Path-enhanced Explainable Recommendation with Knowledge Graphs

Yafan Huang, Feng Zhao, Shihui Song, Xiangyu Gui, Hai Jin*
National Engineering Research Center for Big Data Technology and System, Services Computing Technology and System Lab, Cluster and Grid Computing Lab
School of Computer Science and Technology, Huazhong University of Science and Technology
Wuhan, Hubei, China
{hyfshishen,zhaof,songsh,guixy,hjin}@hust.edu.cn

ABSTRACT

Knowledge graph (KG), which proves to be an effective tool to enhance recommender systems with rich semantics, has captured growing research attention recently. By mining multi-hop relations (named as *path*) between user-item interactions within a KG, implicit user preference and other side information can be clearly uncovered. Nevertheless, existing recommendation methods not only have fundamental limitations in explainability, but also under-utilize user-item path sets and show poor performance in handling cold-start issues, which indicates reliable recommendation results require massive prior knowledge.

To better address these issues, we propose a novel model architecture named *Path-enhanced Recurrent Network* (**PeRN**). Specifically, PeRN integrates a recurrent neural network (RNN) encoder with an meta-path-based entropy encoder to further increase explainability and reduce cold-start costs. RNN-based model has a strong ability to represent sequential path semantics in KG, while entropy encoder, as an efficient statistical analysis approach, leverage meta path information to differentiate paths in one user-item interaction. Moreover, we improve previous path extraction algorithms with a bidirectional scheme to make PeRN more feasible. Comprehensive experiment results on two real-world datasets, compared with several state-of-the-art baselines, demonstrate our significant improvements with reasonable explanations and minimal amount of prior knowledge in constructing KG.

CCS CONCEPTS

• **Information systems → Recommender systems**; • **Computing methodologies → Semantic networks**; *Information extraction.*

KEYWORDS

Recommender System, Knowledge Graph, Meta Path, Recurrent Neural Network

*Corresponding author

ACM Reference Format:
Yafan Huang, Feng Zhao, Shihui Song, Xiangyu Gui, Hai Jin. 2018. Path-enhanced Explainable Recommendation with Knowledge Graphs. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Recommendation system (RS), aiming to locate user preference and provide items that the user might be interested in, has been witnessed rapid growth during the past decade in search engines, video portals and E-commerce. Collaborative Filtering (CF) as a significant recommendation approach, has attracted much research effort [5, 8, 27]. Despite its developments and universality in several popular benchmarks, CF methods do suffer a lot from the faultiness of incapability to harness side information and their excessively single explanation [20]. To solve these problems, various representation learning models (i.e. embedding methods) have been proposed such as Wide&Deep [6], Attentional Factorization Machine (AFM) [30], *etc.* These methods achieve promising results by embedding recommendation data into a low-dimensional continuous vector space and alleviate the disadvantages of CF to a certain extent with encouraging accuracy. But they regard each user-item interaction as an irrelevant vector and do not take relations into account, which means these embedding methods still remain a poor explainability in recommendation.

In order to resolve the limitation of CF and representation learning models, knowledge graph (KG), as a well-structured auxiliary data form evolving from semantic web, has been naturally applied in recommendations to boost its reasoning ability and explainability [21, 24]. Information in KG is organized by triples, e.g. *(head, relation, tail)*, combining *head* and *tail* entities with relations (*relation*). In KG-enhanced recommendation methods, all the users and items can be regarded as entities together with other background information. By exploiting multi-hop relations from target user entity within a KG, the user preference and its semantics can be explicitly revealed. Such correlation is recorded as a path and shown as the following example:

Example: *(Tom, Like, Billie Jean)* $\wedge$ *(Billie Jean, SungBy, Michael Jackson)* $\wedge$ *(Michael Jackson, Sing, Scream)* $\Rightarrow$ *(Tom, mayLike, Scream)*
Evidently, each target user entity owns copious paths which lead to different items that user potentially likes, offering precise sequential information for bi-classification and top-K recommendation tasks. As a result, these path-based methods [12, 26] soon received

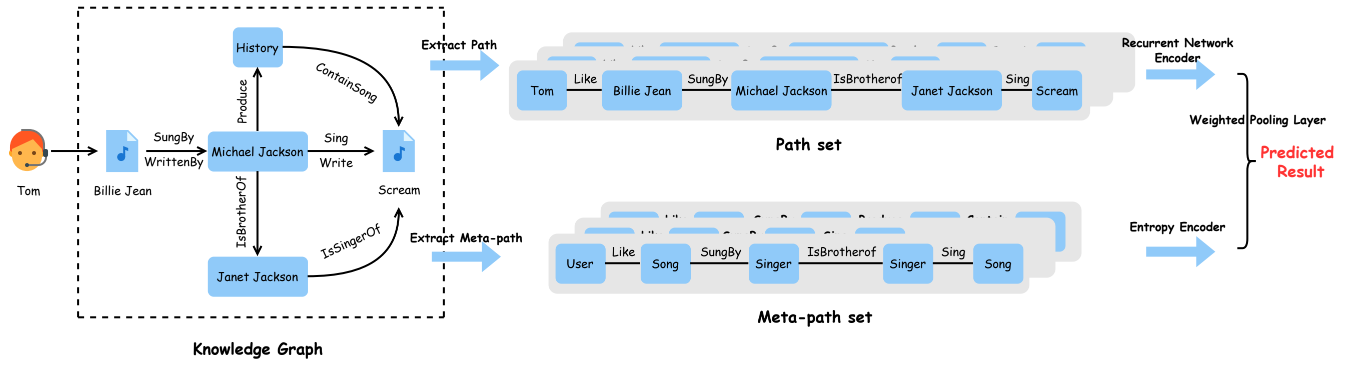


Figure 1: The overall framework of the proposed model PeRN

considerably more research interests than traditional translation-based methods such as TransE [3], TransH [29] and *collaborative knowledge graph embedding* method (CKE) [33].

However, current path-based methods exist a critical defect, that is, they neglect the difference between paths and negatively impact the accuracy and explainability of recommendations. Take the toy KG in Figure 1 for instance, the potential result of "If Tom likes Scream" can be reasoned by the following path set:

- p_1 = Tom$\xrightarrow{Like}$Billie Jean$\xrightarrow{SungBy}$Michael Jackson $\xrightarrow{Produce}$History$\xrightarrow{ContainSong}$Scream;
- p_2 = Tom$\xrightarrow{Like}$Billie Jean$\xrightarrow{SungBy}$Michael Jackson $\xrightarrow{Sing}$Scream;
- p_3 = Tom$\xrightarrow{Like}$Billie Jean$\xrightarrow{SungBy}$Michael Jackson $\xrightarrow{IsBrotherOf}$Janet Jackson$\xrightarrow{Sing}$Scream.

Obviously, p_3 is less credible than p_1 and p_2 in explaining the result above — the act "user *like* song *sungby* singer *isbrotherof* singer *sing* item" is quite uncommon and unconvincing in music recommendation field. Indiscriminately processing the paths reduce information utilization, more seriously, can increase the cold-start costs. In *Knowledge-aware Path Recurrent Network* (KPRN) [28], the prior knowledge which is used to complete KG even accounts for **50%** of the original dataset, making it difficult to apply the model in industry.

