t.* different sparsity levels. *%Improv.* means the improvement ratio of UPGAN over the strongest baseline. We use "$A-E$" to denote the five groups with a decreasing sparsity level.

Dataset	Models	A	B	C	D	E
Movie	DistMult	10.8	14.4	14.6	22.1	77.0
	ConvTransE	10.4	13.7	14.4	**22.4**	76.7
	UPGAN	**13.3**	**15.7**	**14.8**	21.6	**78.9**
	%Improv.	+23.1%	+9.0%	+1.4%	-3.6%	+2.5%
Music	DistMult	73.5	71.9	74.2	72.3	70.5
	ConvTransE	73.8	72.5	75.1	74.1	73.6
	UPGAN	**76.7**	**74.3**	**77.0**	**75.1**	**76.3**
	%Improv.	+3.9%	+2.5%	+2.5%	+1.3%	+3.7%
Book	DistMult	8.1	15.6	33.7	47.5	84.1
	ConvTransE	7.3	15.0	33.9	45.8	82.0
	UPGAN	**8.9**	**17.7**	**36.4**	**52.4**	**87.1**
	%Improv.	+9.9%	+13.5%	+7.4%	+10.3%	+3.6%

5.4.1 Performance Comparison w.r.t. Sparsity Levels . In KG, different entities correspond to a varying number of triples. Various methods need sufficient training triples for learning good entity representations. Here, we examine how our method improves over the baseline methods, especially in the sparse case. For this purpose, we first divide the test queries into five groups *w.r.t.* the frequency of the answer entity. A smaller group ID indicates that the answer entity of that case occur fewer in training set. We present the comparison results in Table 4. We can see that overall our approach is substantially better than over baseline methods in five sparsity levels. Especially, on movie and book datasets, it yields a larger improvement in sparse groups.

5.4.2 Performance Comparison w.r.t. Hop Number. In our dataset, only the aligned KG entities correspond to interaction data from external application systems. We have constructed an interaction-augmented KG, and try to learn high-order relatedness between users and entities. The preference learned from such high-order relatedness has been verified to be effective in improving the KGC task. Hence, we would like to check how the distance of a KG entity to user nodes affects the performance. We consider three groups, namely aligned entities (1-hop), attributional entities corresponding to aligned entities (2-hop) and other entities (3-hop and more). We present the performance comparison of the three groups in Table 5. It can be seen that our method has yielded a substantial improvement in all three groups. The finding indicates that our two-stage learning algorithm is able to perform effective information propagation and learning over the heterogeneous graph. Interestingly, the 1-hop entities do not always receive the most improvement. Indeed, we have found that the improvement is mainly related to the query difficulty instead of the hop number.

Table 5: Performance (H@3 in precent) comparison *w.r.t.* different hop numbers. *%Improv*. means the improvement ratio of UPGAN over the strongest baseline.

Datasets	Hops	Baselines		UPGAN	%Improv.
		DistMult	ConvTransE		
Movie	1	42.8	43.1	**45.0**	(+4.4%)
	2	8.3	8.0	**8.5**	(+2.4%)
	>=3	60.7	59.2	**62.0**	(+2.1%)
Music	1	89.1	89.6	**91.3**	(+1.9%)
	2	71.8	74.0	**76.3**	(+3.1%)
	>=3	32.5	33.1	**34.9**	(+5.4%)
Book	1	66.5	64.2	**70.1**	(+5.4%)
	2	16.3	16.6	**18.2**	(+9.6%)
	>=3	55.5	53.0	**58.7**	(+5.8%)

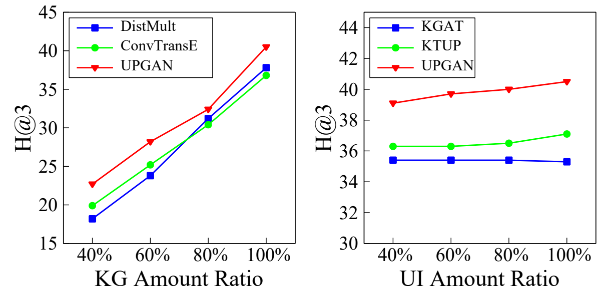
5.4.3 Ablation Study. To effectively utilize the user interaction data, our approach has made several technical extensions. Here, we examine how each of them affects the final performance. We consider the following variants of our approach for comparison:

- *UPGAN$_{\neg G}$*: the variant with only the discriminator component.
- *UPGAN$_{\neg UI}$*: the variant drops the enhanced entity representation from $\boldsymbol{x}_q$ (Eq. 8). In other words, the two-stage learning component has been removed.
- *UPGAN$_{R-GCN}$*: the variant replaces the two-stage learning component with a neural network architecture similar to R-GCN [24]. In this variant, we treat all the types of nodes equally.

