# Personalized Re-ranking with Item Relationships for E-commerce

Weiwen Liu*
The Chinese University of Hong Kong
wwliu@cse.cuhk.edu.hk

Qing Liu
Huawei Noah's Ark Lab
liuqing48@huawei.com

Ruiming Tang
Huawei Noah's Ark Lab
tangruiming@huawei.com

Junyang Chen†
University of Macau
yb77403@umac.mo

Xiuqiang He
Huawei Noah's Ark Lab
hexiuqiang1@huawei.com

Pheng Ann Heng
The Chinese University of Hong Kong
pheng@cse.cuhk.edu.hk

## ABSTRACT

Re-ranking is a critical task for large-scale commercial recommender systems. Given the initial ranked lists, top candidates are re-ranked to improve the accuracy of the ranking results. However, existing re-ranking strategies are sub-optimal due to (i) most prior works do not consider explicit item relationships, like being substitutable or complementary, which may mutually influence the user satisfaction on other items in the lists, and (ii) they usually apply an identical re-ranking strategy for all users, with personalized user preferences and intents ignored. To resolve the problem, we construct a heterogeneous graph to fuse the initial scoring information and item relationships information. We develop a graph neural network based framework, IRGPR, to explicitly model transitive item relationships by recursively aggregating relational information from multi-hop neighborhoods. We also incorporate a novel intent embedding network to embed personalized user intents into the propagation. We conduct extensive experiments on real-world datasets, demonstrating the effectiveness of IRGPR in re-ranking. Further analysis reveals that modeling the item relationships and personalized intents are particularly useful for improving the performance of re-ranking.

**ACM Reference Format:**
Weiwen Liu, Qing Liu, Ruiming Tang, Junyang Chen, Xiuqiang He, and Pheng Ann Heng. 2020. Personalized Re-ranking with Item Relationships for E-commerce. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3340531.3412332

## 1 INTRODUCTION

Re-ranking, as an effective way of improving recommendation performance, has been largely adopted by modern commercial large-scale recommender systems on platforms such as Amazon and Google [26]. A commercial recommender system consists of three stages in general: candidates generation, scoring, and re-ranking, as illustrated in Figure 1. The system firstly generates candidates from a large pool of items, then these candidates are scored and ranked. Finally, the system conducts re-ranking on the top candidates based on rules or certain functions to further improve the recommendation results [26, 39]. The performance of re-ranking directly influences user experience and satisfaction.

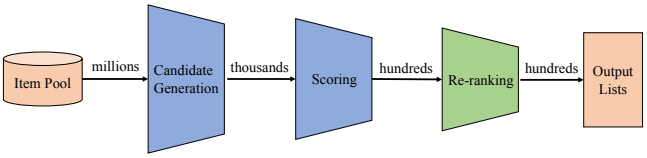



**Figure 1: Architecture of a typical commercial recommender system.**

Two essential facts need to be considered in the re-ranking stage. Firstly, *item relationships* in a ranked list affect the behavior of a user on this list. Typical item relationships for e-commerce include substitutability, complementarity [23, 24] and so on. Substitutable items are interchangeable. Complementary items are usually bought together by users. Items with such specific relationships have mutual influences on each other. For example, a user who is browsing headphones in an online store will be attracted to substitutable headphones that are cheaper or of better quality. Re-ranking the substitutable items to a higher position could improve the recommendation results. For a user who is randomly exploring the system, recommending a list of complementary items to her may better fit the user's interests. The combination of items in the list as a whole affects user satisfaction, therefore incorporating item relationships can improve the accuracy and explainability of re-ranking.

Secondly, different users view a ranked list of items with different preferences and perspectives. For instance, some users prefer a list of substitutable items while others prefer a list of complementary items. Some may be sensitive to *price*. Others may think *quality* is the first priority. As such, manually re-ranking the initially ranked list according to fixed rules or simply using an identical re-ranking strategy for all users would overlook the users' *personalized* interests and intents, which will lead to inferior re-ranking results.

Foundational work has demonstrated the effectiveness of re-ranking models that explicitly model item relationships in a ranked list [1, 17, 26, 39]. For example, DLCM [1] and miDNN [39] utilize

*This work was done when Weiwen Liu was an intern at Huawei Noah's Ark Lab.
†Corresponding author.


*CIKM '20, October 19–23, 2020, Virtual Event, Ireland*

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3412332

a Recurrent Neural Network (RNN) to sequentially encode the distribution of top items in the initial ranked lists and treat it as features of the re-ranking model. Neither of these prior works considers the personalization of the re-ranking model. A recent work PRM [26] uses self-attention architecture to learn the mutual influences between every item pairs in the ranked list. This work shows that the performance of re-ranking improves by realizing personalization with a pre-trained user embedding.

However, these approaches have the following limitations. 1) RNN-based models [1, 39] are not able to model mutual influences of a specific pair of items since all the items in a list are sequentially aggregated together by a recurrent unit. 2) Self-attention mechanism [26] tries to model the mutual influences between every pair of items. But it is not able to distinguish different types of influences, such as complementarity and substitutability, which are common in many e-commerce scenarios. Furthermore, 3) existing works learn item relationships only from the initial ranked lists, while item relationships outside this ranked list are ignored. In fact, the global transitive dependencies among items across different ranked lists carry rich semantic information for re-ranking. Take $u_4$ in Figure 2 as an example. Rather than solely relying on modeling how $v_7$ and $v_8$[1] influences each other, the global item relationships, exploiting all related items ($v_5$, $v_6$, $v_7$, $v_8$) and all transitive relationship paths among them, also contain useful information.

In recent years, the emergence of graph-based representation learning, *e.g.*, Graph Neural Networks (GNN), shows great potential for automated information propagation [21]. Some existing advanced re-ranking approaches can be viewed in the framework of graph neural networks as we will discuss them in detail in Section 2. Recent studies have demonstrated that graph-based representation learning methods are capable of providing general representations to integrate multiple types of information [11, 31, 34]. We believe that using a more sophisticated graph model by introducing user nodes, item nodes, and different types of edges can overcome the limitations of previous re-ranking algorithms, since it enables us to encode different types of item relationships inside and outside the ranked list and to distinguish personalized user intentions.

Specifically, we model items and their associated relationships as the *item relationship graph*, and initial ranking score between users and items as the *user-item scoring graph*, as shown in Figure 2. While the combination of these two graphs is promising for personalized re-ranking, it is nontrivial to extract useful information due to the heterogeneity of the graph. The graph has two types of nodes (users and items) and two types of edges (initial scores and item relations). Since different types of nodes and edges have different semantic meanings, the ability to fuse the diverse semantic information in node representations is required.