To fill the research gap, we propose a new recommendation model named *Path-enhanced Recurrent Network* (PeRN), which extracts not only path information in KG, but also meta path set to differentiate the path contribution to one user-item pair. Figure 1 illustrates the overall framework of our proposed model. Inspired by previous work [34], we innovatively use meta path, a general conception in heterogeneous information network (HIN), to generalize the structure of complicated paths. Meta path is a path schema consisting of a series of entity-type data and relation data. We take two path cases for example: "Tom$\xrightarrow{Like}$Billie Jean$\xrightarrow{SungBy}$Michael Jackson$\xrightarrow{Sing}$Scream" and "Amy$\xrightarrow{Like}$Love Story$\xrightarrow{SungBy}$Taylor Swift$\xrightarrow{Sing}$Speak now". They can both be abstracted into the same meta path "User$\xrightarrow{Like}$Song$\xrightarrow{SungBy}$Singer$\xrightarrow{Sing}$Song". By doing this, all the paths in the dataset can be abstracted into several kinds of user habits, which will be calculated to credibility values in the entropy encoder in our model. Thus the confidence ratio between different paths in the same user-item pair can be obtained. As for path set, we adopt bidirectional long short-term memory (bi-LSTM) network and a two-layer fully connected neural network to compute sequential entity and relation vectors to a certain score. Afterwards a weighted pooling layer is performed to combine these path scores and credibility values from entropy encoder to a predicted result. To learn the parameters effectively, we propose to use logarithmic loss function along with ridge penalty, i.e. L_2 regularization, to train our model. Furthermore, we design bidirectional search method, which is also meta-path-aided, to enhance the efficiency of path extraction and make PeRN more doable. To validate the ability of PeRN in recommendation accuracy and solving cold-start issue, we conduct extensive experiments on two real-world dataset. Additionally, we visualize a user-item interaction example which is random chosen to illustrate the enhancement in explainability of our model.

The main contributions of this work are as threefold:

- We innovatively introduce meta path to general path-based methods and propose a novel end-to-end recurrent neural network model to enhance explainability and reduce cold-start costs of KG recommendation.
- We improve the path extraction method with a bidirectional strategy to efficiently extract path data from KG, which makes it possible to apply PeRN in real scenes.
- We have performed experiments on two real-world datasets from bi-classification and top-K perspectives, compared with several representative baselines, to highlight the importance of integrating KG into recommendation and verify the practicality of our proposed method.

The rest of this paper is arranged as follows. Section 2 gives a brief review on two kinds of related works. Then section 3 and section 4 explain the model PeRN in details. Experiment results for verifying this model are shown in section 5. At last, section 5 contains conclusions plus some ideas for further work.

2 RELATED WORK

This section provides an general summary of several state-of-the-art methods of integrating KG into recommendation, which can be mainly sorted into translation-based and path-based methods.

2.1 Translation-based Methods

Prior research has proposed various techniques [2, 3, 14, 16, 29] to embed KG into a low-dimensional vector space, which make KG computable. These methods can be roughly divided into two categories [25]: 1) Translational distance models, including translation-based and other distance models, and 2) Semantic matching models, including tensor-factorization-based and neural-network-based models. Among them, the translation-based models such as TransE [3] and its variants [13, 14, 29], using the idea of translation to transform entities and relations into vectors to embed KG, have been widely used in KG recommendation because of their simplicity and effectiveness. CKE [33] firstly adopts Bayesian TransR [14] to generate user latent vector and KG-aided item latent vector to collaboratively learn the predicted result. DKN [24], leveraging four different translation-based model [3, 13, 14, 29] to embed KG and enrich side information, also applies a attention-based deep convolutional neural network to news recommendation field. More recently, TUP [4] employs TransH [29] to predict user preference from KG for improving recommender system.

Such translation-based methods significantly augment the accuracy of results. However, they neglect to explore the correlations (i.e. multi-hop relations) between user-item pairs. In other words, these methods fail to explain why the user have interest in these items. Thus we argue that these methods lack explainability and reasoning ability in recommendation.

2.2 Path-based Methods

In the aspect of path-based methods, Yu et al. [32] integrates heterogeneous information network to matrix factorization for personalized entity recommendation. This is a new idea that not only firstly introduces user-item-path conception to recommender systems but also inspires other researchers start to apply path in KG for convinced recommendation. As a consequence, various path-based approaches [11, 28, 34] have sprung up nowadays to memorize sequential implicit information in KG. KSR [11] incorporates gated recurrent unit (GRU) and key-value memory network (KV-MN) to capture sequential user preference from knowledge base, while RKGE [21] leverage an recurrent network batch to embed path semantics to augment interpretability. In KPRN [28], a long short-term memory (LSTM) network is utilized to encode path and explore the connectivity between users and items. Also, EIUM [12] spreads this thought to multi-modal knowledge base and design a self-attention matrix to mine deep information from path.

Though these methods endow the explainability and reasoning ability to recommend systems, there is still a severe defect that is the underutilization of mining semantics in path. This flaw also causes low performance in dealing with cold-start issues: the prior knowledge of completing KG is extremely costly. Moreover, path extraction is also a time-consuming and labor-intensive step in path-based methods as the time complexity of algorithm grows exponentially with the length of path.

Table 1: Notations

Descriptions	Notations
User set	$\mathcal{U} = \{u_1, u_2, ...u_{\|\mathcal{U}\|}\}$
Item set	$\mathcal{I} = \{i_1, i_2, ...i_{\|\mathcal{I}\|}\}$
User-item interaction set	$\mathcal{A} = \{a_1, a_2, ...a_{\|\mathcal{A}\|}\}$
Entity set	$\mathcal{E} = \{e_1, e_2, ...e_{\|\mathcal{E}\|}\}$
Relation set	$\mathcal{R} = \{r_1, r_2, ...r_{\|\mathcal{R}\|}\}$
Knowledge graph	$\mathcal{KG} = \{(h, r, t)\|h, t \in \mathcal{E}, r \in \mathcal{R}\}$
Schema graph	$\mathcal{G} = \mathbb{R}^{g \times g}, g = \|\text{typeof}(\mathcal{E})\|$
Interaction a_k in KG	$a_k = (u_m, i_n), u_m \in \mathcal{E}, i_n \in \mathcal{R}$
A path p between a_k	$p = u_m \xrightarrow{r_1} e_1 \xrightarrow{r_2} e_2, ... \xrightarrow{r_l} i_n$
Meta path mp between a_k	$mp = \text{typeof}(u_m) \xrightarrow{r_1}, ...\text{typeof}(i_n)$
Paths between a_k	$P_k = \{p_1, p_2, ..., p_{\|P_k\|}\}$
Meta paths between a_k	$MP_k = \{mp_1, mp_2, ...mp_{\|MP_k\|}\}$
Path set	$\mathcal{P} = \{P_1, P_2, ...P_{\|\mathcal{A}\|}\}$
Meta path set	$\mathcal{M} = \{mp_1, mp_2, ..., mp_{\|\mathcal{M}\|}\}$
Hidden state vectors	$\overrightarrow{h}_{l+1}, \overleftarrow{h}_{l+1}, h$
Weight matrix	$W_f, W_i, W_C, W_o, W_1, W_2$
Sigmoid function	σ
Loss function	$\mathcal{L}$

3 PROBLEM FORMULATION

Before elaborating our model, we formally define the notations used throughout this paper in Table 1. Same as other recommender systems, we let $\mathcal{U} = \{u_1, u_2, ..., u_{|\mathcal{U}|}\}$ and $\mathcal{I} = \{i_1, i_2, ..., i_{|\mathcal{I}|}\}$ denote user set and item set respectively. $\mathcal{A} = \{(u, i)|u \in \mathcal{U}, i \in \mathcal{I}\} = \{a_1, a_2, ..., a_{|\mathcal{A}|}\}$ represents all the interactions between users and items in our dataset.

