Learning Intents behind Interactions with Knowledge Graph for Recommendation

Xiang Wang*
National University of Singapore
xiangwang@u.nus.edu

Tinglin Huang*
Zhejiang University
tinglin.huang@zju.edu.cn

Dingxian Wang*
eBay
diwang@ebay.com

Yancheng Yuan
The Hong Kong Polytechnic University
yanchengyuanmath@gmail.com

Zhenguang Liu
Zhejiang University
liuzhenguang2008@gmail.com

Xiangnan He†
University of Science and Technology of China
xiangnanhe@gmail.com

Tat-Seng Chua
National University of Singapore
dcscts@nus.edu.sg

ABSTRACT

Knowledge graph (KG) plays an increasingly important role in recommender systems. A recent technical trend is to develop end-to-end models founded on graph neural networks (GNNs). However, existing GNN-based models are coarse-grained in relational modeling, failing to (1) identify user-item relation at a fine-grained level of intents, and (2) exploit relation dependencies to preserve the semantics of long-range connectivity.

In this study, we explore intents behind a user-item interaction by using auxiliary item knowledge, and propose a new model, *Knowledge Graph-based Intent Network* (KGIN). Technically, we model each intent as an attentive combination of KG relations, encouraging the independence of different intents for better model capability and interpretability. Furthermore, we devise a new information aggregation scheme for GNN, which recursively integrates the relation sequences of long-range connectivity (*i.e.,* relational paths). This scheme allows us to distill useful information about user intents and encode them into the representations of users and items. Experimental results on three benchmark datasets show that, KGIN achieves significant improvements over the state-of-the-art methods like KGAT [41], KGNN-LS [38], and CKAN [47]. Further analyses show that KGIN offers interpretable explanations for predictions by identifying influential intents and relational paths. The implementations are available at https://github.com/huangtinglin/Knowledge_Graph_based_Intent_Network.

CCS CONCEPTS

• **Information systems → Recommender systems**.

*authors contribute equally.
†Xiangnan He is the corresponding author.

KEYWORDS

Recommendation, Knowledge Graph, Graph Neural Networks

ACM Reference Format:
Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, Tat-Seng Chua. 2021. Learning Intents behind Interactions with Knowledge Graph for Recommendation. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3442381.3450133

1 INTRODUCTION

Knowledge graph (KG) has shown great potential in improving the accuracy and explainability of recommendation. The rich entity and relation information in KG can supplement the relational modeling between users and items. They not only reveal various relatedness among items (*e.g.,* co-directed by a person), but also can be used to interpret user preference (*e.g.,* attributing a user's choice of a movie to its director).

Learning high-quality user and item representations from such structural knowledge has become the theme of knowledge-aware recommendation. Earlier works [1, 4, 51] generate embeddings from KG triplets and treat them as prior or content information to supplement item representations. Some follow-on studies [15, 44, 49] enrich the interactions with multi-hop paths from users to items for better characterizing user-item relations. However, these methods struggle to obtain high-quality paths, suffering from various issues like labor-intensive feature engineering [44], poor transferability to different domains [15, 17], and/or unstable performance [49]. More recently, a technical trend [38, 39, 41, 47] is to develop end-to-end models founded on graph neural networks (GNNs) [9, 13, 19, 34]. The key idea is to utilize the information aggregation scheme, which can effectively integrate multi-hop neighbors into representations. Benefiting from the integration of connectivity modeling and representation learning, these GNN-based models achieve promising performance for recommendation.

Despite their effectiveness, we argue that current GNN-based methods fall short in modeling two factors: (1) **User Intents**. To the best of our knowledge, none of these studies consider user-item relations at a finer-grained level of intents. An important fact

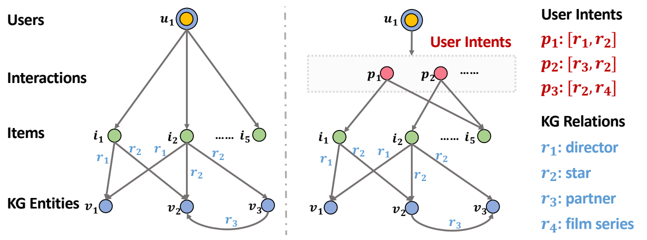


Figure 1: An example of user intents on adopting items (*i.e.,* fine-grained preference), where an arrow is the relation from a head entity to a tail entity. Best viewed in color.

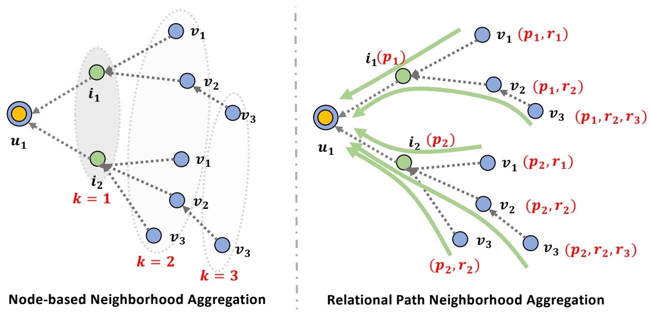

$k = 1$
$k = 2$
$k = 3$
Node-based Neighborhood Aggregation
Relational Path Neighborhood Aggregation

Figure 2: An example of the node-based and relational path-aware aggregation schemes, where a (dashed or solid) arrow is an information flow among nodes. Best viewed in color.

has been ignored: a user typically has multiple intents, driving the user to consume different items. Taking the right of Figure 1 as an example, intent p_1 emphasizes a combination of *director* (r_1) and *star* (r_2) aspects that drives user u_1 to watch the movies i_1 and i_5; while another intent p_2 highlights the *star* (r_2) and *partner* (r_3) aspects the user to select movie i_2. Ignoring the existence of user intents limits the modeling of user-item interactions. (2) **Relational Paths**. In these studies, the information aggregation schemes are mostly node-based — that is, to collect the information from neighboring nodes, without differentiating which paths it comes from. Moreover, KG relations are typically modeled in decay factors [38, 41] of adjacent matrix, in order to control the influences of neighbors. As shown in the left of Figure 2, the representation of u_1 mixes the signal from the one-, two-, and three-hop neighbors (*i.e.,* $\{i_1, i_2\}$, $\{v_1, v_2, v_3\}$, $\{v_3\}$, respectively). It fails to preserve the relation dependencies and sequencies carried by paths (*e.g.,* (p_1, r_2, r_3) in the three-hop path from v_3 to u_1). Hence, such node-based schemes are insufficient to capture the interactions among relations.

In this paper, we focus on exploring user intents behind user-item interactions by using item KG, so as to improve the performance and interpretability of recommendation. We propose a new model, *Knowledge Graph-based Intent Network* (KGIN), which consists of two components to solve the foregoing limitations correspondingly: (1) **User Intent Modeling**. Each user-item interaction is enriched with the underlying intents. While we can express these intents as latent vectors, their semantics are opaque to understand. Hence, we associate each intent with a distribution over KG relations, accounting for the importance of relation combination. Technically, the intent embedding is an attentive combination of relation embeddings, where important relations are assigned with larger attribution scores. Moreover, an independence constraint is introduced to encourage significant differences among intents for better interpretability. (2) **Relational Path-aware Aggregation**. Unlike the node-based aggregation mechanism, we view a relational path as an information channel, and embed each channel into a representation vector. As user-intent-item triplets and KG triplets are heterogeneous, we set different aggregation strategies for these two parts, so as to distill behavioral patterns of users and relatedness of items respectively. In a nutshell, this relational modeling allows us to identify influential intents, and encode relation dependencies and semantics of paths into the representations. We conduct extensive experiments on three real-world datasets. Experimental results show that our KGIN outperforms the state-of-the-art methods such as KGAT [41], KGNN-LS [38], and CKAN [47]. Furthermore, KGIN is able to interpret user behaviors at the granularity of intents.