To resolve the problem, we propose a graph neural network based framework, **I**tem **R**elationship **G**raph Neural Networks for **P**ersonalized **R**e-ranking (IRGPR). IRGPR learns item representations with a novel global item relationship propagation, as well as user representations with a novel personalized intent propagation. A fully connected network is then deployed to learn the re-ranking score. IRGPR has the ability to model the heterogeneous item relationships, where the initial ranking score is also well-preserved by the model. The main contributions of this paper are as follows:

- We discuss the commonality of the start-of-the-art re-ranking models in the framework of graph neural networks and propose a more sophisticated model that considers different types of item relationships for personalized re-ranking. To the best of our knowledge, this is the first work that incorporates explicit item relationships for re-ranking in e-commerce.
- We propose a GNN-based framework, IRGPR, with a global item relationship propagation and a personalized intent propagation for personalized re-ranking. IRGPR can better exploit multi-hop item relationships and initial ranking scores to improve the recommendation results.
- Extensive experiments demonstrate the effectiveness of IRGPR, which outperforms the state-of-the-art models, including RNN-based models and Transformer-based models. Experiments are conducted on several widely used e-commerce datasets from Amazon in terms of ranking metrics such as precision and MAP.

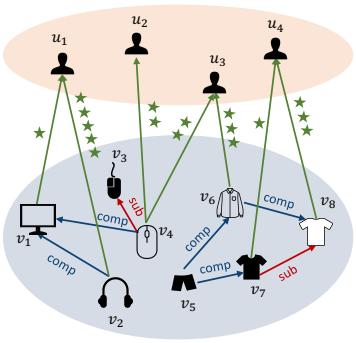



**Figure 2: The illustration of the heterogeneous graph in personalized re-ranking task. Green edges with stars represent the initial ranking scores. Blue and red edges represent being complementary and substitutable, respectively.**

## 2 DISCUSSION: RE-RANKING IN GNN FRAMEWORK

The concept of graph neural networks (GNN) was first proposed by Scarselli *et al.* [29], which extended existing neural networks for graph-structured data. We find that notable examples of neural network based re-ranking models like PRM [26] and DLCM [1] can be viewed in GNN framework. Let $G = (\mathcal{V}, \mathcal{E})$ be a graph with node feature vectors $x_v \in \mathbb{R}^d$ with dimension $d$ for node $v \in \mathcal{V}$. The goal of GNN is to learn a node representation vector $h_v \in \mathbb{R}^d$ which contains the information of neighborhood for each node. The node representation vectors can be obtained by performing the following propagation operation $L$ times to gather information from $L$-hop neighbors [13].

$$m_v^{(l+1)} = \sum_{w \in \mathcal{N}_v} M_l \left( h_v^{(l)}, h_w^{(l)} \right), \tag{1}$$

$$h_v^{(l+1)} = U_l \left( m_v^{(l+1)}, h_v^{(l)} \right) \tag{2}$$

[1]Note that, *Candidate Generation* module in Figure 1 may generate different candidates for individual users, so that we have different candidates to be ranked for different users.


(a) Graph for RNN based model (b) Graph for Transformer based model

**Figure 3: Graphs for RNN and Transformer based models.**

where $l = 0, 1, \ldots, L$ means the $l$-th propagation, $\mathcal{N}_v$ denotes the neighbors of node $v$ in graph $G$, $M_l(\cdot)$ and $U_l(\cdot)$ are the parametric functions to be learned, and $h_v^{(0)} = x_v$. Basically, these two steps present how information from the neighbors of node $v$ is collected and how the collected neighbor information is fused into target node $v$. For example, in Graph Convolutional Neural Networks (GCN) [7], $M_l(\cdot)$ is a linear transformation and $U_l(\cdot)$ is an activation function [13],

$$m_v^{(l+1)} = \sum_{w \in \mathcal{N}_v} A_{vw}^{(l)} h_w^{(l)}, \tag{3}$$

$$h_v^{(l+1)} = \mu\left(m_v^{(l+1)}\right), \tag{4}$$

where $\mu$ is the activation function, *e.g.*, ReLU. The matrices $A_{vw}^{(l)}$ are parameterized by the eigenvectors of the graph Laplacian and some learned parameters of the model.

After obtaining the representation vectors $h_v^L$, the final decisions $\hat{y}$, such as node labels, can then be generated with a parametric function $R(\cdot)$ [13]

$$\hat{y} = R\left(h_v^L | v \in \mathcal{V}\right). \tag{5}$$

We now present how RNN-based re-ranking models, *e.g.*, DLCM [1] and Transformer-based re-ranking model, *e.g.*, PRM [26], can fall into GNN framework.

**Relation to DLCM.** DLCM [1] applies a Gated Recurrent Unit (GRU) on the initial ranked list to model the influences of a item in the list on adjacent items. The list is a special case of the graph with a set of nodes connected in sequence. Different from a standard GNN, a typical one direction RNN performs the propagation sequentially, from left to right as in Figure 3(a). Specifically, for the GRU used in DLCM, the learning procedure can be re-written as:

$$m_v^{(l+1)} = \sum_{w \in \mathcal{N}_v} (1 - u_l) \odot h_w^{(l)} + u_l \odot s_l, \tag{6}$$

$$h_v^{(l+1)} = m_v^{(l+1)}, \tag{7}$$

where

$$u_l = \sigma\left(W_u^0 \cdot x_v + W_u^1 \cdot h_w^{(l)}\right), \tag{8}$$

$$s_l = \tanh\left(W_s^0 \cdot x_v + W_s^1 \cdot \left(r_l \odot h_w^{(l)}\right)\right), \tag{9}$$

$$r_l = \sigma\left(W_r^0 \cdot x_v + W_r^1 \cdot h_w^{(l)}\right). \tag{10}$$

In Eq. (8)-(10), $\sigma(\cdot)$ is the sigmoid function, $W_u^0$, $W_u^1$, $W_s^0$, $W_s^1$, $W_r^0$ and $W_r^1 \in \mathbb{R}^{d \times d}$ are the parametric matrices, and $\odot$ represents element-wise product. For re-ranking, a sequence is less informative than a graph since only adjacent items are connected in the sequence. Information of previously encoded items degrades along with the propagation.

**Relation to PRM.** PRM [26] adopts a Transformer structure to encode the mutual influences between any two items in a given ranked list. If we build a complete graph with nodes denoting the items in a ranked list, as shown in Figure 3(b), PRM is equivalent to a GNN model which can be re-written as:

$$m_v^{(l+1)} = \sum_{w \in \mathcal{N}_v} \alpha_{vw} W_V^{(l)} h_w^{(l)}, \tag{11}$$

$$h_v^{(l+1)} = \text{FFN}(m_v^{(l+1)}), \tag{12}$$

where

$$\alpha_{vw} = \frac{e^{\frac{1}{\sqrt{d}}\left(W_Q^{(l)} h_v^{(l)}\right)^\top \cdot \left(W_K^{(l)} h_w^{(l)}\right)}}{\sum_{i \in \mathcal{N}_v} e^{\frac{1}{\sqrt{d}}\left(W_Q^{(l)} h_v^{(l)}\right)^\top \cdot \left(W_K^{(l)} h_i^{(l)}\right)}}. \tag{13}$$

In Eq. (11)-(13), $1/\sqrt{d}$ is the scaling factor, FFN means feed-forward networks, $W_Q^{(l)}$, $W_K^{(l)}$, and $W_V^{(l)}$ are parameter matrices. The final re-ranking score of each item is computed by several feed-forward layers followed by a softmax layer, the input of the feed-forward layers is the representation learned by Transformer. Note that Eq. (11) can be easily extended to the multi-heads version by concatenating multiple heads together.