Definition 3.1. **Knowledge Graph.** As entity set $\mathcal{E}$ and relation set $\mathcal{R}$ have been denoted, knowledge graph (KG) can be defined as $\mathcal{KG} = \{(h, r, t)|h, t \in \mathcal{E}, r \in \mathcal{R}\}$ where (h, r, t) is a triple combining head entity h and tail entity t by relation r. Here, every user and item in $\mathcal{U}$ and $\mathcal{I}$ could be searched as an entity in $\mathcal{KG}$, which makes path extraction between user-item interactions possible. By abstracting entities in $\mathcal{E}$ to entity types (shown as function typeof() in table above), the schema of $\mathcal{KG}$ can be revealed along with relations, which is denoted as a two dimensional matrix $\mathcal{G}$.

Definition 3.2. **Path** and **Meta Path.** Given an interaction $a_k = (u_m, i_n)$, a sequence of triples that connect user u_m and item i_n can be found within $\mathcal{KG}$ as $\{(u_m, r_1, e_1), (e_1, r_2, e_2), ..., (e_{l-1}, r_l, i_n)\}$, which we record as a path: $p = u_m \xrightarrow{r_1} e_1 \xrightarrow{r_2} e_2 ... \xrightarrow{r_l} i_n$. Therefore, all the qualified paths of one interaction a_k and interaction set $\mathcal{A}$ are denoted as $P_k = \{p_1, p_2, ..., p_{|P_k|}\}$ and $\mathcal{P} = \{P_1, P_2, ...P_{|\mathcal{A}|}\}$ individually. As each path owns a meta path by abstracting its entities to entity types, we denote mp as the meta path of path p and $MP_k = \{mp_1, mp_2, ..., mp_{|MP_k|}\}$ as the meta path set of path set P_k (also of interaction a_k). Meta path of all interactions, denoted as set $\mathcal{M}$, can be obtained by certain traversal of schema graph $\mathcal{G}$.

Besides, there are three points to be emphasized: 1) $\mathcal{U}, \mathcal{I} \subsetneq \mathcal{E}$ and $\mathcal{U} \cap \mathcal{I} = \varnothing$. 2) MP_k and P_k are both generated from a_k, so

$|MP_k| \leqslant |P_k|$, the equal sign holds when all path types in $|P_k|$ are different. 3) $\mathcal{M} = MP_1 \cup MP_2 \cup ... \cup MP_{|\mathcal{A}|}$.

Task Definition: With the given user-item interaction a_k and its path set P_k, the goal of PeRN is formulated as follows:

$$\hat{y}_k = f_\Delta(P_k) \tag{1}$$

where $\hat{y}_k$ is the predicted score of interaction a_k and f denotes the function of PeRN with parameters Δ.

4 PATH-ENHANCED RECURRENT NETWORK

In this section, we give a thorough description and elaboration of our proposed PeRN for incorporating KG to recommendation. So as to expound PeRN more clearly, we divide this section into 5 basic units: Firstly, we design a bidirectional path extraction algorithm to boost the efficiency of discovering path set between user and item entities in KG. Then, we adopt a Bi-LSTM network as a model to embed path set and memorize it as a set of predicted score. Furthermore, the entropy encoder is created to differentiate the contributions of paths in one path set by analyzing the information gain of their meta path. Finally, weighted pooling layer and optimization steps are used for jointly combining the scores and learning.

4.1 Bidirectional Path Extraction

KG generally contains millions of entities and relations, which indicates it is labor-intensive and time-consuming to find all paths between two entities. In addition, the difficulty of searching a path increases exponentially with its length that makes path extraction even harder.

Aiming to work out this issue, we design a meta-path-aided bidirectional path extraction algorithm to retrieve all qualified paths. As previous work [19] has discussed, we regard paths longer than six hops as noises. Thus this bidirectional scheme, described in Algorithm 1, can change the complexity of the search step from a maximum of six hops to a maximum of three hops. In preliminaries, we firstly abstract KG to its schema graph by modifying entities in triples to entity types. By removing all duplicates in these processed triples, the schema graph can be displayed via a matrix, of which every side is a list of all entity types. And elements in matrix can store relation type information between entity types. Accordingly, the whole meta path set $\mathcal{M}$ can be retrieved by traversal of this directed graph. Given a knowledge graph $\mathcal{KG}$, user-item interaction set $\mathcal{A}$ and meta path set $\mathcal{M}$, algorithm 1 can help us find a whole path set $\mathcal{P}$. After the initialization of $\mathcal{P}$, for each interaction a_k, we adopt depth-first-search (DFS) idea to retrieve candidate paths (lines 4-5) and meta-path-aided idea (lines 6-20) to choose target paths bidirectionally. As a matter of fact, candidate sets S_1 and S_2 can be used in parallel for different meta path so that we significantly enhanced the efficiency of path extraction by designing multi-threaded program in reality.

4.2 Recurrent Network Encoder

With the booming development of deep learning models, recurrent Neural Network(RNN) has become increasingly more widely used in processing sequence data like path information [21, 28].

Algorithm 1: Bidirectional Path Extraction

```
Input: Knowledge graph $\mathcal{KG} = \{(h,r,t)|h,t \in \mathcal{E}, r \in \mathcal{R}\}$,
       user-item interaction set $\mathcal{A} = \{a_1, a_2, ...a_{|\mathcal{A}|}\}$,
       meta path set $\mathcal{M} = \{mp_1, mp_2, ..., mp_{|\mathcal{M}|}\}$.
Output: Path set $\mathcal{P} = \{P_1, P_2, ..., P_{|\mathcal{A}|}\}$.
1  Initialize: $\mathcal{P} \leftarrow \varnothing$ ;
2  for each $a_k = (u_m, i_n)$ in $\mathcal{A}$ do
3      $P_k \leftarrow \varnothing$;
4      $S_1 \leftarrow$ retrieve all paths by head = $u_m$ within 3 hops;
5      $S_2 \leftarrow$ retrieve all paths by head = $i_n$ within 3 hops;
6      for each meta path $mp$ in $\mathcal{M}$ do
7          $l \leftarrow$ length of $mp$;
8          $P_1 \leftarrow \varnothing$ // left sub path set;
9          $P_2 \leftarrow \varnothing$ // right sub path set;
10         for each $p_l$ in $S_1$ do
11             if $p_l$ satisfies $mp[0 \rightarrow 2 * \lfloor l/2 \rfloor]$ then
12                 Add $p_l$ to $P_1$;
13         for each $p_r$ in $S_2$ do
14             if $p_r$ satisfies $mp[2 * \lfloor l/2 \rfloor \rightarrow 2 * l]$ then
15                 Add $p_r$ to $P_2$;
16         for each $p_l, p_r$ in $P_1, P_2$ do
17             if $p_l[tail] = p_r[tail]$ then
18                 $p \leftarrow$ combine $p_l$ and reverse($p_r$);
19                 Add $p$ to $P_k$;
20     Add $P_k$ to $\mathcal{P}$;
21 return $\mathcal{P}$
```