We summarize the contributions of this work as:

- Revealing user intents behind the interactions in KG-based recommendation, for better model capacity and interpretability;
- Proposing a new model, KGIN, which considers user-item relationships at the finer granularity of intents and long-range semantics of relational paths under the GNN paradigm;
- Conducting empirical studies on three benchmark datasets to demonstrate the superiority of KGIN.

2 PROBLEM FORMULATION

We first introduce the structural data: user-item interactions and knowledge graph, and then formulate our task.

Interaction Data. Here we focus on the implicit feedback [26] in recommendation, where the signal that a user provides about her preference is implicit (*e.g.,* view, click, purchase). Let $\mathcal{U}$ be a set of users and $\mathcal{I}$ a set of items. Let $\mathcal{O}^+ = \{(u, i)|u \in \mathcal{U}, i \in \mathcal{I}\}$ be a set of observed feedback, where each (u, i) pair indicates that user u has interacted with item i before. In some previous works like KGAT [41], an additional relation *interact-with* is introduced to explicitly present the user-item relationship and convert a (u, i) pair to the $(u, \textit{interact-with}, i)$ triplet. As such, the user-item interactions can be seamlessly combined with KG.

Knowledge Graph. KG stores the structured information of real-world facts, such as item attributes, taxonomy, or external commonsense knowledge, in the form of a heterogeneous graph or heterogeneous information network [28, 29]. Let $\mathcal{V}$ be a set of real-world entities and $\mathcal{R}$ be the relation set, which involves relations in both canonical and inverse directions (*e.g.,* *director* and *directed-by*). Let $\mathcal{G} = \{(h, r, t)|h, t \in \mathcal{V}, r \in \mathcal{R}\}$ be a collection of triplets, where each (h, r, t) triplet indicates that a relation r exists from head entity h to tail entity t. For example, (*Martin Freeman, star, The Hobbit I*) describes that *Martin Freeman* is the *star* of movie *The Hobbit I*. With the mappings between items and KG entities ($\mathcal{I} \subset \mathcal{V}$), KG is able to profile items and offer complementary information to the interaction data.

Task Description. Given the interaction data $\mathcal{O}^+$ and the KG $\mathcal{G}$, our task of knowledge-aware recommendation is to learn a function that can predict how likely a user would adopt an item.

3 METHODOLOGY

We now present the proposed Knowledge Graph-based Intent Network (KGIN). Figure 3 displays the working flow of KGIN. It consists of two key components: (1) user intent modeling, which uses multiple latent intents to profile user-item relationships and formulates each intent as an attentive combination of KG relations, meanwhile encourages different intents to be independent of each others; and (2) relational path-aware aggregation, which highlights the relation dependence in long-range connectivity, so as to preserve the holistic semantics of relational paths. KGIN ultimately yields high-quality representations of users and items.

3.1 User Intent Modeling

Unlike the previous GNN-based studies [38, 41, 47] that assume no or only one *interact-with* relation between users and items, we aim to capture the intuition that behaviors of users are influenced by multiple intents. Here we frame the intent as the reason of users' choices to items, which reflects the commonality of all users' behaviors. Taking movie recommendation as an example, possible intents are diverse considerations on movie attributes, such as the combination of *star* and *partner*, or *director* and *genre*. Different intents abstract different behavioral patterns of users. This can supercharge the widely-used collaborative filtering [26] effect with the finer-grained assumption — users driven by similar intents would have similar preference on items. Such intuition motivates us to model user-item relations at the granularity of intents.

Assuming $\mathcal{P}$ as the set of intents shared by all users, we can slice a uniform user-item relation into the $|\mathcal{P}|$ intents, and decompose each (u, i) pair into $\{(u, p, i) | p \in \mathcal{P}\}$. As a result, we reorganize the user-item interaction data as a heterogeneous graph, termed **intent graph (IG)**, which differs from the homogeneous collaborative graph adopted in previous works [41, 47].

3.1.1 *Representation Learning of Intents.*

Although we can express these intents with latent vectors, it is hard to identify the semantics of each intent explicitly. One straightforward solution is to couple each intent with one KG relation, which is proposed by KTUP [4]. However, this solution only considers single relations in isolation, without accounting for the interactions and combinations of relations, thereby fails to refine high-level concepts of user intents. For example, the combination of relations r_1 and r_2 is influential to intent p_1, while relations r_3 and r_4 contributes more to intent p_2. Hence, we assign each intent $p \in \mathcal{P}$ with a distribution over KG relations — technically, exert an attention strategy to create the intent embedding as:

$$\mathbf{e}_p = \sum_{r \in \mathcal{R}} \alpha(r, p) \mathbf{e}_r, \tag{1}$$

where $\mathbf{e}_r$ is the ID embedding of relation r, which is assigned with an attention score $\alpha(r, p)$ to quantify its importance, formally:

$$\alpha(r, p) = \frac{\exp(w_{rp})}{\sum_{r' \in \mathcal{R}} \exp(w_{r'p})}, \tag{2}$$

where w_{rp} is a trainable weight specific to certain relation r and certain intent p. Here we use the weights for simplicity, and leave the further exploration of complex attention modules in future work. It is worthwhile mentioning that the attentions are not tailored to a single user, but refine common patterns of all users.

3.1.2 *Independence Modeling of Intents.*

Different intents should contain different information about user preference [23, 24]. If one intent can be inferred by the others, it might be redundant and less informative to describe user-item relation; in contrast, the intent with unique information will offer a useful angle to characterize behavioral patterns of users. Hence, for better model capacity and explainability, we encourage the representations of intents to differ from each others.

Here we introduce a module of independence modeling to guide the representation learning of independent intents. This module can be simply achieved by applying a statistical measure, such as mutual information [2], Pearson correlation [33], and distance correlation [32, 33, 43], as a regularizer. Here we offer two implementations:

- **Mutual information.** We minimize the mutual information between the representations of any two different intents, so as to quantify their independence. Such an idea coincides with contrastive learning [7, 12]. More formally, the independence modeling is:

$$\mathcal{L}_{\text{IND}} = \sum_{p \in \mathcal{P}} -\log \frac{\exp\left(s(\mathbf{e}_p, \mathbf{e}_p)/\tau\right)}{\sum_{p' \in \mathcal{P}} \exp\left(s(\mathbf{e}_p, \mathbf{e}_{p'})/\tau\right)}, \tag{3}$$

  where $s(\cdot)$ is the function measuring the associations of any two intent representations, which is set as cosine similarity function here; and τ is the hyper-parameter to the temperature in softmax function.
- **Distance correlation.** It measures both linear and nonlinear associations of any two variables, whose coefficient is zero if and only if these variables are independent. Minimizing the distance correlation of user intents enables us to reduce the dependence of different intents, which is formulated as:

$$\mathcal{L}_{\text{IND}} = \sum_{p, p' \in \mathcal{P},\ p \neq p'} dCor(\mathbf{e}_p, \mathbf{e}_{p'}), \tag{4}$$

  where $dCor(\cdot)$ is the distance correlation between intents p and p':

$$dCor(\mathbf{e}_p, \mathbf{e}_{p'}) = \frac{dCov(\mathbf{e}_p, \mathbf{e}_{p'})}{\sqrt{dVar(\mathbf{e}_p) \cdot dVar(\mathbf{e}_{p'})}}, \tag{5}$$

  where $dCov(\cdot)$ is the distance covariance of two representations, and $dVar(\cdot)$ is the distance variance of each intent representation.

Optimizing this loss allows us to encourage the divergence among different intents and makes these intents have distinct boundary, thus endows better explainability of user intents.

3.2 Relational Path-aware Aggregation

Having modeled the user intents, we move on to the representation learning of users and items under the GNN-base paradigm. Previous GNN-based recommender models [38, 39, 41] have shown that the neighborhood aggregation scheme is a promising end-to-end way to integrate multi-hop neighbors into representations. More specifically, the representation vector of an ego node is computed by recursively aggregating and transforming representations of its multi-hop neighbors.