Transformer assumes each node interacts with every other node in the graph. Thus, the principle of PRM is to propagate information on the fully connected item graph based on the initial ranked list, which can also be accomplished by graph neural networks. However, Transformer does not scale well to large graphs which are common in a real-world e-commerce recommender system, since the number of edges in the Transformer scales quadratically with the number of nodes. Therefore, this method is limited to modeling mutual influences among items in small graphs.

In sum, we provide a new perspective to understand the state-of-the-art works for re-ranking in the GNN framework. However, these works have their own limitations. As illustrated in Figure 3, items in RNN only connect to a single item according to the order in the ranked lists, while items connect to every other item in Transformer. In this paper, we introduce a perspective to model item relationships and personalized user intents under the GNN framework for re-ranking to improve the quality of recommendation.

## 3 PROPOSED MODEL

In this section, we first present the problem formulation of personalized re-ranking for recommendation. Then we propose to construct a unified heterogeneous graph to explicitly connect two complementary sources of information – item relationships and initial ranking scores. We design a graph neural network based framework, IRGPR, to resolve the problem. By recursively aggregating information on the heterogeneous graph, item mutual influences and user intents are encoded into the item representations and user representations, respectively.

The purpose of this work is to present a personalized re-ranking scheme that considers item relationships and user intents. Given a set of users $\mathcal{U} = \{u_1, \ldots, u_p\}$, a set of items $\mathcal{V} = \{v_1, \ldots, v_q\}$, and an initial ranked list $R(u)$ of items for user $u \in \mathcal{U}$, our task is to re-rank $R(u)$ and generate a list $S(u)$ that better meets the users' interests and needs.

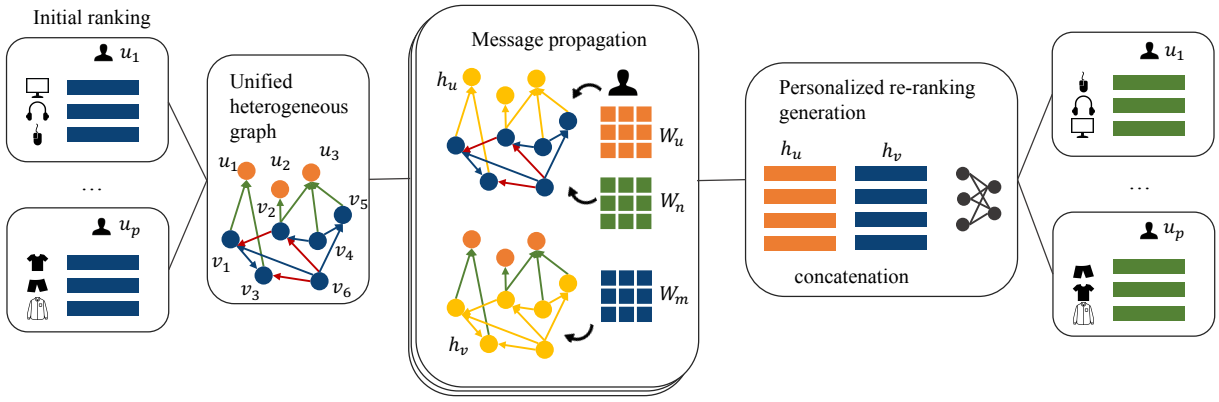



**Figure 4: The overall design of our proposed re-ranking model. We first build a unified heterogeneous graph, followed by multiple message propagation steps, which can be decomposed into a global item relationship propagation and a personalized intent propagation. After obtained the user/item representations, we deploy a feed-forward neural network to generate the personalized re-ranking list.**

## 3.1 IRGPR Framework

Figure 4 displays the overall design of our proposed IRGPR model. Given the initial ranking scores of candidate items for each user and the collected item relationships, we firstly construct a *unified heterogeneous graph* that consists of user and item nodes. As users and items have different semantic meanings in the graph, we design two featured propagation schemes, *i.e.*, *global item relationship propagation* and *personalized intent propagation*, for item nodes and user nodes, respectively. The awareness of different types of nodes allows IRGPR to better preserve useful information for re-ranking. By performing global item relationship propagation, the global transitive dependencies among the items across different ranked lists are encoded into the item representation. Personalized intent propagation extracts initial scoring information and personalized user intents from the neighborhoods. We then deploy a multi-layer perception and utilize a pair-wise loss function to estimate the re-ranking scores of the items for each user with personalization. Details are elaborated in the following sections.

## 3.2 Unified Heterogeneous Graph

For incorporating item relationships into personalized re-ranking, it is challenging to fuse two completely different sources of information, *i.e.*, initial ranking scores and item relationships, effectively to improve the results of re-ranking. Initial ranking scores quantify the user's preferences for each specific item, while item relationships indicate general transitive relation among items. To address the challenge, we propose to construct a unified heterogeneous graph with two types of nodes and multiple types of relations to fuse different sources of information, as shown in Figure 2. The constructed graph can be viewed as a combination of an item relationship graph and a user-item scoring graph.

**Item Relationship Graph.** Item relationships such as complementarity or substitutability explicitly show mutual influences among items. Based on the collected item relationships, we can construct an item relationship graph $\mathcal{G}_I$, with item feature vector $x_v \in \mathbb{R}^d$ for each item $v \in \mathcal{V}$, and edge feature vector $e_{v_i v_j} \in \mathbb{R}^c$ for the edge $(v_i, v_j)$ connecting item $v_i$ to item $v_j$. Edge feature $e_{v_i v_j}$ reflects the item relationships, *e.g.*, complementarity or substitutability. As item relationships are not necessarily symmetric, we model the relationships by a directed graph, which can be easily reduced to the undirected case.

**User-Item Scoring Graph.** To exploit the scoring information provided by the initial ranker, we build the user-item scoring graph $\mathcal{G}_S$, which involves two types of nodes, *i.e.*, user $u \in \mathcal{U}$ and item $v \in \mathcal{V}$. Each node is associated with a node feature vector, *i.e.*, $x_u \in \mathbb{R}^d$ and $x_v \in \mathbb{R}^d$. The edge connecting $v$ to $u$ indicates that $v$ is ranked at top positions by the initial ranker for user $u$, with the edge feature $e_{vu}$ being the initial ranking score. To be consistent with the item relationship graph $\mathcal{G}_I$, the user-item scoring $\mathcal{G}_S$ is also directed graph, which can be reduced to undirected graphs. Edges from items to users ensure information from related items can be aggregated to users.

Item relationships present certain intrinsic characteristics of the items which exist in the entire pool of items in the system. Thus, we propose a global item relationship propagation scheme to find item representations that encode such relationships. The learned item representations are shared by the users. On the other hand, users' preferences on the list of items varies for different individuals. As such, we propose a personalized propagation scheme with a novel intent embedding network to learn the personalized user intents.

## 3.3 Global Item Relationship Propagation

The global item relationship propagation is performed on the item relationship graph $\mathcal{G}_I$, aiming at learning better item representations to improve the performance of re-ranking. Through multiple iterations of information propagation on $\mathcal{G}_I$, item relational information propagates along the paths, which effectively imparts graph topological structures and high-order influences into the item representations. For flexibility in handling various kinds of relationships and edge features, rather than performing the propagation separately for each edge type as in [30, 36], we propose to

view item relationships as general edge feature vectors and embed such features into the propagation operation.