In our model PeRN, as relations in paths are not always in one direction like (Michael Jackson, Cooperate_with, Janet Jackson) and (Janet Jackson, Cooperate_with, Michael Jackson), we choose a bidirectional long short-term memory (bi-LSTM) based model, illustrated in Figure 2, to better sequential information and output the predicted score of the path. In a word, here input a path p and output its predicted score s from this network.

A path here could be recorded as a sequence of entities and relations like $p = [u_m, r_1, e_1, r_2, e_2..., r_l, i_n]$ as well as the number of relations less than or equal to six. In order to embed this multi-hop data to a series of vector, we firstly transform this l-hop path to $l+1$ consecutive vectors like $(u_m, \text{typeof}(u_m), r_1)$, $(e_1, \text{typeof}(e_1), r_2)$, $...(e_{l-1}, \text{typeof}(e_{l-1}), r_l)$, $(i_n, \text{typeof}(i_n), null)$. Then, map entity index which is a number from 0 to millions to a $d-2$ dimension vector with numbers close in size. After concatenating entity vector with entity type index and relation index, here we can get a d dimension vector represent a hop of the path. The q-th vector in path can be defined via the following equation:

$$\alpha_q = \text{Map}(e_{q-1}) \oplus e'_{q-1} \oplus r_q \tag{2}$$

where $\oplus$ is the operation of concatenation and e' is the type of entity e. After this step, path p is represented as a vector set $\{\alpha_1, \alpha_2, ..., \alpha_{l+1}\}$, which could be the input of this model. With a strong ability to memorize sequential semantics [15], LSTM [9] is chosen to be the main body of recurrent network encoder, and the

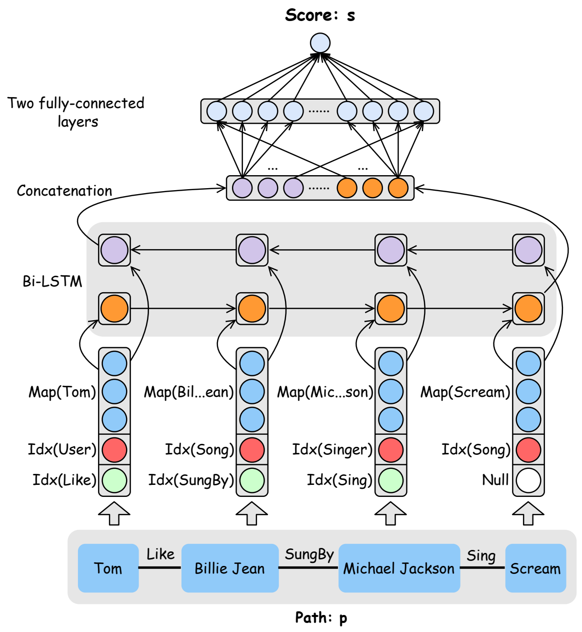


Figure 2: The architecture of recurrent network encoder in PeRN. Here entity type song is the item in recommendation issue.

q-th vector α_q of input vector set is computed as follows:

$$\begin{aligned} f_q &= \sigma(W_f \dot{(} h_{q-1} \oplus \alpha_q) + b_f) \\ i_q &= \sigma(W_i \dot{(} h_{q-1} \oplus \alpha_q) + b_i) \\ \widetilde{C}_q &= \tanh(W_C \dot{(} h_{q-1} \oplus \alpha_q) + b_C) \\ C_q &= C_{q-1} \odot f_q + \widetilde{C}_q \odot i_q \\ o_q &= \sigma(W_o \dot{(} h_{q-1} \oplus \alpha_q) + b_o) \\ h_q &= o_q \odot \tanh(C_q) \end{aligned} \tag{3}$$

where f_q, i_q and $o_q \in \mathbb{R}^{d'}$ denote forget, input and output gate; $\widetilde{C}_q$ and $C_q \in \mathbb{R}^{d'}$ denote represent candidate values vector and memory state vector; W_f, W_i, W_C and $W_o \in \mathbb{R}^{d' \times (d'+d)}$ are the weight matrices initialized with a random value, while b_f, b_i, b_C and b_o are bias vectors of each gate or cell; σ and tanh denote sigmoid and hyperbolic tangent activation function; $\odot$ and $\oplus$ represent hadamard product and concatenation respectively. Consequently, the hidden state vector $h_q \in \mathbb{R}^{d'}$ of q-th step is obtained by last hidden state h_{q-1} and given α_q. After $l+1$ iterations, the last hidden state vector h_{l+1} can be considered to hold the sequential path information. To explain more precisely, we simplify this process into the following equation:

$$h_{l+1} = \text{LSTM}([\alpha_1, \alpha_2, ..., \alpha_{l+1}]) \tag{4}$$

As shown in Figure 2, our model PeRN takes bi-LSTM into consideration which adopts a forward LSTM and a backward LSTM and then concatenate its bi-directional results to remember path information more firmly. This step can be formulated in the equations below:

$$\begin{aligned} \overrightarrow{h}_{l+1} &= \text{LSTM}([\alpha_1, \alpha_2, ..., \alpha_{l+1}]) \\ \overleftarrow{h}_{l+1} &= \text{LSTM}([\alpha_{l+1}, ..., \alpha_2, \alpha_1]) \\ h &= \overrightarrow{h}_{l+1} \oplus \overleftarrow{h}_{l+1} \end{aligned} \tag{5}$$

After embedding the path p into a representative vector h, the final step of recurrent network encoder is to convert it to a predicted score by establishing a simple neural network with two fully-connected layers. So the score s of path can be calculated in the following equation:

$$s = W_2^{\mathrm{T}} \text{ReLU}(W_1^{\mathrm{T}} h) \tag{6}$$

here W_1^{T} and W_2^{T} mean the coefficient weights of layer 1 and layer 2 respectively and we adopt rectified linear unit (ReLU) as activation function in each neuron with omitted bias. In fact, score s is a 1×1 dimension matrix, and we directly treat it as a scalar for simplicity.

4.3 Entropy Encoder

Given an interaction $a_k = (u_m, i_n)$ and its extracted path set $P_k = \{p_1, p_2, ..., p_{|P_k|}\}$, the scores of each path could be stored in a set $S_k = \{s_1, s_2, ..., s_{|P_k|}\}$. By abstracting entity instance to entity type in path, the meta path set of a_k could be denoted as $MP_k = \{mp_1, mp_2, ..., mp_{|MP_k|}\}$, where $|MP_k| \leqslant |P_k|$ (*cf.* Section 3). Clearly, it is irrational to predict a_k by weighted average their path scores (*cf.* Section 1). To tackle this issue and increase the explainability, we design a information entropy based method.