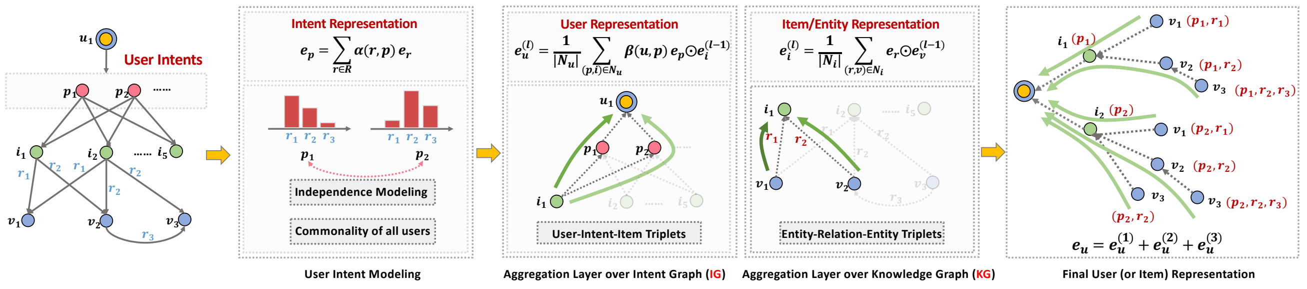

u_1
User Intents
p_1 p_2 ……
i_1 i_2 …… i_5
r_1 r_2 r_1 r_2 r_2 r_3
v_1 v_2 v_3
Intent Representation
$e_p = \sum_{r \in R} \alpha(r,p)\, e_r$
r_1 r_2 r_3 p_1 p_2
Independence Modeling
Commonality of all users
User Intent Modeling
User Representation
$e_u^{(l)} = \frac{1}{|N_u|} \sum_{(p,i) \in N_u} \beta(u,p)\, e_p \odot e_i^{(l-1)}$
User-Intent-Item Triplets
Aggregation Layer over Intent Graph (IG)
Item/Entity Representation
$e_i^{(l)} = \frac{1}{|N_i|} \sum_{(r,v) \in N_i} e_r \odot e_v^{(l-1)}$
Entity-Relation-Entity Triplets
Aggregation Layer over Knowledge Graph (KG)
i_1 (p_1)
v_1 (p_1, r_1)
v_2 (p_1, r_2)
v_3 (p_1, r_2, r_3)
i_2 (p_2)
v_1 (p_2, r_1)
v_2 (p_2, r_2)
v_3 (p_2, r_2, r_3)
v_3 (p_2, r_2)
$e_u = e_u^{(1)} + e_u^{(2)} + e_u^{(3)}$
Final User (or Item) Representation

Figure 3: Illustration of the proposed KGIN framework. Best viewed in color.

However, we argue that current aggregation schemes are mostly node-based, which limit the benefit of the structural knowledge, due to two issues: (1) The aggregators focus on combining the information of neighborhood, without distinguishing which paths they originate from. Considering the example in Figure 2, there are three information channels between the ego node u_1 and its 2-hop neighbor v_2: $u_1 \xleftarrow{p_1} i_1 \xleftarrow{r_2} v_2$ and $u_1 \xleftarrow{p_2} i_2 \xleftarrow{r_2} v_2$. When constructing the neural messages passed by v_2, the node-based aggregators largely transform and rescale v_2's representation by decay factors, without considering influences of different channels. Hence, they are insufficient to preserve the structural information in representations. Moreover, (2) current node-based aggregators usually model KG relations in the decay factors via attention networks [38, 41, 47] to control how much information is propagated from neighbors. This limits the contributions of KG relations to node representations. Moreover, no relation dependency (*e.g.,* (p_2, r_2, r_3) in path $u_1 \xleftarrow{p_2} i_2 \xleftarrow{r_2} v_2 \xleftarrow{r_3} v_3$) is captured in an explicit fashion. Hence, we aim to devise a relational path-aware aggregation scheme to solve these two limitations.

3.2.1 *Aggregation Layer over Intent Graph.*

We first move on to refine collaborative information from IG. As mentioned, the CF effect [26] is powerful to characterize user patterns, by assuming that behavioral similar users would have similar preference on items. This inspires us to treat personal history (*i.e.,* the items a user has adopted before) as the pre-existing features of individual users. Moreover, in our IG, we can capture finer-grained patterns at a granular level of user intents, by assuming that users with similar intents would exhibit similar preference towards items. Considering a user u in IG, we use $\mathcal{N}_u = \{(p, i) | (u, p, i) \in \mathcal{C}\}$ to represent the intent-aware history and the first-order connectivity around u. Technically, we can integrate the intent-aware information from historical items to create the representation of user u as:

$$\mathbf{e}_u^{(1)} = f_{\text{IG}}\Big(\big\{(\mathbf{e}_u^{(0)}, \mathbf{e}_p, \mathbf{e}_i^{(0)}) | (p, i) \in \mathcal{N}_u\big\}\Big), \tag{6}$$

where $\mathbf{e}_u^{(1)} \in \mathbb{R}^d$ is the representation of user u; and $f_{\text{IG}}(\cdot)$ is the aggregator function to characterize each first-order connection (u, p, i). Here we implement $f_{\text{IG}}(\cdot)$ as:

$$\mathbf{e}_u^{(1)} = \frac{1}{|\mathcal{N}_u|} \sum_{(p,i) \in \mathcal{N}_u} \beta(u, p)\mathbf{e}_p \odot \mathbf{e}_i^{(0)}, \tag{7}$$

where $\mathbf{e}_i^{(0)}$ is the ID embedding of item i; $\odot$ is the element-wise product. We harness it with two insights. (1) For a given a user, different intents will have varying contributions to motivate her behaviors. Hence, we introduce an attention score $\beta(u, p)$ to differentiate the importance of intent p as:

$$\beta(u, p) = \frac{\exp(\mathbf{e}_p^\top \mathbf{e}_u^{(0)})}{\sum_{p' \in \mathcal{P}} \exp(\mathbf{e}_{p'}^\top \mathbf{e}_u^{(0)})}, \tag{8}$$

where $\mathbf{e}_u^{(0)} \in \mathbb{R}^d$ is the ID embedding of user u to make the importance score personalized. (2) Unlike the ideas of using the decay factors [38, 41, 47] or regularization terms [41] in previous studies, we highlight the role of intent relations during the aggregation. Hence we construct the item i's message via the element-wise product $\beta(u, p)\mathbf{e}_p \odot \mathbf{e}_i^{(0)}$. As a result, we are able to explicitly express the first-order intent-aware information in the user representations.

3.2.2 *Aggregation Layer over Knowledge Graph.*

We then focus on the aggregation scheme in KG. As one entity can be involved in multiple KG triplets, it can take other connected entities as its attributes, which reflect the content similarity among items. For example, movie *The Hobbit I* can be described by its *director Peter Jackson* and *star Martin Freeman*. More formally, we use $\mathcal{N}_i = \{(r, v) | (i, r, v) \in \mathcal{G}\}$ to represent the attributes and the first-order connectivity about item i, and then integrate the relation-aware information from connected entities to generate the representation of item i:

$$\mathbf{e}_i^{(1)} = f_{\text{KG}}\Big(\big\{(\mathbf{e}_i^{(0)}, \mathbf{e}_r, \mathbf{e}_v^{(0)}) | (r, v) \in \mathcal{N}_i\big\}\Big) \tag{9}$$

where $\mathbf{e}_i^{(1)} \in \mathbb{R}^d$ is the representation collecting the information from the first-order connectivity; and $f_{\text{KG}}(\cdot)$ is the aggregation function to extract and integrate information from each connection (i, r, v). Here we account for the relational context in the aggregator. Intuitively, each KG entity has different semantics and meanings in different relational contexts. For instance, entity *Quentin Tarantino* expresses signals pertinent to the *director* and *star* concepts in two triplets (*Quentin Tarantino, director, Django Unchained*) and (*Quentin Tarantino, star, Django Unchained*), respectively. However, previous studies [38, 41, 47] only model KG relations in the decay factors via the attention mechanism, in order to control the contributions of *Quentin Tarantino* to the representation of *Django Unchained*. Instead, we model the relational context in the aggregator as:

$$\mathbf{e}_i^{(1)} = \frac{1}{|\mathcal{N}_i|} \sum_{(r,v) \in \mathcal{N}_i} \mathbf{e}_r \odot \mathbf{e}_v^{(0)}, \tag{10}$$

where $\mathbf{e}_v^{(0)}$ is the ID embedding of entity v. For each triplet (i, r, v), we devise a relational message $\mathbf{e}_r \odot \mathbf{e}_v^{(0)}$ by modeling the relation r as the projection or rotation operator [30]. As a result, the relational message is able to reveal different meanings carried by the triplets, even when they get the same entities. Analogously, we can obtain the representation $\mathbf{e}_v^{(1)}$ of each KG entity $v \in \mathcal{V}$.