The proposed global item relationship propagation is centered on the *items*. Given an item $v_i$ and its feature vector $x_{v_i} \in \mathbb{R}^d$, we learn the item representation $h_{v_i} \in \mathbb{R}^d$ by recursively aggregating the information from its multi-hop neighborhood of items. Firstly, the initial representation $h_{v_i}^{(0)}$ is initialized by $x_{v_i}$. Then, the representation is updated by two steps, *i.e.*, a *message aggregation step* which is to capture the relational information from connected neighbors and a *message update step* which is to fuse the information from the neighbors and information of the item itself. As such, relational and transitive dependencies among items across different ranked lists are encoded into the item representation.

*3.3.1 Message Aggregation Step.* In our design, messages from neighbors carry both the information of the neighbor items and the information of the connecting edges. As linear transformation has been proven to be effective at accumulating and encoding features from local structured neighborhoods [30], we define the message from item $v_j$ to $v_i$ with a linear transformation:

$$m_{v_i \leftarrow v_j} = W_m h_{v_j} = f_m\left(e_{v_j v_i}, h_{v_j}\right) \cdot h_{v_j}, \tag{14}$$

where $m_{v_i \leftarrow v_j}$ is the message from item $v_j$ to item $v_i$, with a dimension of $d$. The nonlinear function $f_m(\cdot)$ embeds edge features and item features into a hidden continuous space, so that item transitive and relational influences can be captured. The function $f_m(\cdot)$ takes as input both the edge features $e_{v_j v_i}$ and the previous item representations of neighbor items $h_{v_j}$, and outputs a transformation matrix $W_m$ of dimension $d \times d$. Note that the usage of $f_m$ makes the propagation more flexible, allowing edge features to be of any dimensions and can be either categorical or continuous. Specifically, in our experiment, we concatenate $e_{v_j v_i}$ and $h_{v_j}$, then employ a Multi-Layer Perception (MLP) to learn the mapping. We reshape the output to a matrix of dimension $d \times d$:

$$f_m\left(e_{v_j v_i}, h_{v_j}\right) = \text{MLP}\left(e_{v_j v_i} || h_{v_j}\right), \tag{15}$$

where $||$ represents the concatenation operation.

Then, we aggregate messages from the item's neighborhood with a *Mean* aggregator that averages the messages from the neighbors:

$$m_{v_i} = \frac{1}{|\mathcal{N}_{v_i}|} \sum_{v_j \in \mathcal{N}_{v_i}} m_{v_i \leftarrow v_j}, \tag{16}$$

where $\mathcal{N}_{v_i}$ denotes the set of direct neighbors of item $v_i$, and $|\mathcal{N}_{v_i}|$ is the number of neighbors of item $v$. This *Mean* aggregator is similar to the convolutional propagation rule used in transductive GCN [21]. One could use some more complex aggregators such as Attention-based aggregator [33].

*3.3.2 Message Update Step.* After obtaining the messages $m_{v_i}$ from the neighborhood, we use the previous item representation of the center node and the newly received neighborhood messages to update the representation of the center node. We use a Gated Recurrent Unit (GRU) [6] in this update step, which has been proved to be effective in [22].

$$h_{v_i}^{'} = \text{GRU}(h_{v_i}, m_{v_i}). \tag{17}$$

## 3.4 Personalized Intent Propagation

The personalized intent propagation is proposed to leverage the scoring information given by the initial ranker to model personalized user intents. We perform similar propagation on the user-item scoring graph. Different from the global item relationship propagation, the center nodes are *users*, and the neighbors of users are *items*. Edges from the connected items representing the scores estimated by the initial ranker. Moreover, to explore users' implicit intents, we propose an *intent embedding network* to incorporate users' personalized interests into the propagation.

**Intent Embedding Network.** As different users may view the list of items from different perspectives. User features, *e.g.*, historical behaviors and profile features, implicitly reflect their personalized intents. We propose an intent embedding network to infer personalized intents by embedding user features into a $d \times d$ transformation matrix for each user $u$, so that the personalized intents can be propagated along the paths in the user-item scoring graph,

$$W_u = f_I(x_u), \tag{18}$$

where $f_I(\cdot)$ is a nonlinear mapping, *e.g.*, MLP. Therefore, $W_u \in \mathbb{R}^{d \times d}$ represents personalized user implicit intents. Similar to $W_m$ in Eq. (14), we design another mapping $f_n(e_{vu}) = \text{MLP}(e_{vu})$ to transform initial ranking scores, *i.e.*, $e_{vu}$, into a transformation matrix $W_n \in \mathbb{R}^{d \times d}$. Then, we combine the nonlinear mappings to aggregate the messages from the neighbors of user $u$,

$$m_u = \frac{1}{|\mathcal{N}_u|} \sum_{v \in \mathcal{N}_u} W_u W_n h_v \tag{19}$$

$$= \frac{1}{|\mathcal{N}_u|} \sum_{v \in \mathcal{N}_u} f_I(x_u) f_n(e_{vu}) h_v, \tag{20}$$

where $W_u$ is user-specific parameters, $\mathcal{N}_u$ is the set of direct neighbors of user $u$, and $|\mathcal{N}_u|$ is the number of neighbors. Basically, this message $m_u$ carries the information of the user's preferences towards items, *i.e.*, neighbors $\mathcal{N}_u$, in the initial ranked list where the item representations encode the global item relationships. Finally, the user representation is updated as follows $h_u^{'} = \text{GRU}(h_u, m_u)$.

## 3.5 Personalized Re-Ranking Generation

After obtaining the user and item representations with multiple propagation steps, we are ready to perform the *personalized* re-ranking. For a user $u$, we concatenate the user representation $h_u$ and an item representation $h_v$, $v \in \mathcal{V}$, and feed it to a feed-forward neural network to generate the re-ranking score for item $v$.

$$\hat{y}_{uv} = \sigma\left(\text{MLP}(h_u || h_v)\right), \tag{21}$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ is the sigmoid function. The user representation preserves personalized preferences aggregated from the designed unified heterogeneous graph, the generated results are thereby personalized.

We optimize IRGPR in an end-to-end manner with a widely adopted Bayesian Personalized Ranking (BPR) loss [28]. BPR loss is a pairwise loss that pushes the model to rank positive samples (interacted items) higher than negative samples (unobserved interactions).

$$\mathcal{L} = \sum_{(u, v_i, v_j) \in \mathcal{T}} \log \sigma\left(\hat{y}_{uv_i} - \hat{y}_{uv_j}\right), \tag{22}$$

**Table 1: Statistics of the Amazon data.**

| categories | #user | #item | #rating | density (1e-3) | #AB | #AV | #BT | #BV |
|---|---|---|---|---|---|---|---|---|
| Video Games | 2,390 | 48,938 | 148,420 | 1.27 | 1,143,763 | 170,107 | 27,460 | 117,400 |
| Musical Instruments | 565 | 65,150 | 31,806 | 0.86 | 531,379 | 480,710 | 26,955 | 117,902 |
| Movies & TV | 18,193 | 200,515 | 1,800,336 | 0.49 | 2,766,430 | 172,940 | 80,924 | 224,627 |
| Electronics | 16,187 | 424,116 | 847,556 | 0.12 | 2,550,227 | 2,823,653 | 126,166 | 769,868 |
| Clothing, Shoes, and Jewelry | 9,746 | 755,510 | 463,774 | 0.06 | 2,188,897 | 5,875,987 | 208,744 | 1,693 |

where $\mathcal{T}$ is the training set of positive-negative sample pairs. It is time-consuming to train GNNs on large-scale graphs [38]. Thus, we adopt an importance sampling strategy to sample neighbor nodes when performing information propagation. The importance factor is computed according to the degree of the nodes [16]. The model is trained in a mini-batch mode.