Inspired by its applications and enhancements in computer vision field recently [18, 22], we design an entropy-based weighting encoder to differentiate path contributions by computing its information gain to specific interaction. In practice, all the interactions are divided by positive feedback and negative feedback which indicates each path p in its path set P_k shares the same target 1 or 0. Here we collectively define path and meta path with target 1 as confidence path (CP), and otherwise non-confidence path (NP). By traversing and counting all path p in the whole path set $\mathcal{P}$, it is attainable to obtain the frequency of "p is CP" considering it is a binary classification problem. So the information entropy of event "if p is CP" (denote as event D which owns values 0 and 1) is defined as following:

$$\text{Ent(D)} = -\sum_{j \in 0,1} \text{P(D}{=}j) \log_2 \text{P(D}{=}j) \tag{7}$$

Although there are millions of paths in $\mathcal{P}$, the numbers of meta path are limited, which we denote as integer m. So we can endow a path m boolean features to determine its type of meta path and further judge its contribution to event D. Similar as Equation(5), for feature $E_g (g \in [1, m])$, we can find its conditional entropy for event D as following:

$$\text{Ent(D}|E_g) = \sum_{i \in 0,1} \text{P}(E_g{=}i)\text{Ent(D}|E_g{=}i) \tag{8}$$

where Ent(D$|E_g{=}i$) is conditional entropy by fixing $E_g{=}i$ and written as the equation below:

$$\text{Ent(D}|E_g{=}i) = -\sum_{j \in 0,1} \text{P(D}{=}j|E_g{=}i) \log_2 \text{P(D}{=}j|E_g{=}i) \tag{9}$$

In this way, the information gain of event D by given feature E_g could be obtained:

$$\text{Gain}(\text{D},E_g) = \text{Ent}(\text{D}) - \text{Ent}(\text{D}|E_g) \tag{10}$$

For currently given interaction a_k and its meta path set MP_k, each $mp_i (i \in [1, |MP_k|])$ could find its unique corresponding information gain by matching E_g=mp_i. Here we normalize the information gain of each meta path to get their weights:

$$w_i = \frac{\text{Gain}(\text{D},E_g{=}mp_i)}{\Sigma_{j=1}^{|MP_k|}\text{Gain}(\text{D},E_g{=}mp_j)} \tag{11}$$

where w_i is the weight of i-th path p_i in path set of a_k and can further differentiate the score s_i in S_k. And all weights of the same interaction a_k will be put in a weight set W_k for later step.

4.4 Weighted Pooling Layer

After getting a score set $S_k = \{s_1, s_2, ..., s_{|P_k|}\}$ and its corresponding weight set $W_k = \{w_1, w_2, ..., w_{|P_k|}\}$, we adopt a weighted pooling layer combining them to attain the final predictive score, which is formulated as:

$$\hat{y}_k = \sigma(\sum_{i=1}^{|P_k|} w_i s_i) \tag{12}$$

where σ is a sigmoid activation function to map final score into a range of 0 to 1.

4.5 Optimization

Since all the interactions in $\mathcal{A}$ can be observed as positive feedback and negative feedback (*cf.* Section 4.3), we regard our recommendation task as a binary classification issue with 0 for negative and 1 for positive, similar as previous work [28]. We adopt cross entropy loss to optimize our result. With the given interaction a, our loss function is defined as follows:

$$\mathcal{L} = -\sum_{a \in \mathcal{A}} (y \log \hat{y} + (1-y)\log(1-\hat{y})) \tag{13}$$

where a is all interactions in $\mathcal{A}$, y and $\hat{y}$ is the observed feedback and predictive score of a. We also conduct L_2 regularization on the parameters of our PeRN, which is dropped here for simplicity.

5 EXPERIMENTS AND ANALYSIS

In this section, we perform various experiments on 2 real-world recommendation datasets, which are music recommendation and movie recommendation scenarios, to evaluate our PeRN with several state-of-the-art baselines.

5.1 Dataset Description

To more realistically construct KG for recommendation and mine user preference by path, we here adopt two benchmark datasets: 1) KKBox[1], which is a music domain recommendation dataset from WSDM cup Challenge 2018 and provided by KKBox Music Streaming Service. 2) IM-1M, composed of IMDb[2] and MovieLens-1M[3] datasets, is a common movie recommendation dataset that has been widely used in KG-enhanced recommendation task recently [21, 28]. The statistics of KKBox and IM-1M is shown in Table 1 below.

[1]https://www.kaggle.com/c/kkbox-music-recommendation-challenge
[2]https://www.kaggle.com/suchitgupta60/imdb-data
[3]https://grouplens.org/datasets/movielens/

Table 2: Statistics of KKBox and IM-1M

	Dataset	KKBox	IM-1M
User-Item Interaction	#Users	34,403	6,040
	#Items	2,296,320	3,274
	#Interactions	3,696,465	370,023
	Data Density	0.0047%	1.87%
Knowledge Graph	#Entities	2,562,937	15,439
	#Entities Types	5	5
	#Relation Types	8	9
	#Triples	16,237,068	442,409
Path	#Path	41,400,408	345,344
	Avg.Path.Length	5.11	4.74
	#Meta Path Types	21	46
	Avg.Meta.Path.Length	5	5.37

The fundamental section of KG-enhanced recommendation dataset is to construct domain KG. Firstly, the basic part of KG can be selectively generated from background information in original dataset. Then, in order to integrate user set to KG, which makes finding paths between users and items possible, a group of interactions should be split to KG as compensations. KKBox not only provides a huge amount of user-item interaction data with positive feedback 1 and negative feedback 0, but also several side-information like artist, lyricist, composer, genre, which indicates this music domain KG can be straightly constructed. In IM-1M, IMDb (Internet Movie Database) contains comprehensive auxiliary information such as core members, duration, genre and budget of the movie, while MovieLens-1M provides more than 1,000,000 user-item interactions with rating scores {1, 2, 3, 4, 5}. Thus Movie domain KG can be constructed by mapping movie titles of IMDb and MovieLens-1M. To better fit our proposed model, all the interactions of which rating score equals 3 were omitted here for bi-classification and the other 4 scores were normalized for top-K task. In processed KKBox and IM-1M, about 50% of original interactions will be treated as valid interactions (exact numbers are shown in Table 2). The other part, which is used for completing KG, can measure the severity of cold-start issue by **P**ercentage of interactions used to **c**omplete **KG** (**PcKG** for short). In terms of path data, we give our definition of KG at first: the schema graphs of music domain KG and movid domain KG that we designed are illustrated in Figure 3. In KKBox we select user, item(song) , language, artist and genre as target entity types. Relation "CooperatesWith" is extracted by the condition of "multiple artists in one song". In IM-1M, we choose user, item(movie), director, actor and genre as target entity types. Also, there are "actor_1_name" and "actor_2_name" attributes in same movie, of which we regard as relation "Co-starsWith". Meta