In Table 6, we can see that the performance order can be summarized as: UPGAN$_{\neg UI}$ < UPGAN$_{R-GCN}$ < UPGAN$_{\neg G}$ < UPGAN. These results indicate that the proposed techniques are useful to improve the performance. Especially, user interaction data with a suitable modeling way is more important for our approach.

Table 6: Ablation analysis on the book dataset (in percent).

Models	MR	MRR	H@1	H@3	H@10
UPGAN	**3463**	**37.0**	**30.6**	**40.5**	**48.8**
UPGAN$_{\neg G}$	3546	36.1	29.4	39.8	48.1
UPGAN$_{R-GCN}$	3883	35.8	29.8	39.0	47.0
UPGAN$_{\neg UI}$	5501	35.0	28.8	38.3	46.7



(a) Varying the amount of KG triples. (b) Varying the amount of user interaction data.

Figure 3: Performance tuning on AMAZON book dataset.

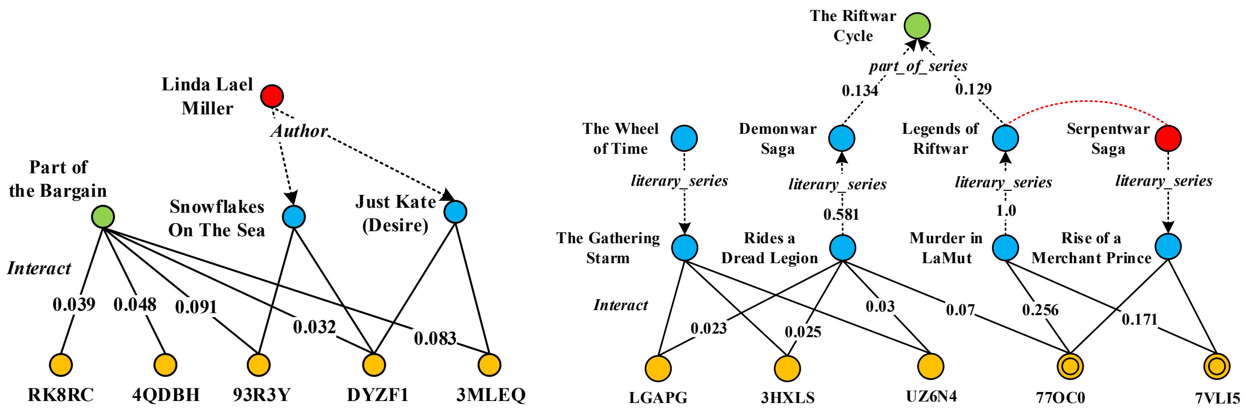
5.5 Performance Sensitivity Analysis

In this part, we further investigate the influence of training data and model parameters on the performance. Due to space limit, we only report the results on the book dataset, and omit similar results of the two datasets.

5.5.1 Varying the amount of KG triples. The amount of available KG information directly influences the performance of various KGC methods. Here we examine how our approach performs with the varying amount of KG triples. We select DistMult and ConvTransE as comparison methods. We take 40%, 60%, 80% and 100% from the complete training data to generate four new training sets, respectively. The test set is fixed as original. Fig. 3(a) presents the H@3 performance *w.r.t.* different ratios of KG triples. It can be seen that UPGAN is consistently better than DistMult and ConvTransE with four training sets, especially performs best with an extremely sparse (40%) amount of KG triples. This observation implies that UPGAN is able to alleviate the influence of data sparsity for KGC methods to some extent. Besides, it can yield more improvement with fewer KG triples.

5.5.2 Varying the amount of user interaction data. Since our approach utilizes user interaction data for the KGC task, we continue to examine how its amount affects the final performance. As comparisons, we select two collaborative recommendation and KGC models, namely KGAT and KTUP. Similarly, we take 40%, 60%, 80% and 100% from the complete user interaction data to generate four new datasets respectively. The training set of KG triples and the test set are fixed as original. As we can see from Fig. 3(b), UPGAN is substantially better than KGAT and KTUP for all the four ratios, which indicates the effectiveness of our approach in leveraging user interaction data. Another observation is that the performance of UPGAN gradually increases and the change is relatively stable.