# 4 EXPERIMENTS

We conduct extensive experiments to verify the effectiveness of our proposed re-ranking model comparing with the state-of-the-art approaches.

## 4.1 Datasets

We use the e-commerce datasets from Amazon collected by McAuley *et al.* [23]. The complete dataset has over 1 million products and 42 million co-purchase relationships across around 20 top-level product categories. Four different types of item relationships are contained in the datasets:

- Also Bought (AB): Users bought $x$ also bought $y$ across sessions;
- Also Viewed (AV): Users viewed $x$ also viewed $y$;
- Bought Together (BT): Users frequently bought $x$ and $y$ ($x$ and $y$ were purchased as part of a single basket);
- Buy after Viewing (BV): Users who viewed $x$ eventually bought $y$.

According to the description of the datasets [23], AB and BT are referred to as being complementary, while AV and BV are being substitutable. We focus on five main categories of products, *i.e.*, Video Games, Musical Instruments, Movies & TV, Electronics, and Clothing, Shoes, and Jewelry that cover different areas of interest. The detailed statistics of the datasets are shown in Table 1. We remove the isolated users who have less than 30 interactions. Reviews are adopted as node features as in [27]. We consider implicit feedback rather than the explicit ratings, namely, all interacted user-item pairs will be used as positive samples. We randomly select 80% users and the corresponding ratings as the training set, 10% as the validation set, and the remaining 10% as the testing set.

## 4.2 Baselines

To show the effectiveness of re-ranking, we compare our proposed model with the baseline model that generates the initial ranked list. We also compare our proposed model with two state-of-the-art re-ranking models including an RNN-based model, *i.e.*, DLCM, and a Transformer-based model, *i.e.*, PRM.

- **DeepFM:** DeepFM [15] is a state-of-the-art point-wise ranking strategy for commercial use, which combines factorization machines and deep neural networks. In our implementation, the initial ranked list is generated by DeepFM.
- **DLCM:** DLCM is an RNN-based model, which employs a recurrent neural network to sequentially encode the top results using their feature vectors to learn local context features and to re-rank the top results [1].
- **PRM:** PRM is the state-of-the-art model for personalized re-ranking [26]. PRM exploits a Transformer structure to encode the mutual information among all items in the lists.

## 4.3 Evaluation Metrics

We adopt Precision@$k$ and MAP@$k$ to evaluate the performance of different methods, with $k \in \{5, 10, 20\}$. These metrics are also adopted by related work [1, 26].

- **Precision@$k$:** Precision@$k$ is defined as the fraction of clicked items in the top-$k$ recommended items for all test samples:

$$\text{Precision@}k = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{\sum_{i=1}^{k} \mathbb{1}\{S_u(i)\}}{k}, \tag{23}$$

  where $\mathcal{U}$ is the set of all users, $S_u$ is the re-ranked list for user $u$, $S_u(i)$ refers to the $i$-th item in the list, and $\mathbb{1}\{\cdot\}$ is the indicator function, which equals to 1 if item $i$ is positively interacted, and 0 otherwise.
- **MAP@$k$:** MAP@$k$ is the mean average precision at cutoff $k$, which is defined by:

$$\text{MAP@}k = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{\sum_{i=1}^{k} \text{Precision@}k * \mathbb{1}\{S_u(i)\}}{k}. \tag{24}$$

## 4.4 Experimental Settings

We use the grid search to select hyper-parameters for all the methods on the validation set. The details are as follows: the node embedding size is chosen from $\{4, 8, 16, 32\}$, learning rate is in $\{0.001, 0.01, 0.1\}$, the batch size is in $\{16, 64, 128, 512\}$, and the optimization method is Adam [20] or Adagrad [9]. For our proposed IRGPR, the message propagation iterations $L$ is from $\{2, 3, 4, 5\}$, the number of sampled neighbors $s$ is in $\{3, 5, 10, 15\}$. All models are trained for 300 epochs to ensure convergence and we use early stopping. For a fair comparison, the input node features $x_v$ are the same for all the compared models which are reviews of products. Reviews are encoded using a doc2vec technique as in [27], followed by a fully-connected embedding layer mapping raw node features from the original dimension to the dimension of the node embedding size $d$. The dimension of the edge feature vector $c = 4$ for four different types of item relationships.

## 4.5 Expermental Results and Analysis

In this section, we present our experimental results toward answering the following experimental research questions (RQs):

**Table 2: Experimental results on Amazon data. DeepFM is the initial ranker.**

| | | Precision@5 | MAP@5 | Precision@10 | MAP@10 | Precision@20 | MAP@20 |
|---|---|---|---|---|---|---|---|
| Video Games | DeepFM | 0.7506 | 0.8137 | 0.7494 | 0.7983 | 0.6967 | 0.7774 |
| | DLCM | 0.7554 | 0.8238 | 0.7476 | 0.8047 | 0.6923 | 0.7812 |
| | PRM | 0.7651 | 0.8310 | 0.7561 | 0.8081 | 0.6795 | 0.7842 |
| | **IRGPR** | **0.8241**$^*$ | **0.8956**$^*$ | **0.7855**$^*$ | **0.8584**$^*$ | **0.7019**$^*$ | **0.8169** |
| | imp.% | +7.71% | +7.77% | +3.89% | +6.22% | +0.75% | +4.17% |
| Musical Instruments | DeepFM | 0.6056 | 0.7405 | 0.5111 | 0.6893 | 0.5089 | 0.5984 |
| | DLCM | 0.6111 | 0.7517 | 0.5528 | 0.6957 | 0.4931 | 0.6201 |
| | PRM | 0.6233 | 0.7750 | 0.5472 | 0.7152 | 0.5028 | 0.6169 |
| | **IRGPR** | **0.6751**$^*$ | **0.8285**$^*$ | **0.5573**$^*$ | **0.7607**$^*$ | **0.5101**$^*$ | **0.7364**$^*$ |
| | imp.% | +8.31% | +6.90% | +0.81% | +6.36% | +0.24% | +18.76% |
| Movies & TV | DeepFM | 0.7398 | 0.8692 | 0.6724 | 0.8102 | 0.6008 | 0.7419 |
| | DLCM | 0.7745 | 0.8428 | 0.7752 | 0.8239 | 0.6493 | 0.8098 |
| | PRM | 0.8077 | 0.8841 | 0.7544 | 0.8577 | 0.6596 | 0.8216 |
| | **IRGPR** | **0.8300**$^*$ | **0.8945**$^*$ | **0.7862**$^*$ | **0.8664**$^*$ | **0.6859**$^*$ | **0.8239** |
| | imp.% | +2.76% | +1.18% | +1.42% | +1.01% | +3.99% | +0.28% |
| Electronics | DeepFM | 0.8068 | 0.9349 | 0.6925 | 0.8675 | 0.5947 | 0.7795 |
| | DLCM | 0.8266 | 0.8984 | 0.7726 | 0.8685 | 0.6311 | 0.7875 |
| | PRM | 0.8261 | 0.9185 | 0.7897 | 0.8705 | **0.6490** | 0.8233 |
| | **IRGPR** | **0.8776**$^*$ | **0.9386**$^*$ | **0.8031**$^*$ | **0.9026**$^*$ | 0.6472 | **0.8481**$^*$ |
| | imp.% | +6.17% | +0.40% | +1.70% | +3.69% | -0.28% | +3.01% |
| Clothing, Shoes, and Jewelry | DeepFM | 0.5970 | 0.7859 | 0.5619 | 0.7071 | 0.5281 | 0.6427 |
| | DLCM | 0.6872 | 0.7915 | 0.6087 | 0.7463 | 0.5402 | 0.6761 |
| | PRM | 0.6811 | 0.7989 | 0.6358 | 0.7546 | **0.5748** | 0.7000 |
| | **IRGPR** | **0.7057**$^*$ | **0.8426**$^*$ | **0.6381**$^*$ | **0.7878**$^*$ | 0.5628 | **0.7176**$^*$ |
| | imp.% | +2.69% | +5.47% | +0.36% | +4.40% | -2.09% | +2.51% |