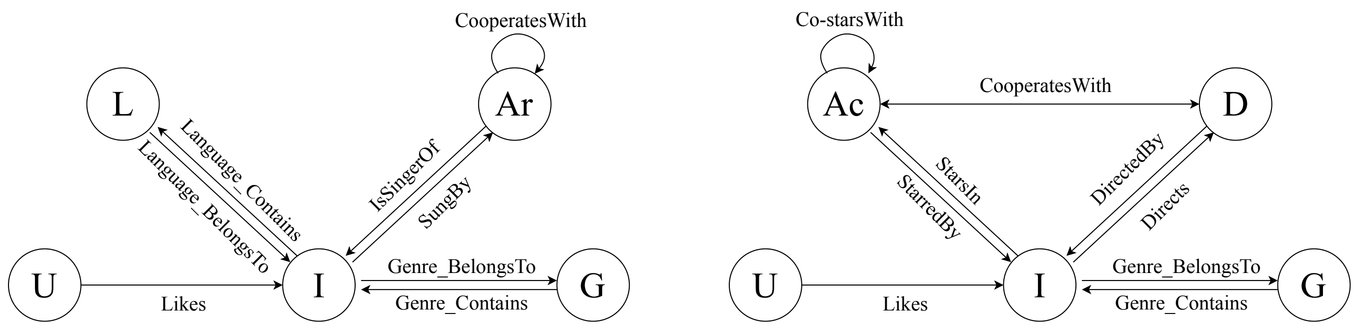


Figure 3: Schema graphs of KKBox (left) and IM-1M (right). In KKBox, U: user, I: item (song), L: language, Ar: artist, G: genre. In IM-1M, U: user, I: item (movie), Ac: actor, D: director, G: genre.

paths can be extracted by certain traversal in schema graph, and paths are extracted by Algorithm 1.

5.2 Experimental Settings

Evaluation Metrics

Aiming at evaluating our PeRN more comprehensively and demonstrate its rationality, we use evaluation metrics from the perspectives of binary classification recommendation task, top-K recommendation task and ability to handle cold-start issue, which are shown as below:

- Precision (**P**), recall (**R**), F_1**-score**, mean squared error (**MSE**), area under curve (**AUC**): This five metrics are adopt to represent an overall accuracy of bi-classfication task. Among them, precision and recall are commonly used to solve the imbalance of positive and negative samples. F_1-score is the harmonic mean of precision and recall. MSE is adopted here to display the actual prediction error of the model. AUC, of which the threshold we set is in range of 0.1 to 0.9, is calculated as following equation:

$$\text{AUC} = \frac{\sum_{i \in positive}(P + N + 1 - rank_i) - \frac{P(1+P)}{2}}{P \times N} \tag{14}$$

  where P and N are numbers of positive and negative interactions, $rank_i$ denotes the ranking of i's predicted score.
- Normalized discounted cumulative gain (**NDCG@K**): As the most common metrics in top-k recommendation task, NDCG@K evaluates the model performance by position influence, which is calculated as following:

$$\text{NDCG@K} = \frac{1}{\alpha}\text{DCG@K} = \frac{1}{\alpha}\sum_{i=1}^{K}\frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{15}$$

  where rel_i indicates the relevance of the recommendation at position i, K is the size of target list, α is a standardization constant that is the maximum value of DCG@K. Considering sizes of these two datasets are different, we set K in KKBox amd IM-1M as {3, 5, 10, 15} and {2, 6, 10, 12} separately.
- Percentage of interactions used to complete KG (**PcKG**): In measuring cold-start cost, we set PcKG in range of {10%, 20%, 30%, 40%, 50%}. Here the lower PcKG with higher score of other metrics indicates the higher ability in handling cold-start issue.

Baselines

We compared our PeRN with the following widely-used recommendation methods that are based on matrix factorization (MF), factorization machine (FM), KG translation, meta path in heterogeneous information network (HIN) and path in KG. The brief descriptions of these methods are as follows:

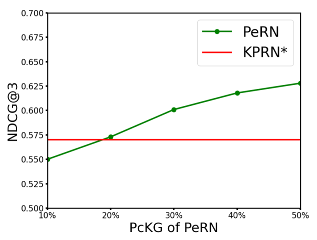
- MF [17]: This is a standard matrix factorization method with Bayesian personalized ranking (BRP) loss by regarding interactions as elements in user-item matrix to predict unknown interaction.
- AFM [30]: The method is a factorization model combined with attention mechanism which excavates potential relations and interactions among the user preferences and item features.
- RippleNet [23]: The key of this method is preference propagation which regards historical interests of users as a seed set and propagates user preferences in KG which contains extra information.
- MEIRec [7]: This is a meta-path-guided method which contains rich user-item information and interactions based on HIN and utilizes structural information fully for Intent Recommendation.
- KPRN [28]: KPRN is a representative method for integrating path into KG recommendations by recurrent neural network. Though KPRN achieves great performance in both binary classification and top-K recommendation issue, it has a server cold-start issue that its PcKG is up to **50**%.

Parameter Settings

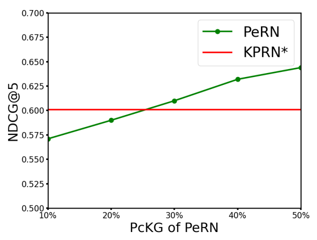
We omit any pre-trained parameters in our proposed model for pair comparison with several baselines. To reasonably embed path into a vector space, we map large fluctuated entity index to a 62 dimension vector and concatenate it with entity type index and relation type index. So size of each input vector of LSTM is 64 and with length up to 7 (i.e. 6-hops path). If the length of path is less than 7, then complete it with zero vector. And batch size is 256 here. We adopt stochastic gradient descent (SGD) in optimization step with the learning rate in range of {0.001, 0.01, 0.1, 0.5}, while

Table 3: Summary of performance on binary classification recommendation task between all baselines and our proposed PeRN on KKBox and IM-1M datasetS. Bolded numbers indicate the best result of each columns, and '*' indicates the arithmetic square root operation is performed on MSE for simplicity.

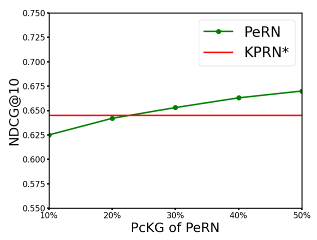
Dataset	KKBox					IM-1M				
Metrics	P	R	F_1	MSE*	AUC	P	R	F_1	MSE*	AUC
MF	0.509	0.528	0.518	0.496	0.511	0.612	0.608	0.610	0.431	0.586
AFM	0.517	0.533	0.525	0.483	0.536	0.647	0.632	0.639	0.413	0.601
RippleNet	0.699	0.732	0.715	0.287	0.762	0.742	0.713	0.727	0.284	0.694
MEIRec	0.753	0.774	0.763	0.242	0.819	0.792	0.804	0.798	0.221	0.734
KPRN	0.805	0.822	0.813	0.206	0.834	**0.843**	0.826	0.834	0.172	0.812
PeRN	**0.842**	**0.861**	**0.851**	**0.195**	**0.866**	0.835	**0.871**	**0.853**	**0.154**	**0.851**