Besides data amount, we also examine the effect of two parameters, namely the embedding dimensions K and the number of hidden layers in the generator. Overall, we find that it yields a good



(a) Querying the *author* of "*Part of Bargain*". (b) Querying the *literary series* of "*The Riftwar Cycle*".

Figure 4: Two cases from AMAZON book dataset. We use green, red, blue and yellow circles to denote the target entity, correct entity, KG entity and user respectively. The weights on the edges are computed by our approach. Since the number of the reachable users from the target node is large, we only present five selected users for illustration.

performance when $K = 128$, where the other values in the set {16, 32, 64, 128, 256} give worse results. While, for another parameter, our experiment results show that using two hidden layers give the best performance while the change with other numbers in {1, 2, 3, 4} is very small. Due to space limit, we omit the results here.

5.6 Case Study

In this part, we present two cases for illustrating how our approach utilizes user interaction data for the KGC task.

The first case is related to a query about the *author* for the book "*Part of Bargain*". In our training set, there are few related triples for the book entity "*Part of Bargain*". By only considering KG information, it is difficult for a KGC method to identify the correct answer, since the learned entity representations are not reliable with very limited training data. When incorporating the user-item interaction data, we can clearly see that it has several overlapping users with the other two books "*Snowflakes On The Sea*" and "*Just Kate (Desire)*". Interestingly, the three related books are written by the same author "*Linda Lael Miller*". By running our approach, we can identify the correct answer to this query.

The second case is related to a query about the relation *part_of_series* for the book series, which aims to identify the literary series (*a.k.a.,* sub-series) that belong to "*The Riftwar Cycle*" (target entity). Following the first case, we check whether the related users on the graph can be useful for this query. Starting from the target entity, we can identify 128 related users in total with the BFS extension based on the interaction-augmented KG. Given two candidate literary series "*Serpentwar Saga*" and "*The Wheel of Time*", a straightforward method is to count the number of a literary series that has been read by the related users. However, "*The Wheel of Time*" is much more popular than the correct entity "*Serpentwar Saga*" (33 *v.s.* 17). It indicates that simply using the user interaction data may incorporate noise. As a comparison, by running our approach, we can identify more important users on the graph. As we can see, the two users with ID "77OC0" and "7VLI5" are assigned with very large attention weights by our algorithm. An interesting observation is that "*Legends of the Riftwar*" and "*Serpentwar Saga*" can be associated via the two selected users. Based on the known fact that "*Legends of the Riftwar*" belongs to "*The Riftwar Cycle*", our approach is capable of identifying "*Serpentwar Saga*" as the final answer.

6 CONCLUSION

In this paper, we developed an adversarial learning approach for effectively learning useful information from user interaction data for the KGC task. Especially, we have made three major technical contributions. First, we constructed an interaction-augmented KG for unifying KG and user interaction data, and design a two-stage representation learning algorithm for collaboratively learning effective representations for heterogeneous nodes. Second, by integrating enhanced entity representations, we designed a user preference guided discriminator for evaluating the plausibility of a candidate entity given a query. Third, we designed a query-specific generator for producing hard negative entities for given a query. We constructed evaluation experiments with three large datasets. The results showed that our proposed model is superior to previous methods in terms of effectiveness for the KGC task.

Currently, only three datasets with aligned entity-item linkage have been used for evaluation. We believe our approach is applicable to more domains. In the future, we will investigate into how our models perform in other domains.

ACKNOWLEDGEMENT

This work was partially supported by the National Natural Science Foundation of China under Grant No. 61872369 and 61832017, the Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of China under Grant No. 18XNLG22 and 19XNQ047, and Beijing Outstanding Young Scientist Program under Grant No. BJJWZYJH012019100020098, and Beijing Academy of Artificial Intelligence (BAAI). Xin Zhao is the corresponding author.

REFERENCES

[1] Qingyao Ai, Vahid Azizi, Xu Chen, and Yongfeng Zhang. 2018. Learning Heterogeneous Knowledge Base Embeddings for Explainable Recommendation. *Algorithms* 11, 9 (2018), 137.

[2] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: A Nucleus for a Web of Open Data. In *Proceedings of the 6th International The Semantic Web and 2Nd Asian Conference on Asian Semantic Web Conference* (Busan, Korea) *(ISWC'07/ASWC'07)*. 722–735.

[3] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.* 2787–2795.

[4] Liwei Cai and William Yang Wang. 2018. KBGAN: Adversarial Learning for Knowledge Graph Embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. 1470–1480.

[5] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Chua Tat-seng. 2019. Unifying Knowledge Graph Learning and Recommendation: Towards a Better Understanding of User Preference. In *WWW*.