We conduct a two-sided significant test between the proposed IRGPR and the strongest baseline, where * means the p-value is smaller than 0.05. imp.% computes the improvement achieved by IRGPR over the strongest baseline.

- **RQ1:** How well does IRGPR perform comparing to the state-of-the-art models for re-ranking?
- **RQ2:** How does the modeling of item relationships, personalization, and the initial ranker affect the performance of IRGPR?
- **RQ3:** What is the impact of hyper-parameters on IRGPR?
- **RQ4:** Finally, is IRGPR capable of capturing meaningful patterns from item relationships? What is the difference between the initial ranked list and the re-ranked list?

*4.5.1 RQ1: Overall Performance.* Table 2 reports the overall comparison of our proposed IRGPR and baselines on five Amazon datasets. Bold numbers are the best results and underlined numbers are the results of the strongest baselines. The initial ranking scores are generated by DeepFM which is a widely adopted deep neural network based method with point-wise loss in commercial recommender systems. From Table 2, we have several important observations.

*Firstly, incorporating item mutual influences improves the quality of re-ranking.* We can observe that DLCM and PRM which explicitly model mutual influences between items outperform DeepFM in most cases on all the datasets, which validates the positive effect of modeling mutual influences for re-ranking. Though DeepFM has been widely adopted by commercial recommender systems, it assumes that items are independent in a list which is not true. Thus, the performance of the initial ranker can be improved. Yet, the two challenges of re-ranking remain unresolved in these methods.

*Secondly, the proposed IRGPR consistently outperforms other baselines on all the datasets under various metrics.* For example, the proposed IRGPR improves the strongest baseline by 7.71%, 8.31%, 2.76%, 6.17%, and 2.69% in terms of precision@5 on the five datasets, respectively. DLCM exploits RNN to capture sequential patterns in the ranked lists, while PRM uses self-attention architecture to consider influences between pairs of items. Both of them have limitations as we discussed in Section 1. Moreover, DLCM and PRM are not able to distinguish different types of influences, such as complementarity and substitutability. The results demonstrate the effectiveness of the designed global item relationship propagation and personalized intent propagation. The two-sided significant test shows the improvements are statistically significant.

*4.5.2 RQ2: Ablation Study.* In this section, we perform an ablation study to verify the importance of different components in IRGPR for re-ranking. IRGPR is composed of several designed components such as the modeling of item relationships, personalization, as well as the methods to realize the personalization like the linear mappings, and the intent embedding network. In Table 3, we present the performance of several variants of IRGPR by removing one particular component of IRGPR. We report experimental results in terms of Precision@10 (P@10) and MAP@10 on the densest and sparsest dataset, *Video Games* and *Clothing, Shoes, and Jewelry*, as examples.

**-Item Relationships:** In this experiment, we remove all item relationships from the graph. As such, the proposed heterogeneous graph is reduced to a bipartite graph with edges connecting users and items indicating the initial ranking scores. The impact of deleting item relationships can be easily noticed. Precision@10 drops by 5.36% and 2.13% on Video Games dataset and Clothing, Shoes, and Jewelry dataset, respectively, proving that item relationships play a critical role in determining the quality of re-ranking.

**-Personalization:** In this experiment, we remove the user representation in the personalized re-ranking generation, aiming to validate the usage of personalization for re-ranking,

$$\hat{y}_{uv} = \sigma\left(\text{MLP}(h_v)\right),$$

**Table 3: Ablation study of designed components for IRGPR.**

| | Video Games | | Clothing, Shoes, and Jewelry | |
|---|---|---|---|---|
| | P@10 | MAP@10 | P@10 | MAP@10 |
| **IRGPR** | **0.7855** | **0.8584** | **0.6381** | **0.7878** |
| -Item Relationships | 0.7434<br>-5.36% | 0.8416<br>-1.96% | 0.6245<br>-2.13% | 0.7692<br>-2.36% |
| -Personalization | 0.7357<br>-6.34% | 0.8202<br>-4.45% | 0.5834<br>-8.57% | 0.7413<br>-5.90% |
| -Intent Embedding Network | 0.756<br>-3.76% | 0.8439<br>-1.69% | 0.6226<br>-2.43% | 0.7865<br>-0.17% |
| -Linear Mappings | 0.7489<br>-4.66% | 0.84<br>-2.14% | 0.6166<br>-3.37% | 0.7736<br>-1.80% |

**Table 4: IRGPR with different initial rankers.**

| | | Video Games | | Clothing, Shoes, and Jewelry | |
|---|---|---|---|---|---|
| Initial Ranker | Re-ranking | P@10 | MAP@10 | P@10 | MAP@10 |
| DeepFM | NULL | 0.7494 | 0.7983 | 0.5619 | 0.7071 |
| | IRGPR | 0.7855 | 0.8584 | 0.6381 | 0.7878 |
| | imp.% | 4.82% | 7.53% | 13.56% | 11.41% |
| SVMRank | NULL | 0.5994 | 0.6255 | 0.4936 | 0.6116 |
| | IRGPR | 0.7831 | 0.8605 | 0.6419 | 0.7824 |
| | imp.% | 30.65% | 37.57% | 30.05% | 27.92% |
| LambdaMart | NULL | 0.6994 | 0.7390 | 0.5487 | 0.6596 |
| | IRGPR | 0.7837 | 0.8577 | 0.6392 | 0.784 |
| | imp.% | 12.05% | 16.06% | 16.50% | 18.86% |

where all users share the same item representations. Results show a large drop in all metrics across datasets by entirely removing the personalization mechanism. These results provide direct evidence on the importance of incorporating user personalized preferences in re-ranking.