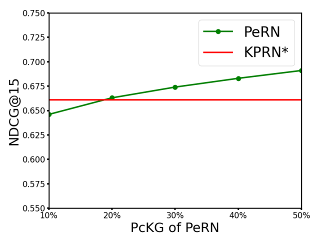
(a) KKBox NDCG@3



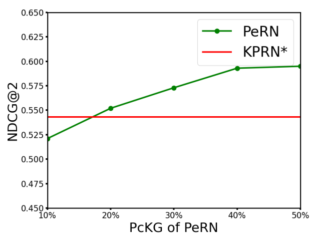
(b) KKBox NDCG@5



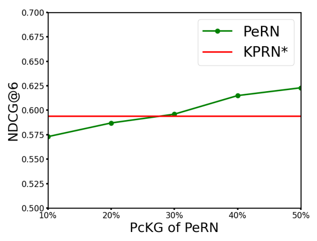
(c) KKBox NDCG@10



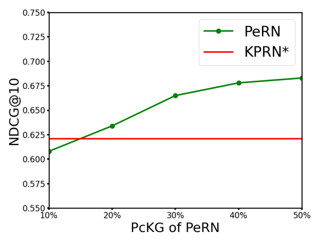
(d) KKBox NDCG@15



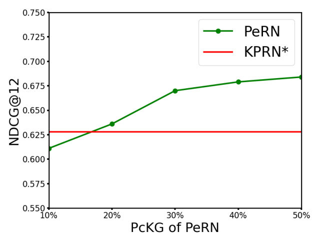
(e) IM-1M NDCG@2



(f) IM-1M NDCG@6



(g) IM-1M NDCG@10



(h) IM-1M NDCG@12

Figure 4: Performance of PeRN in top-K task and cold-start costs, measured by NDCG@{3, 5, 10, 15} in KKBox and NDCG@{2, 6, 10, 12} in IM-1M. '*' here indicates the PcKG of KPRN is constantly 50%.

L_2 regularization coefficients are tuned between $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}\}$.

5.3 Performance Comparison

Table 2 and Figure 4 reports our experiment results on binary classification recommendation and the ability in solving cold-start issue of top-K recommendation task respectively. In-depth experimental analyses are as follows:

Bi-classification Task:

In binary classification recommendation issue, the main challenge is to achieve considerable performances on two user-item interaction matrices. In KKBox, data density is only **0.0047%**, which is extremely sparse and renders various traditional recommendation method unusable. MF and AFM are recommendation methods based on matrix operations and have obtained ridiculous results when processing highly sparse KKBox. RippleNet achieves significant improvements than MF and AFM by integrating KG into user-item interactions, but its predictive results are not so appreciable as the path-based MEIRec and KPRN considering the high sparseness when embedding KG. As for our proposed PeRN, substantially outperforms the state-of-the-art methods and show best performance in several binary classification recommendation evaluation metrics. In IM-1M, as the data density is 1.87% here, PeRN and several baselines have achieved a slightly better results in P, R F_1 and MSE. In addition, original rating scores in IM-1M are displayed in range of normalized {1, 2, 4, 5} rather than 1 for positive and 0 for negative, so the actual classification effects are not so excellent as expected, which is shown by a lower AUC score than KKBox. Nonetheless, our proposed PeRN still owns more promising results compared with other baselines.

Top-K Task and Cold-start Costs:

As Figure 4 illustrates, our proposed PeRN also has a well performance in alleviating cold-start issue in top-K recommendation task compared with state-of-the-art path-based method KPRN. Here KPRN utilizes 50% of given target user-item interactions in both IM-1M and KKBox datasets in the process of constructing KG, which causes a severe cold-start issue. To demonstrate the ability of PeRN to handle cold-start issue in top-K task, we adopt PcKG (percentage of interactions used to complete KG), which is shown as a percentage, to evaluate our model against baseline. Specifically, we use PcKG = {10%, 20%, 30%, 40%, 50%} of interactions by each user in original dataset , which aims to ensure all the users could be found as an entity in KG, to complete music domain and movie domain KG. Every new completion has to re-extract all the path in order to confirm the accuracy of experiment. Measured by NDCG@K, where K = {3, 5, 10, 15} in KKBox and K = {2, 6, 10, 12} in IM-1M considering the size of paths is different within each datasets, our PeRN is able to achieve much more encouraging performance. In KKBox, with only 20%-30% PcKG can reach a higher accuracy in top-K recommendation task than KPRN with 50% PcKG, while this number normally decreases to 10%-20% in IM-1M. In other word, we use only 30% prior knowledge on average and attain a better prediction result in top-K recommendation task. Nevertheless, we have to admit that these results are supposed to increase more with a more thorough KG to be designed and established. We hold the strong belief that reducing and refining prior knowledge is the key to decrease cold-start costs.

5.4 Explainability Analysis

In terms of explainability, we randomly select a user-item interaction from IM-1M dataset and visualize its three paths, which is shown as Figure 5, to illustrate model's enhanced reasoning ability in a real scene. The IDs of user and target item are 3408 and 318 respectively. Red numbers denote the weighting score of each path. Some observations and analyses about "why user_3408 likes the movie The ShawShank Redemption" are presented as below.

As Figure 5 shows, three different paths between (U_id: 3408, I_id: 318) are clearly found, which can explain the question "why user_3408 likes item_318" at a shallow level.

In order to mine more in-depth information from paths and further differentiate the contribution of each path, we transform these three paths to meta paths by abstracting entity to entity type:

- $mp_1 = \text{User}\xrightarrow{Likes}\text{Item}\xrightarrow{BelongsTo}\text{Genre}\xrightarrow{Contains}\text{Item}$;
- $mp_2 = \text{User}\xrightarrow{Likes}\text{Item}\xrightarrow{StarredBy}\text{Actor}\xrightarrow{StarsIn}\text{Item}$;
- $mp_3 = \text{User}\xrightarrow{Likes}\text{Item}\xrightarrow{StarredBy}\text{Actor}\xrightarrow{Co-starsWith}$ $\text{Actor}\xrightarrow{StarsIn}\text{Item}$;

Evidently, these meta paths account for different user habits in movie recommendation field. mp_1 indicates that user tend to show more interests to movies with the same genre. mp_2 tells us that user may like the movies acted by the same actor. More interestingly, we

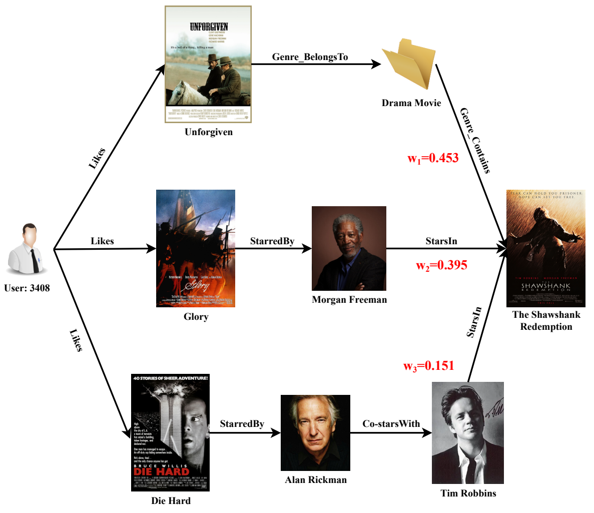


Figure 5: Visualization of paths between selected user-item interaction.

also can get a conclusion that user_3408 may be a fan of Morgan Freeman by simple analysis of mp_2. Besides, mp_3 reveals that user might have some interests in the cooperation between two famous actors. Such correlations between entity types assist us acquire a more precise user preference.