3.2.3 ***Capturing Relational Paths.*** Having modeled the first-order connectivity in Equations (6) and (9), we further stack more aggregation layers to gather the influential signals from higher-order neighbors. Technically, we recursively formulate the representations of user u and item i after l layers as:

$$\mathbf{e}_u^{(l)} = f_{\mathrm{IG}}\Big(\{(\mathbf{e}_u^{(l-1)}, \mathbf{e}_p, \mathbf{e}_i^{(l-1)}) | (p, i) \in \mathcal{N}_u\}\Big),$$
$$\mathbf{e}_i^{(l)} = f_{\mathrm{KG}}\Big(\{(\mathbf{e}_i^{(l-1)}, \mathbf{e}_r, \mathbf{e}_v^{(l-1)}) | (r, v) \in \mathcal{N}_i\}\Big), \tag{11}$$

where $\mathbf{e}_u^{(l-1)}$, $\mathbf{e}_i^{(l-1)}$, $\mathbf{e}_v^{(l-1)}$ separately denote the representations of user u, item i, and entity v, which memorize the relational signals being propagated from their (l-1)-hop neighbors. Benefiting from our relational modeling, these representations are able to store the holistic semantics of multi-hop paths, and highlight the relational dependencies. Let $s = i \xrightarrow{r_1} s_1 \xrightarrow{r_2} \cdots s_{l-1} \xrightarrow{r_l} s_l$ be a l-hop path rooted at item i, which contains a sequence of connected triplets. Its relational path is represented as the sequence of relations merely, *i.e.,* $(r_1, r_2, \cdots, r_l)$. We can rewrite the representation $\mathbf{e}_i^{(l)}$ as follows:

$$\mathbf{e}_i^{(l)} = \sum_{s \in \mathcal{N}_i^l} \frac{\mathbf{e}_{r_1}}{|\mathcal{N}_{s_1}|} \odot \frac{\mathbf{e}_{r_2}}{|\mathcal{N}_{s_2}|} \odot \cdots \odot \frac{\mathbf{e}_{r_l}}{|\mathcal{N}_{s_l}|} \odot \mathbf{e}_{s_l}^{(0)}, \tag{12}$$

where $\mathcal{N}_i^l$ is the set of all i's l-hop paths. Clearly, this representation reflects the interactions among relations and preserves the holistic semantics of paths. This is significantly different from the current aggregation mechanism adopted in knowledge-aware recommenders, which overlook the importance of KG relations and thus fail to capture the relational paths.

3.3 Model Prediction

After L layers, we obtain the representations of user u and item i at different layers and then sum them up as the final representations:

$$\mathbf{e}_u^* = \mathbf{e}_u^{(0)} + \cdots + \mathbf{e}_u^{(L)}, \qquad \mathbf{e}_i^* = \mathbf{e}_i^{(0)} + \cdots + \mathbf{e}_i^{(L)}. \tag{13}$$

By doing so, the intent-aware relationships and the KG relation dependencies of paths are encoded in the final representations.

Thereafter, we employ the inner product on the user and item representations to predict how likely the user would adopt the item:

$$\hat{y}_{ui} = \mathbf{e}_u^{*\top} \mathbf{e}_i^*. \tag{14}$$

3.4 Model Optimization

We opt for the pairwise BPR loss [26] to reconstruct the historical data. Specifically, it considers that for a given user, her historical items should be assigned with higher prediction scores than the unobserved items:

$$\mathcal{L}_{\mathrm{BPR}} = \sum_{(u,i,j) \in O} -\ln \sigma(\hat{y}_{ui} - \hat{y}_{uj}), \tag{15}$$

where $O = \{(u, i, j) | (u, i) \in O^+, (u, j) \in O^-\}$ is the training dataset consisting of the observed interactions O^+ and unobserved counterparts O^-; $\sigma(\cdot)$ is the sigmoid function. By combining the independence loss and BPR loss, we minimize the following objective function to learn the model parameter:

$$\mathcal{L}_{\mathrm{KGIN}} = \mathcal{L}_{\mathrm{BPR}} + \lambda_1 \mathcal{L}_{\mathrm{IND}} + \lambda_2 \|\Theta\|_2^2, \tag{16}$$

where $\Theta = \{\mathbf{e}_u^{(0)}, \mathbf{e}_v^{(0)}, \mathbf{e}_r, \mathbf{e}_p, \mathbf{w} | u \in \mathcal{U}, v \in \mathcal{V}, p \in \mathcal{P}\}$ is the set of model parameters (note that the item set $\mathcal{I} \subset \mathcal{V}$); λ_1 and λ_2 are two hyperparameters to control the independence loss (Equation (6)) and L_2 regularization term, respectively.

3.5 Model Analysis

3.5.1 ***Model Size.*** Recent studies [48] have shown that using nonlinear feature transformations possibly makes GNNs difficult to train. Hence, in the aggregation scheme of KGIN, we discard the nonlinear activation functions and feature transformation matrices. Hence, the model parameters of KGIN consist of (1) ID embeddings of users, KG entities (including items), and KG relations $\{\mathbf{e}_u^{(0)}, \mathbf{e}_v^{(0)}, \mathbf{e}_r | u \in \mathcal{U}, v \in \mathcal{V}\}$; and (2) ID embeddings of user intents $\{\mathbf{e}_p | p \in \mathcal{P}\}$ and the attention weights $\mathbf{w}$.

3.5.2 ***Time Complexity.*** The time cost of KGIN mainly comes from the user intent modeling and aggregation scheme. In the aggregations over IG, the computational complexity of user representations is $O(L|C|d)$, where L, $|C|$, and d denote the number of layers, the number of triplets in IG, and the embedding size, respectively. In the aggregation over KG, the time cost of updating entity representations is $O(L|\mathcal{G}|d)$, where $|\mathcal{G}|$ is the number of KG triplets. As for the independence modeling, the cost of distance correlation is $O(|\mathcal{P}|(|\mathcal{P}| - 1)/2)$, where $|\mathcal{P}|$ is the number of user intents. In total, the time complexity of the whole training epoch is $O(L|C|d + L|\mathcal{G}|d + |\mathcal{P}|(|\mathcal{P}| - 1)/2)$. Under the same experimental settings (*i.e.,* representation sizes at different layers), KGIN has comparable complexity to KGAT and CKAN.

4 EXPERIMENTS

We provide empirical results to demonstrate the effectiveness of our proposed KGIN. The experiments are designed to answer the following research questions:

- **RQ1:** How does KGIN perform, comparing to the state-of-the-art knowledge-aware recommender models?
- **RQ2:** What is the impact of the designs (*e.g.,* the number and independence of user intents, the depth of relational paths) on the improvement of KGIN's relational modeling?
- **RQ3:** Can KGIN provide insights on user intents and give an intuitive impression of explainability?