**-Intent Embedding Network:** We remove the intent embedding network $f_I(x_u)$ from Eq.(20) to study the importance of explicitly modeling user intents in re-ranking, where $m_u$ reduces to

$$m_u = \frac{1}{|\mathcal{N}_u|} \sum_{v \in \mathcal{N}_u} f_n(e_{uv}) h_v.$$

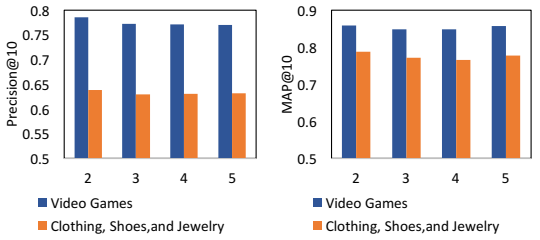
We observe that the accuracy drops on both datasets. For example, Precision@10 decreases by 3.76% on Video Games dataset. The results show our proposed IRGPR benefits from the incorporation of the intent embedding network.

**-Linear Mappings:** For personalized intent propagation, we then replace the composition of linear maps in Eq.(19) by an MLP with a linear concatenation of the user feature and ratings as input,

$$m_u = \frac{1}{|\mathcal{N}_u|} \sum_{v \in \mathcal{N}_u} \text{MLP}(x_u || e_{uv}) h_v,$$

where $||$ is the concatenation operation and $\text{MLP}(\cdot)$ is a feed-forward network that outputs the transformation matrix of size $d \times d$. Results in Table 3 illustrate that matrix multiplication may have better expressive power than a linear concatenation in re-ranking.

**Initial Ranker.** Next, we change the initial ranker and study how the quality of the initial ranker influences the performance of IRGPR. Apart from the point-wise model DeepFM, we implement two representative pair-wise and list-wise ranking models: (i) **SVMRank** is a variant of the support vector machine algorithm and use the pair-wise loss to model the scoring function [18]; (ii) **LambdaMart** combines LambdaRank and Multiple Additive Regression Trees (MART) with a list-wise loss to model the ranking function [4]. The results in Table 4 indicate that the change of initial ranker has a minor impact on the performance of our proposed IRGPR. Although the ranking quality of SVMRank and LambdaMart is not as good as DeepFM due to limited expressive ability, our proposed IRGPR has the capability of refining and improving the initial rankers by a great margin.




**(a) Precision@10** **(b) MAP@10**

**Figure 5: Impact of the number of aggregation iteration $L$ of IRGPR.**

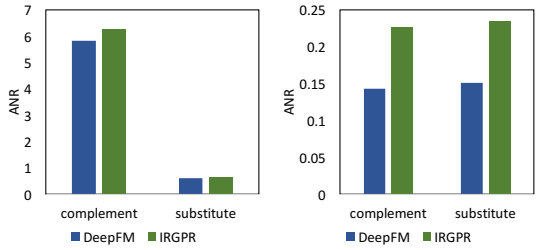

**(a) Precision@10** **(b) MAP@10**

**Figure 6: Impact of the number of sampled neighbors $s$ of IRGPR.**

*4.5.3 RQ3: Impact of Hyper-Parameters.* We focus here on two representative hyper-parameters of our proposed IRGPR to discuss their impact on the performance, namely, the number of iterations $L$ of each propagation and the number of sampled neighbors $s$ during the propagation.

**The number of iterations $L$ of propagation.** The number of propagation iterations corresponds to the maximal number of hops of the neighbors in the graph. We report Precision@10 and MAP@10 of IRGPR on Video Games dataset and Clothing, Shoes, and Jewelry dataset in Figure 5 by varying the number of propagation iterations. We observe that the number of 2 is sufficient to achieve the best performance. The accuracy does not improve significantly with a larger number of iterations. It is probably due to the over-smoothing problem which makes the learned representations indistinguishable by performing too many propagation iterations [5].

**The number of sampled neighbors $s$.** Next, we vary the sample size of the neighborhood $s$. Figure 6 demonstrates that the best results are obtained when $s = 5$, indicating that the performance of IRGPR benefits from a suitable size of sampled neighbors. For example, IRGPR achieves a Precision@10 of 0.7855 and 0.6381 on




**(a) Video Games** **(b) Clothing, Shoes, and Jewelry**

**Figure 7: ANR of DeepFM and IRGPR.**

Video Games and Clothing, Shoes, and Jewelry, respectively. Figure 6 suggests that IRGPR suffers from a small sampled size due to the loss in the neighborhood information. Moreover, it is also worth noting that a large size may involve noisy neighbors and degrade the performance.

*4.5.4 RQ4: Evidence of the Importance of Item Relationships for Re-ranking.* To investigate how the re-ranking quality relates to item relationships, we compute the Average Number of different item Relationships (ANR) occurred at the top positions of the ranked lists of DeepFM and IRGPR in Figure 7, which is defined by

$$\text{ANR}_r@10 = \frac{1}{|\mathcal{D}|} \sum_{a \in \mathcal{D}} N_r(a),$$

where $\mathcal{D}$ is the lists of items in the test data, $|\mathcal{D}|$ is the number of lists in $\mathcal{D}$, $N_r(a)$ is the number of a specific relationship $r$ (complementary or substitutable) among the top-10 items in list $a$. As discussed in Section 4.5.1, IRGPR significantly outperforms the initial ranker DeepFM. We analyze how the item relationships contribute to the large improvement of performance.

Figure 7 shows that the re-ranking lists generated by IRGPR contain more complementary and substitutable relationships compared to the initial ranker. In particular, ANR for complementary relationships of IRGPR is increased by 7.65% and 57.78% on Video Games, and Clothing, and Shoes, and Jewelry, respectively. While ANR for substitutable relationships is increased by 2.94% and 55%, respectively. This illustrates that understanding item relationships is generally beneficial to re-ranking, although item relationships are sparse and most of the items are unrelated. For example, for complementary items, users may be interested in a list displaying an aesthetically matching shirt and pants outfit together. As for substitutable items, presenting substitutable items together, like two pairs of headphones with different prices, may promote the possibility of purchasing for the one that is more cost-effective.

# 5 RELATED WORK

This work draws on the following research areas: (1) re-ranking and (2) graph neural networks.

## 5.1 Re-ranking Algorithms

Several existing works have demonstrated the effectiveness of improving recommendation results with re-ranking models [1, 26, 39] or by fine-tuning the initial ranker [17]. For example, Ai *et al.* [1] use a recurrent neural network to sequentially encode the top-ranked items with their feature vectors, and learn a local context model to re-rank the top results. Similarly, Zhuang *et al.* [39] design a recurrent neural network with an attention mechanism to capture long-distance influences between items.

However, these approaches based on recurrent neural networks have constrained expression ability to model the mutual influences between items in the list. In particular, they are not able to explicitly relate two items in a list, since all the items in a list are sequentially aggregated together by recurrent neural networks.