[6] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *AAAI-18*.

[7] Luis Galárraga, Simon Razniewski, Antoine Amarilli, and Fabian M. Suchanek. 2017. Predicting Completeness in Knowledge Bases. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017.* 375–383.

[8] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada.* 2672–2680.

[9] Google. 2016. Freebase Data Dumps. https://developers.google.com/freebase/data.

[10] Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing Knowledge Graphs in Vector Space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015.* 318–327.

[11] F. Maxwell Harper and Joseph A. Konstan. 2016. The MovieLens Datasets. *TiiS* 5, 4 (2016), 1–19.

[12] Ruining He and Julian Mcauley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *WWW*.

[13] Binbin Hu, Yuan Fang, and Chuan Shi. 2019. Adversarial Learning on Heterogeneous Information Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019.* 120–129.

[14] Jin Huang, Wayne Xin Zhao, Hong-Jian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018.* 505–514.

[15] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings.*

[16] Yankai Lin, Zhiyuan Liu, Huan-Bo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling Relation Paths for Representation Learning of Knowledge Bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015.* 705–714.

[17] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.* 2181–2187.

[18] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011.* 809–816.

[19] Zhaoqing Pan, Weijie Yu, Xiaokai Yi, Asifullah Khan, Feng Yuan, and Yuhui Zheng. 2019. Recent Progress on Generative Adversarial Networks (GANs): A Survey. *IEEE Access* 7 (2019), 36322–36333.

[20] Guangyuan Piao and John G. Breslin. 2018. Transfer Learning for Item Recommendations and Knowledge Graph Completion in Item Related Domains via a Co-Factorization Model. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings.* 496–511.

[21] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized Markov chains for next-basket recommendation. In *International Conference on World Wide Web.* 811–820.

[22] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved Techniques for Training GANs. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain.* 2226–2234.

[23] Markus Schedl. 2016. The LFM-1b Dataset for Music Retrieval and Recommendation. In *ICMR*.

[24] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings.* 593–607.

[25] Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-end Structure-Aware Convolutional Networks for Knowledge Base Completion. (2019).

[26] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan Wang. 2015. An Overview of Microsoft Academic Service (MAS) and Applications. In *Proceedings of the 24th International Conference on World Wide Web* (Florence, Italy) *(WWW '15 Companion)*. 243–246.

[27] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: A Core of Semantic Knowledge. In *Proceedings of the 16th International Conference on World Wide Web* (Banff, Alberta, Canada) *(WWW '07)*. 697–706.

[28] Zhu Sun, Jie Yang, Jie Zhang, Alessandro Bozzon, Long-Kai Huang, and Chi Xu. 2018. Recurrent knowledge graph embedding for effective recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018.* 297–305.

[29] Richard S. Sutton, David A. McAllester, Satinder P. Singh, and Yishay Mansour. 1999. Policy Gradient Methods for Reinforcement Learning with Function Approximation. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999].* 1057–1063.

[30] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. 2015. Representing Text for Joint Embedding of Text and Knowledge Bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015.* 1499–1509.

[31] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016.* 2071–2080.

[32] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations* (2018).

[33] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. GraphGAN: Graph Representation Learning With Generative Adversarial Nets. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18).* 2508–2515.

[34] 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, WWW 2019, San Francisco, CA, USA, May 13-17, 2019.* 2000–2010.

[35] PeiFeng Wang, Shuangyin Li, and Rong Pan. 2018. Incorporating GAN for Negative Sampling in Knowledge Representation Learning. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18).* 2005–2012.

[36] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. In *KDD*.

[37] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.* 1112–1119.

[38] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.*

[39] 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, San Francisco, CA, USA, August 13-17, 2016.* 353–362.

[40] Wayne Xin Zhao, Hong-Jian Dou, Yuanpei Zhao, Daxiang Dong, and Ji-Rong Wen. 2019. Neural Network Based Popularity Prediction by Linking Online Content with Knowledge Bases. In *Advances in Knowledge Discovery and Data Mining - 23rd Pacific-Asia Conference, PAKDD 2019, Macau, China, April 14-17, 2019, Proceedings, Part II.* 16–28.

[41] Wayne Xin Zhao, Gaole He, Kunlin Yang, Hong-Jian Dou, Jin Huang, Siqi Ouyang, and Ji-Rong Wen. 2019. KB4Rec: A Data Set for Linking Knowledge Bases with Recommender Systems. *Data Intelligence* 1, 2 (2019), 121–136.