4.1 Experimental Settings

4.1.1 ***Dataset Description.*** We use three benchmark datasets for book, music, and fashion outfit recommendation in the experiments: (1) We use the Amazon-Book and Last-FM datasets released by KGAT [41]; And (2) we further introduce the Alibaba-iFashion dataset [8] to investigate the effectiveness of item knowledge. This is a fashion outfit dataset collected from Alibaba online shopping systems. The outfits are viewed as the items being recommended

Table 1: Statistics of the datasets.

		Amazon-Book	Last-FM	Alibaba-iFashion
User-Item Interaction	#Users	70,679	23,566	114,737
	#Items	24,915	48,123	30,040
	#Interactions	847,733	3,034,796	1,781,093
Knowledge Graph	#Entities	88,572	58,266	59,156
	#Relations	39	9	51
	#Triplets	2,557,746	464,567	279,155

to users, where each outfit consists of multiple fashion staffs (*e.g.,* tops, bottoms, shoes, accessories), and these staffs follow a fashion taxonomy and are assigned with different fashion categories (*e.g.,* jeans, T-shirts). We extract such attributes as the KG data of outfits. Moreover, in order to ensure the data quality, we adopt the 10-core setting, *i.e.,* discarding users and items with less than ten interactions, and filtering out KG entities involved less than ten triplets. The statistics of datasets are summarized in Table 1, where we only list the number of canonical relations and construct triplets with the inverse relations in experiments. Closely Following prior studies [41, 45], we use the same data partition. In the training phase, each observed user-item interaction is a positive instance, while an item that the user did not adopt before is randomly sampled to pair the user as a negative instance.

4.1.2 ***Evaluation Metrics.*** In the evaluation phase, we conduct the all-ranking strategy [20], rather than sampled metrics like leaving one item out [44] or sampling a smaller set of users [38, 39, 47]. To be more specific, for each user, the full items that she has not adopted before are viewed as negative, and the relevant items in the testing set are treated as positive. All these items are ranked based on the predictions of recommender model. To evaluate top-K recommendation, we adopt the protocols [20]: recall@K and ndcg@K, where K is set as 20 by default. We report the average metrics for all users in the testing set.

4.1.3 ***Alternative Baselines.*** We compare KGIN with the state-of-the-art methods, covering KG-free (MF), embedding-based (CKE), and GNN-based (KGAT, KGNN-LS, CKAN, and RGCN) methods:

- **MF** [26] (matrix factorization) only considers the user-item interactions, while leaving KG untouched. Technically, it uses ID embeddings of users and items to perform the prediction.
- **CKE** [51] is a representative embedding-based method, which leverages KG embeddings of entities derived from TransR [22] as ID embeddings of items under the MF framework. Where, KG relations are only used as the constraints in TransR to regularize the representations of endpoints.
- **KGNN-LS** [38] is a GNN-based model, which converts KG into user-specific graphs, and then considers user preference on KG relations and label smoothness in the information aggregation phase, so as to generate user-specific item representations. It models relations in decay factors.
- **KGAT** [41] is a state-of-the-art GNN-based recommender. It applies an attentive neighborhood aggregation mechanism on a holistic graph, which combines KG with the user-item graph, to generate user and item representations. User-item relationships and KG relations serve as the attentive weights in adjacent matrix.

Table 2: Overall performance comparison.

	Amazon-Book		Last-FM		Alibaba-iFashion	
	recall	ndcg	recall	ndcg	recall	ndcg
MF	0.1300	0.0678	0.0724	0.0617	0.1095	0.0670
CKE	0.1342	0.0698	0.0732	0.0630	0.1103	0.0676
KGAT	0.1487	0.0799	0.0873	0.0744	0.1030	0.0627
KGNN-LS	0.1362	0.0560	0.0880	0.0642	0.1039	0.0557
CKAN	0.1442	0.0698	0.0812	0.0660	0.0970	0.0509
R-GCN	0.1220	0.0646	0.0743	0.0631	0.0860	0.0515
KGIN-3	**0.1687***	**0.0915***	**0.0978***	**0.0848***	**0.1147***	**0.0716***
%Imp.	13.44%	14.51%	11.13%	13.97%	3.98%	5.91%

- **CKAN** [47] is built upon KGNN-LS, which utilizes different neighborhood aggregation schemes on the user-item graph and KG respectively, to obtain user and item embeddings.
- **R-GCN** [27] is originally proposed for the knowledge graph completion task, which views various KG relations as different channels of information flow when aggregating neighboring nodes. Here we transfer it to the recommendation task.

4.1.4 ***Parameter Settings.*** We implement our KGIN model in PyTorch, and have released our implementations (code, datasets, parameter settings, and training logs) to facilitate reproducibility. For a fair comparison, we fix the size of ID embeddings d as 64, the optimizer as Adam [18], and the batch size as 1024 for all methods. A grid search is conducted to confirm the optimal settings for each method — more specifically, the learning rate is tuned in $\{10^{-4}, 10^{-3}, 10^{-2}\}$, the coefficients of additional constraints (*e.g.,* L_2 regularization in all methods, independence modeling in KGIN, TransR in CKE and KGAT, label smoothness in KGNN-LS) are searched in $\{10^{-5}, 10^{-4}, \cdots, 10^{-1}\}$, and the number of GNN layers L is tuned in $\{1, 2, 3\}$ for GNN-based methods. Moreover, for KGNN-LS and CKAN, we set the size of neighborhood as 16 and the batch size as 128. We initialize model parameters with Xavier [11], while using the pre-trained ID embeddings of MF as the initialization of KGAT.

The detailed settings of KGIN are provided in Appendix A.1. We observe that using Equations (3) and (4) have similar trends and performance, hence report the results of Equation (3). We use KGIN-3 to denote the recommender model with three relational path aggregation layers, and similar notations for others. Without specification, we fix the number of user intents $|\mathcal{P}|$ as 4 and the number of relational path aggregation layers L as 3. Moreover, in Sections 4.3.1 and 4.3.2, we study their influence by varying K in $\{1, 2, 4, 8\}$ and L in $\{1, 2, 3\}$, respectively.

4.2 Performance Comparison (RQ1)

We begin with the comparison *w.r.t.* recall@20 and ndcg@20. The empirical results are reported in Table 2, where %Imp. denotes the relative improvements of the best performing method (starred) over the strongest baselines (underlined). We find that:

- KGIN consistently outperforms all baselines across three datasets in terms of all measures. More specifically, it achieves significant improvements over the strongest baselines *w.r.t.* ndcg@20 by 14.51%, 13.97%, and 5.91% in Amazon-Book, Last-FM, and Alibaba-iFashion, respectively. This demonstrates the rationality and effectiveness of KGIN. We attribute these improvements to the relational modeling of KGIN: (1) By uncovering user intents,

Table 3: Impact of presence of user intents and KG relations.

	Amazon-Book		Last-FM		Alibaba-iFashion	
	recall	ndcg	recall	ndcg	recall	ndcg
w/o I&R	0.1518	0.0816	0.0802	0.0669	0.0862	0.0530
w/o I	0.1627	0.0870	0.0942	0.0819	0.1103	0.0678

Table 4: Impact of the number of layers L.

	Amazon-Book		Last-FM		Alibaba-iFashion	
	recall	ndcg	recall	ndcg	recall	ndcg
KGIN-1	0.1455	0.0766	0.0831	0.0707	0.1045	0.0638
KGIN-2	0.1652	0.0892	0.0920	0.0791	0.1162	0.0723
KGIN-3	0.1687	0.0915	0.0978	0.0848	0.1147	0.0716

KGIN is able to better characterize the relationships between users and items, and result in more powerful representations of users and items. In contrast, all baselines ignore the hidden user intents, and model user-item edges as a homogeneous channel to collect information; (2) Benefiting from our relational path aggregation scheme, KGIN can preserve the holistic semantics of paths and collect more informative signals from KG, than the GNN-based baselines (*i.e.,* KGAT, CKAN, KGNN-LS); (3) Applying different aggregation schemes on IG and KG makes KGIN better able to encode the collaborative signals and item knowledge into user and item representations.