Inspired by the Transformer architecture [33] used in machine translation, Pei *et al.* [26] propose to use the Transformer to model the mutual influences between items. The Transformer structure uses the self-attention mechanism where the relationship between any two items is reflected with an attention score. However, the self-attention mechanism assumes that any two items in a list have mutual influences, which is probably not true. Moreover, the self-attention mechanism is not able to distinguish different types of item relationships which can be modeled by our work.

Other related work, such as Jin *et al.* [17], proposes a pairwise approach that considers the problem of refining the initial ranker for semi-supervised learning. Wang *et al.* [35] discusses correlation between pairs of documents for information retrieval.

Pei *et al.* [26] is most related to our work. However, our work models the mutual influences in a more explicit and expressive way by incorporating item relationships and learning multi-hop relational dependencies among users and items. Personalized intents of users are also captured by our proposed model.

## 5.2 Graph Neural Networks

Many recent research efforts have demonstrated the power of Graph Neural Networks (GNN) to model graph-structured data [3, 7, 8, 12]. For example, Graph Convolutional Neural Networks (GCN) have achieved the state-of-the-art in node representation for signed graphs [8], sentence classification [19] and image recognition [12]. Recently, Morris *et al.* [25] build a k-dimensional GNN which can consider higher-order graph structures at multiple scales. Bresson *et al.* [3] propose a residual graph neural network (ConvNets) which brings a significant improvement in subgraph matching and graph clustering, as they illustrated, residuality could better learn multi-layer architectures in complex graph-structured data.

With the growth of the study of GNN, it is widely applied for recommendation, including social recommendation [11, 31, 34] and knowledge graph based recommendation [14, 32]. It is even adapted to traditional collaborative filtering recommendation methods [2, 36]. Recently, Fan *et al.* [11] provide a principled GNN approach incorporating social connections and user purchase history to capture the interactions between user-items for recommendation. Song *et al.* [31] use a dynamic graph attention network and incorporated recurrent neural networks for modeling user behaviors in session-based social recommendation. Grad-Gyenge *et al.* [14] build a graph embedding method that took advantage of the knowledge graph to map users toward items for recommendation. Ying *et al.* combine efficient random walks and graph convolutions for web-scale recommendation [37]. Considering the user-item interaction, Wang *et al.* [36] construct a user-item interaction bipartite graph and proposed a graph-based collaborative filtering method to capture higher-order connectivity in the user-item interactions. A

metapath-guided GNN is proposed for search intent recommendation [10]. However, few of these approaches consider graph neural networks for personalized re-ranking.

## 6 CONCLUSION

In this work, we propose a novel GNN-based model, IRGPR, for personalized re-ranking. We highlight the significance of incorporating item relationships, such as complements or substitutes, to model the mutual influences between items in re-ranking. To overcome the difficulties of fusing initial scoring information and item relationships, we build a unified heterogeneous graph. We propose a global item relationship propagation to exploit the global topological structure and to capture the relational item dependencies, as well as a personalized intent propagation to explicitly learn user personalized intentions. Extensive experiments on large real-world datasets demonstrate the rationality and effectiveness of IRGPR. Besides, further insights are provided on unifying the state-of-the-art re-ranking models into the GNN framework.

## REFERENCES


[1] Qingyao Ai, Keping Bi, Jiafeng Guo, and W Bruce Croft. 2018. Learning a deep listwise context model for ranking refinement. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. 135–144.

[2] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).

[3] Xavier Bresson and Thomas Laurent. 2017. Residual gated graph convnets. *arXiv preprint arXiv:1711.07553* (2017).

[4] Christopher JC Burges. 2010. From ranknet to lambdarank to lambdamart: An overview. *Learning* 11, 23-581 (2010), 81.

[5] Deli Chen, Yankai Lin, Wei Li, Peng Li, Jie Zhou, and Xu Sun. 2020. Measuring and Relieving the Over-Smoothing Problem for Graph Neural Networks from the Topological View. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence*. 3438–3445.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 1724–1734.

[7] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*. 3844–3852.

[8] Tyler Derr, Yao Ma, and Jiliang Tang. 2018. Signed graph convolutional networks. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 929–934.

[9] John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research* 12, Jul (2011), 2121–2159.

[10] 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. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2478–2486.

[11] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *The World Wide Web Conference*. ACM, 417–426.

[12] Yingruo Fan, Jacqueline CK Lam, and Victor OK Li. 2018. Multi-region ensemble convolutional neural network for facial expression recognition. In *International Conference on Artificial Neural Networks*. Springer, 84–94.

[13] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 1263–1272.

[14] László Grad-Gyenge, Attila Kiss, and Peter Filzmoser. 2017. Graph embedding based recommendation techniques on the knowledge graph. In *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization*. ACM, 354–359.

[15] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*. 1725–1731.

[16] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[17] Rong Jin, Hamed Valizadegan, and Hang Li. 2008. Ranking refinement and its application to information retrieval. In *Proceedings of the 17th international conference on World Wide Web*. 397–406.

[18] Thorsten Joachims. 2006. Training linear SVMs in linear time. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2006*. ACM, 217–226.

[19] Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. 1746–1751.

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*.

[21] Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *5th International Conference on Learning Representations* (2017).

[22] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. 2016. Gated graph sequence neural networks. In *4th International Conference on Learning Representations*.

[23] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*. ACM, 785–794.

[24] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.

[25] Christopher Morris, Martin Ritzert, Matthias Fey, William L Hamilton, Jan Eric Lenssen, Gaurav Rattan, and Martin Grohe. 2019. Weisfeiler and leman go neural: Higher-order graph neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 4602–4609.

[26] Changhua Pei, Yi Zhang, Yongfeng Zhang, Fei Sun, Xiao Lin, Hanxiao Sun, Jian Wu, Peng Jiang, Junfeng Ge, Wenwu Ou, et al. 2019. Personalized re-ranking for recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*. 3–11.

[27] Vineeth Rakesh, Suhang Wang, Kai Shu, and Huan Liu. 2019. Linked variational autoencoders for inferring substitutable and supplementary items. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 438–446.

[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*. 452–461.

[29] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. 2008. The graph neural network model. *IEEE Transactions on Neural Networks* 20, 1 (2008), 61–80.

[30] 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 *European Semantic Web Conference*. Springer, 593–607.

[31] Weiping Song, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. 2019. Session-based social recommendation via dynamic graph attention networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 555–563.

[32] John K Tarus, Zhendong Niu, and Ghulam Mustafa. 2018. Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. *Artificial Intelligence Review* 50, 1 (2018), 21–48.

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. 5998–6008.

[34] Jianling Wang, Kaize Ding, Ziwei Zhu, Yin Zhang, and James Caverlee. 2020. Key Opinion Leaders in Recommendation Systems: Opinion Elicitation and Diffusion. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 636–644.

[35] Jun Wang and Jianhan Zhu. 2009. Portfolio theory of information retrieval. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. 115–122.

[36] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.

[37] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 974–983.

[38] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2018. Graph neural networks: A review of methods and applications. *arXiv preprint arXiv:1812.08434* (2018).

[39] Tao Zhuang, Wenwu Ou, and Zhirong Wang. 2018. Globally optimized mutual influence aware ranking in e-commerce search. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*.