After an overall statistical analysis by entropy encoder and a learning process of our model, the weighting scores of three paths are shown in Figure 5. Specifically, score of mp_1 is a bit higher than mp_2, while score of mp_3 is much lower than the other two. These results unambiguously explain that the events "user likes the movie by same actor" and "user likes the movie of the same genre" are more common in movie recommendation. Also, the low weight of w_3 demonstrates that the behaviour "User Likes Item StarredBy Actor Co-starsWith Actor StarsIn Item" is pretty bizarre compared with the other two, which is consistent with our behavioral cognition in real life.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose a path-enhanced recurrent network (PeRN), which shows high performance in dealing with cold-start issues and enhances the explanation of KG recommendation. We innovatively exploit information from meta path to compensate for general path-based methods and introduce a bidirectional path extraction algorithm, since path extraction is time-consuming. Extensive experiments show the well-developed explanation and effectiveness of our method.

In future, we hope to further continue our work from following research directions: (1) We hope to improve current network model to boost its memory ability. In processing serialized data, such as path and meta path in our PeRN, the memory ability of model is just as crucial as its learning ability, which is always the most critical part in judging the quality of a model. Current researchers have paid too much attention to the RNN-based model, such as LSTM or GRU. We think memory network, such as key-value memory network, can also achieve amazing results and hope to leverage it in better memorizing path information in KG. (2) We hope to extract more information from path. Although we analyze the relationship between different paths by using entropy-based encoder in PeRN, we do think there are more interesting correlations between different path, such as triangular relationships, multiple relationships or relational reasoning in paths. We will mine more from this perspective to improve the utilization of path information and better excavate user preferences in KG recommendation. (3) In the past two years, researches and applications based on graph neural network (GNN), such as RecoGCN [31] and GSN [10], have been becoming a new focus. We plan to propose new GNN-based method for explainable recommendation with KG.

ACKNOWLEDGMENTS

REFERENCES

[1] 2019. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019.*

[2] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data - Application to word-sense disambiguation. *Machine Learning* 94, 2 (2014), 233–259.

[3] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *27th Annual Conference on Neural Information Processing Systems 2013.* 2787–2795.

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

[5] Xu Chen, Hongteng Xu, Yongfeng Zhang, Jiaxi Tang, Yixin Cao, Zheng Qin, and Hongyuan Zha. 2018. Sequential Recommendation with User Memory Networks. In *Proceedings of WSDM.* 108–116.

[6] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, Rohan Anil, Zakaria Haque, Lichan Hong, Vihan Jain, Xiaobing Liu, and Hemal Shah. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of RecSys.* 7–10.

[7] Shaohua Fan, Junxiong Zhu, Xiaotian Han, Chuan Shi, Linmei Hu, Biyu Ma, and Yongliang Li. 2019. Metapath-guided Heterogeneous Graph Neural Network for Intent Recommendation, See [1], 2478–2486.

[8] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of WWW.* 173–182.

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780.

[10] Wenpeng Hu, Zhangming Chan, Bing Liu, Dongyan Zhao, Jinwen Ma, and Rui Yan. 2019. GSN: A Graph-Structured Network for Multi-Party Dialogues. In *Proceedings of IJCAI*, Sarit Kraus (Ed.). ijcai.org, 5010–5016.

[11] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *Proceedings of SIGIR.* 505–514.

[12] 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 MM.* 548–556.

[13] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge Graph Embedding via Dynamic Mapping Matrix. In *Proceedings of ACL.* The Association for Computer Linguistics, 687–696.

[14] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *In Twenty-Ninth AAAI Conference on Artificial Intelligence.* 2181–2187.

[15] Yuanliang Meng, Anna Rumshisky, and Alexey Romanov. 2017. Temporal Information Extraction for Question Answering Using Syntactic Dependencies in an LSTM-based Architecture. In *Proceedings of EMNLP*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, 887–896.

[16] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of ICML.* 809–816.

[17] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of UAI.* 452–461.

[18] Proteek Chandan Roy and Vishnu Naresh Boddeti. 2019. Mitigating Information Leakage in Image Representations: A Maximum Entropy Approach. In *Proceedings of CVPR.* Computer Vision Foundation / IEEE, 2586–2594.

[19] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S. Yu, and Tianyi Wu. 2011. PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks. *PVLDB* 4, 11 (2011), 992–1003.

[20] Zhu Sun, Qing Guo, Jie Yang, Hui Fang, Guibing Guo, Jie Zhang, and Robin Burke. 2019. Research commentary on recommendations with side information: A survey and research directions. *Electron. Commer. Res. Appl.* 37 (2019).

[21] 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 RecSys.* 297–305.

[22] Weitao Wan, Jiansheng Chen, Tianpeng Li, Yiqing Huang, Jingqi Tian, Cheng Yu, and Youze Xue. 2019. Information Entropy Based Feature Pooling for Convolutional Neural Networks. In *Proceedings of ICCV.* IEEE, 3404–3413.

[23] 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 CIKM.* 417–426.

[24] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep Knowledge-Aware Network for News Recommendation. In *Proceedings of WWW.* 1835–1844.

[25] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge Graph Embedding: A Survey of Approaches and Applications. *IEEE Trans. Knowl. Data Eng.* 29, 12 (2017), 2724–2743.

[26] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation, See [1], 950–958.

[27] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural Graph Collaborative Filtering. In *Proceedings of SIGIR.* 165–174.

[28] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable Reasoning over Knowledge Graphs for Recommendation. In *In Thirty-Third AAAI Conference on Artificial Intelligence.* 5329–5336.

[29] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *In Twenty-Eighth AAAI Conference on Artificial Intelligence.* 1112–1119.

[30] Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. 2017. Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks. In *Proceedings of IJCAI.* 3119–3125.

[31] Fengli Xu, Jianxun Lian, Zhenyu Han, Yong Li, Yujian Xu, and Xing Xie. 2019. Relation-Aware Graph Convolutional Networks for Agent-Initiated Social E-Commerce Recommendation. In *Proceedings of CIKM*, Wenwu Zhu, Dacheng Tao, Xueqi Cheng, Peng Cui, Elke A. Rundensteiner, David Carmel, Qi He, and Jeffrey Xu Yu (Eds.). ACM, 529–538.

[32] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: a heterogeneous information network approach. In *Proceedings of WSDM.* 283–292.

[33] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative Knowledge Base Embedding for Recommender Systems. In *Proceedings of SIGKDD.* 353–362.

[34] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-Graph Based Recommendation Fusion over Heterogeneous Information Networks. In *Proceedings of SIGKDD.* 635–644.