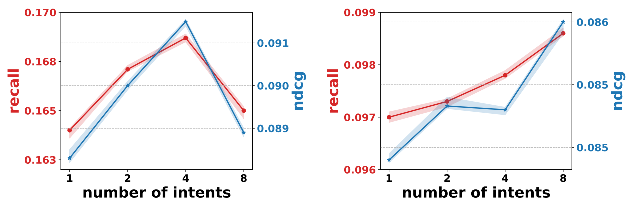
- Jointly analyzing KGIN across the three datasets, we find that the improvement on Amazon-Book is more significant than that on Alibaba-iFashion. This is reasonable since (1) both interaction and KG data on Amazon-Book offer denser and richer information than that on Alibaba-iFashion; and (2) in Alibaba-iFashion, the first-order connectivity (*fashion outfit, including, fashion staff*) dominate the KG triplets. This indicates that KGIN is good at fulfilling the potentials of long-range connectivity.
- Leaving KG untapped limits the performance of MF. By simply incorporating KG embeddings into MF, CKE performs better than MF. Such findings are consist to prior studies [4], indicating the importance of side information like KG.
- GNN-based methods (*i.e.,* KGAT, CKAN, KGNN-LS) outperform CKE in Amazon-Book and Last-FM, suggesting the importance of modeling long-range connectivity. These improvements come from using the local structure of a node — more specifically, multi-hop neighborhood — to improve the representation learning. However, in Alibaba-iFashion, the performance of CKE is better than them. Some possible reasons are: (1) These GNN-based methods involve additional nonlinear feature transformation, which are rather heavy and burdensome to train, thus degrades performance [14, 48]; (2) TransR in CKE successfully captures the major first-order connectivity in Alibaba-iFashion.
- The results of KGAT, KGNN-LS, and CKAN are at the same level, being better than R-GCN. Although the utility of transforming neighbors' information via KG relations in R-GCN is better than working as decay factors in the others, R-GCN is not originally designed for recommendation and thus fails to model user-item relationships properly.

4.3 Relational Modeling of KGIN (RQ2)

As the relational modeling is at the core of KGIN, we also conduct ablation studies to investigate the effectiveness — specifically, how the presence of user intents and KG relations, the number of relational path aggregation layers, the granularity of user intents, and the independence of user intents influence our model.

Table 5: Impact of independence modeling.

	Amazon-Book		Last-FM		Alibaba-iFashion	
	w/ Ind	w/o Ind	w/ Ind	w/o Ind	w/ Ind	w/o Ind
distance correlation	0.0389	0.3490	0.0365	0.4944	0.0112	0.3121



(a) Amazon-Book **(b) Last-FM**

Figure 4: Impact of intent number ($|\mathcal{P}|$). Best viewed in color.

4.3.1 *Impact of Presence of User Intents & KG Relations.*

We first answer the question: Is it of importance to consider user intents or KG relations? Towards this end, two variants are constructed by (1) discarding all user intents and KG relations, termed KGIN-$3_{\text{w/o I\&R}}$, and (2) removing all user intents only ($|\mathcal{P}| = 0$), termed KGIN-$3_{\text{w/o I}}$. We summarize the results in Table 3.

Obviously, compared with KGIN-3 in Table 2, removing all relations (*i.e.,* KGIN-$3_{\text{w/o I\&R}}$) dramatically reduces the predictive accuracy, indicating the necessity of relational modeling. To be more specific, KGIN-$3_{\text{w/o I\&R}}$ only propagates nodes' information in one space, without preserving any relational semantics, thus distorts the internal relationships among nodes. Analogously, leaving hidden user intents unexplored (*i.e.,* KGIN-$3_{\text{w/o I}}$) also downgrades the performance. Although KGIN-$3_{\text{w/o I}}$ retains the modeling of KG relations, it only considers coarser-gained preference of users and thus leads to suboptimal user representations.

4.3.2 *Impact of Model Depth.*

We then consider varying the number of relational path aggregation layers. Stacking more layers is able to integrate the information carried by longer-range connectivity (*i.e.,* longer paths) into node representations. Here we search L in the range of $\{1, 2, 3\}$ and summarize the results in Table 4. We observe that:

- Increasing the model depth is able to enhance the predictive results in most cases. To be more specific, KGIN-2 substantially achieves significant improvements over KGIN-1. We attribute such improvements to two reasons: (1) Stacking more layers explore more relevant items connected by some KG triplets and deepens the understanding of user interest. KGIN-1 only takes the first-order connectivity (*e.g.,* user-intent-item triplets, KG triplets) into consideration, while KGIN-2 reveals the two-hop paths; (2) More information pertinent to user intents are derived from longer relational paths, thus better profiling user preference on items.
- Continuing one more exploration beyond KGIN-2, the results of KGIN-3 are consistently better in Amazon-Book and Last-FM. This empirically shows that higher-order connectivity is complementary to the second-order one, thus resulting in better node representations.

- However, the results of KGIN-3 are worse than KGIN-2 in Alibaba-iFashion. This again admits the inherent characteristics of Alibaba-iFashion — most of KG triplets are the first-order connectivity (*fashion outfit, including, fashion staff*) of items, which have been captured in KGIN-2.

4.3.3 ***Impact of Intent Modeling.*** To analyze the influence of intents number, we vary $|\mathcal{P}|$ in range of $\{1, 2, 4, 8\}$ and illustrate the performance changing curves on Amazon-Book and Last-FM datasets in Figure 4. We find that:

- Increasing the intent number enhances the performance in most cases. Specifically, when only modeling a coarse-grained relation (*i.e.,* $|\mathcal{P}| = 1$), KGIN-3 performs poor across the board. This again emphasizes the benefits of exploring multiple user intents.
- In Amazon-Book, continuing one more partition beyond $|\mathcal{P}| = 4$ impairs the accuracy. One possible reason is that independence modeling encourages the irrelevance among intents, but also makes some intents too fine-grained to carry useful information. We leave the exploration of intent granularity to future work.
- Interestingly, comparing to the results on Amazon-Book, setting $|\mathcal{P}| = 8$ improves the accuracy on Last-FM, although Amazon-Book contains richer set of KG relations as compared to Last-FM. We attribute this to the difference of two datasets. In particular, KG in Last-FM is converted from the attributes of albums, songs, and artists, while KG in Amazon-Book is extracted from Freebase and contains noisy relations irrelevant to user behaviors.

We also conduct an ablation study to investigate the influence of independence modeling (*cf.* Section 3.1.2). Specifically, we disable this module to build a variant KGIN-$3_{\text{w/o Ind}}$, and show the results *w.r.t.* distance correlation in Table 5. Clearly, while approaching the comparable performance of recommendation to KGIN-3, KGIN-$3_{\text{w/o Ind}}$ achieves larger correlation coefficients and fails to differentiate user intents, which are still opaque to understand user behaviors.

4.4 Explainability of KGIN (RQ3)

In this section, we present the semantics of user intents, and offer two examples of Amazon-Book and Last-FM to give an intuitive impression of our explainability. As shown in Figure 5, we have the following observations:

- KGIN first induces intents — the commonality of all users — with various combinations of KG relations. For an intent, the weight of a relation reflects its importance to influence user behaviors. For example, in Amazon-Book, the top two relations of the first intent p_1 are *theater.play.genre* and *theater.plays.in-this-genre*, while *date-of-the-first-performance* and *fictional-universe* are assigned with the highest scores for the second intent p_3. Clearly, the learned intents abstract the shared reasons of user's choices. Moreover, thanks to the independence modeling, the intents tend to have distinct boundary, thus describing user behaviors from different and independent angles. However, p_1 and p_3 are highly relevant. This makes sense since only 9 relations exist in Last-FM.
- It can be found that some relations get high weights in multiple intents, like *version* in Last-FM. This indicates that such relations are common factors pertinent to user behaviors. Combining it with other relations like *featured-artist*, KGIN induces the intent p_1 as the special version of music created by a certain artist.
- KGIN creates instance-wise explanations for each interaction — the personalization of a single user. For the interaction u_{231}-i_{21904} in Amazon-Book, KGIN searches the most influential intent p_1 based on the attention scores (*cf.* Equation (8)). Thus, it explains this behavior as *User* u_{231} *selects music* i_{21904} *since it matches her interest on the featured artist and certain version.*

5 RELATED WORK

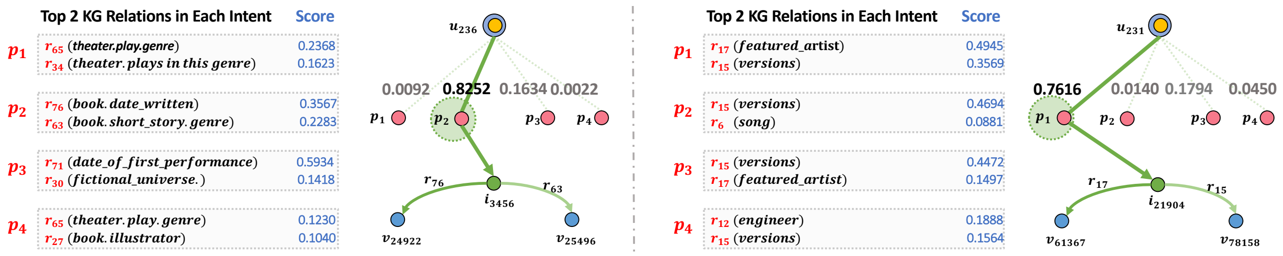
Existing recommender models incorporated with KG roughly fall into four groups.

Embedding-based Methods [1, 4, 16, 35, 37, 51] focus mainly on the first-order connectivity (*i.e.,* user-item pairs in interaction data, triplets in KG), hire KG embedding techniques (*e.g.,* TransE [3] and TransH [46]) to learn entity embeddings, and then use them as prior or context information of items to guide the recommender model. For example, CKE [51] applies TransE on KG triplets, and feed the knowledge-aware embeddings of items into matrix factorization (MF) [26]. KTUP [4] employs TransH on user-item interactions and KG triplets simultaneously, to jointly learn user preference and perform KG completion. Although these methods demonstrate the benefits of knowledge-aware embeddings, they ignore the higher-order connectivity. This make them fail to capture the long-range semantics or sequential dependencies of paths between two nodes, and thus limits their ability to uncover the underlying user-item relationships.

Path-based Methods [5, 15, 25, 31, 36, 44] account for the long-range connectivity by extracting paths that connect the target user and item nodes via KG entities. Then these paths are used to predict user preference, such as via recurrent neural networks [31, 44] and memory network [36]. For example, RippleNet [36] memorizes the item representations along with paths rooted at each user, and uses them to enhance user representations. Clearly, the recommendation accuracy heavily relies on the quality of paths. However, two mainstream path extraction methods suffer from some inherent limitations: (1) Applying brute-force search easily leads to labor-intensive and time-consuming feature engineering, when large-scale graphs are involved [44]; (2) When using meta-path patterns to filter path instances, it requires domain experts to predefine the domain-specific patterns, thus resulting in poor transferability to different domains [15, 17].

Policy-based Methods [40, 45, 49, 52, 53] get inspiration from recent success of reinforcement learning (RL), and design RL agents to learn path-finding policy. For example, PGPR [49] exploits a policy network to explore items of interest for a target user. These RL-based policy networks can be viewed as efficient and cheap alternatives to the brute-force search, which serve as the backbone models of conversational recommender systems [10, 21]. However, the sparse reward signals, huge action spaces, and policy gradient-based optimization make these networks hard to train and converge to a stable and satisfying solution [50, 52].

GNN-based Methods [17, 38, 39, 41, 47] are founded upon the information aggregation mechanism of graph neural networks (GNNs) [13, 14, 19, 34, 42]. Typically, it incorporates information



	Top 2 KG Relations in Each Intent	Score
p_1	r_{65} (*theater.play.genre*)	0.2368
	r_{34} (*theater. plays in this genre*)	0.1623
p_2	r_{76} (*book. date_written*)	0.3567
	r_{63} (*book. short_story. genre*)	0.2283
p_3	r_{71} (*date_of_first_performance*)	0.5934
	r_{30} (*fictional_universe.*)	0.1418
p_4	r_{65} (*theater. play. genre*)	0.1230
	r_{27} (*book. illustrator*)	0.1040

	Top 2 KG Relations in Each Intent	Score
p_1	r_{17} (*featured_artist*)	0.4945
	r_{15} (*versions*)	0.3569
p_2	r_{15} (*versions*)	0.4694
	r_6 (*song*)	0.0881
p_3	r_{15} (*versions*)	0.4472
	r_{17} (*featured_artist*)	0.1497
p_4	r_{12} (*engineer*)	0.1888
	r_{15} (*versions*)	0.1564

Figure 5: Explanations of user intents and real cases in Amazon-Book (left) and Last-FM (right). Best viewed in color.

from the one-hop nodes to update the representations of ego nodes; when recursively performing such propagations, information from multi-hop nodes can be encoded in the representations. As such, these methods are able to model long-range connectivity. For instance, KGAT [41] combines user-item interactions and KG as a heterogeneous graph, and then applies the aggregation mechanism on it. CKAN [47] uses two different strategies to separately spread collaborative signals and knowledge-aware signals. More recently, NIRec [17] is proposed to combine path- and GNN-based models, which propagates interactive patterns between two nodes through meta-path-guided neighborhoods.

However, to the best of our knowledge, current GNN-based methods assume that only one relation exists among users and items, but leave hidden intents unexplored. Moreover, most of them fail to preserve the relational dependency in paths. Our work differs from them in these relational modeling — we focus on exhibiting user-item relationships at the granularity of intents, and encoding relational paths into the representations, towards better performance and interpretability.

6 CONCLUSION AND FUTURE WORK

In this work, we focused on the relational modeling of knowledge-aware recommendation, especially in GNN-based methods. We proposed a novel framework, KGIN, which approaches better relational modeling from two dimensions: (1) uncovering user-item relationships at the granularity of intents, which are coupled with KG relations to exhibit the explainable semantics; and (2) relational path-aware aggregation, which integrates relational information from multi-hop paths to refine the representations. We further offered in-depth analysis of KGIN *w.r.t.* the effectiveness and explainability of recommendation.

Current works usually frame the KG-based recommendation as a supervised task, where the supervision signal comes from historical interactions only. Such supervisions are too sparse to offer high-quality representations. In future work, we will explore self-supervised learning in recommendation, in order to generate auxiliary supervisions via self-supervised tasks and uncover the internal relationships among data instances. Furthermore, we would like to introduce causal concepts, such as causal effect inference, counterfactual reasoning, and deconfounding, into knowledge-aware recommendation to discover and amplify biases [6].

ACKNOWLEDGMENTS

This research is part of NExT++ research, which is supported by the National Research Foundation, Singapore under its International Research Centres in Singapore Funding Initiative. It is also supported by the National Natural Science Foundation of China (U19A2079, 61972372). This work is also supported by Janet Jenq, Heather Thibodeau and Beatriz Reyero del Rio of eBay Customer Science department and Mitch Wyle and Petra Hofer of eBay eRupt organization.

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] Mohamed Ishmael Belghazi, Aristide Baratin, Sai Rajeswar, Sherjil Ozair, Yoshua Bengio, R. Devon Hjelm, and Aaron C. Courville. 2018. Mutual Information Neural Estimation. In *ICML*, Vol. 80. 530–539.
[3] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-relational Data. In *NeurIPS*. 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 *WWW*. 151–161.
[5] Rose Catherine and William W. Cohen. 2016. Personalized Recommendations using Knowledge Graphs: A Probabilistic Logic Programming Approach. In *RecSys*. 325–332.
[6] Jiawei Chen, Hande Dong, Xiang Wang, Fuli Feng, Meng Wang, and Xiangnan He. 2020. Bias and Debias in Recommender System: A Survey and Future Directions. *CoRR* (2020).
[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. 2020. A Simple Framework for Contrastive Learning of Visual Representations. In *ICML*, Vol. 119. 1597–1607.
[8] Wen Chen, Pipei Huang, Jiaming Xu, Xin Guo, Cheng Guo, Fei Sun, Chao Li, Andreas Pfadler, Huan Zhao, and Binqiang Zhao. 2019. POG: Personalized Outfit Generation for Fashion Recommendation at Alibaba iFashion. In *KDD*. 2662–2670.
[9] Vijay Prakash Dwivedi, Chaitanya K. Joshi, Thomas Laurent, Yoshua Bengio, and Xavier Bresson. 2020. Benchmarking Graph Neural Networks. *CoRR* (2020).
[10] Chongming Gao, Wenqiang Lei, Xiangnan He, Maarten de Rijke, and Tat-Seng Chua. 2021. Advances and Challenges in Conversational Recommender Systems: A Survey. *CoRR* (2021).
[11] Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*, Vol. 9. 249–256.
[12] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *AISTATS*, Vol. 9. 297–304.
[13] William L. Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive Representation Learning on Large Graphs. In *NeurIPS*. 1024–1034.
[14] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. In *SIGIR*.
[15] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In *KDD*. 1531–1540.
[16] Jin Huang, Wayne Xin Zhao, Hongjian Dou, Ji-Rong Wen, and Edward Y. Chang. 2018. Improving Sequential Recommendation with Knowledge-Enhanced Memory Networks. In *SIGIR*. 505–514.
[17] Jiarui Jin, Jiarui Qin, Yuchen Fang, Kounianhua Du, Weinan Zhang, Yong Yu, Zheng Zhang, and Alexander J. Smola. 2020. An Efficient Neighborhood-based Interaction Model for Recommendation on Heterogeneous Graph. In *KDD*. 75–84.
[18] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.
[19] Thomas N. Kipf and Max Welling. 2017. Semi-Supervised Classification with Graph Convolutional Networks. In *ICLR*.

[20] Walid Krichene and Steffen Rendle. 2020. On Sampled Metrics for Item Recommendation. In *KDD*. 1748–1757.

[21] Wenqiang Lei, Gangyi Zhang, Xiangnan He, Yisong Miao, Xiang Wang, Liang Chen, and Tat-Seng Chua. 2020. Interactive Path Reasoning on Graph for Conversational Recommendation. In *KDD*. 2073–2083.

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

[23] Jianxin Ma, Peng Cui, Kun Kuang, Xin Wang, and Wenwu Zhu. 2019. Disentangled Graph Convolutional Networks. In *ICML*, Vol. 97. 4212–4221.

[24] Jianxin Ma, Chang Zhou, Peng Cui, Hongxia Yang, and Wenwu Zhu. 2019. Learning Disentangled Representations for Recommendation. In *NeurIPS*. 5712–5723.

[25] Weizhi Ma, Min Zhang, Yue Cao, Woojeong Jin, Chenyang Wang, Yiqun Liu, Shaoping Ma, and Xiang Ren. 2019. Jointly Learning Explainable Rules for Recommendation with Knowledge Graph. In *WWW*. 1210–1221.

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

[27] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *ESWC*. 593–607.

[28] Chuan Shi, Binbin Hu, Wayne Xin Zhao, and Philip S. Yu. 2019. Heterogeneous Information Network Embedding for Recommendation. *TKDE* 31, 2 (2019), 357–370.

[29] Chuan Shi, Yitong Li, Jiawei Zhang, Yizhou Sun, and Philip S. Yu. 2017. A Survey of Heterogeneous Information Network Analysis. *TKDE* 29, 1 (2017), 17–37.

[30] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. RotatE: Knowledge Graph Embedding by Relational Rotation in Complex Space. In *ICLR*.

[31] Zhu Sun, Jie Yang, Jie Zhang, Alessandro Bozzon, Long-Kai Huang, and Chi Xu. 2018. Recurrent knowledge graph embedding for effective recommendation. In *RecSys*. 297–305.

[32] Gábor J Székely and Maria L Rizzo. 2009. Brownian distance covariance. *The annals of applied statistics* (2009), 1236–1265.

[33] Gábor J Székely, Maria L Rizzo, Nail K Bakirov, et al. 2007. Measuring and testing dependence by correlation of distances. *The annals of statistics* 35, 6 (2007), 2769–2794.

[34] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

[35] Chenyang Wang, Min Zhang, Weizhi Ma, Yiqun Liu, and Shaoping Ma. 2020. Make It a Chorus: Knowledge- and Time-aware Item Modeling for Sequential Recommendation. In *SIGIR*. 109–118.

[36] 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 *CIKM*. 417–426.

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

[38] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. 2019. Knowledge-aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems. In *KDD*. 968–977.

[39] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge Graph Convolutional Networks for Recommender Systems. In *WWW*. 3307–3313.

[40] Pengfei Wang, Yu Fan, Long Xia, Wayne Xin Zhao, ShaoZhang Niu, and Jimmy Huang. 2020. KERL: A Knowledge-Guided Reinforcement Learning Model for Sequential Recommendation. In *SIGIR*. 209–218.

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

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

[43] Xiang Wang, Hongye Jin, An Zhang, Xiangnan He, Tong Xu, and Tat-Seng Chua. 2020. Disentangled Graph Collaborative Filtering. In *SIGIR*. 1001–1010.

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

[45] Xiang Wang, Yaokun Xu, Xiangnan He, Yixin Cao, Meng Wang, and Tat-Seng Chua. 2020. Reinforced Negative Sampling over Knowledge Graph for Recommendation. In *WWW*. 99–109.

[46] Zhigang Wang and Juan-Zi Li. 2016. Text-Enhanced Representation Learning for Knowledge Graph. In *IJCAI*. 1293–1299.

[47] Ze Wang, Guangyan Lin, Huobin Tan, Qinghong Chen, and Xiyang Liu. 2020. CKAN: Collaborative Knowledge-aware Attentive Network for Recommender Systems. In *SIGIR*. 219–228.

[48] Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. 2019. Simplifying Graph Convolutional Networks. In *ICML*, Vol. 97. 6861–6871.

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

[50] Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *EMNLP*. 564–573.

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

[52] Kangzhi Zhao, Xiting Wang, Yuren Zhang, Li Zhao, Zheng Liu, Chunxiao Xing, and Xing Xie. 2020. Leveraging Demonstrations for Reinforcement Recommendation Reasoning over Knowledge Graphs. In *SIGIR*. 239–248.

[53] Sijin Zhou, Xinyi Dai, Haokun Chen, Weinan Zhang, Kan Ren, Ruiming Tang, Xiuqiang He, and Yong Yu. 2020. Interactive Recommender System via Knowledge Graph-enhanced Reinforcement Learning. In *SIGIR*. 179–188.

A APPENDIX

A.1 Reproducibility

We list the parameter settings of KGIN on three datasets in Table 6, where the hyperparameters include the learning rate ρ, the embedding size d, the number of aggregation layers L, the number of user intents $|\mathcal{P}|$, the coefficient λ_1 of the independence modeling $\mathcal{L}_{\text{IND}}$, and the coefficient λ_2 of L_2 regularization. We have released our codes, datasets, model parameters, and training logs at https://github.com/huangtinglin/Knowledge_Graph_based_Intent_Network to facilitate reproducibility.

Table 6: Hyperparameter settings of KGIN.

	ρ	d	L	$\lvert\mathcal{P}\rvert$	λ_1	λ_2
Amazon-Book	10^{-4}	64	3	4	10^{-5}	10^{-5}
Last-FM	10^{-4}	64	3	4	10^{-4}	10^{-5}
Alibaba-iFashion	10^{-4}	64	3	4	10^{-4}	10^{-5}