Relation-Enhanced Multi-Graph Attention Network for Recommendation

Yu Fan, Lei Zhang, Pengfei Wang
School of Computer Science
Beijing University of Posts and Telecommunications
Beijing, China
fanyubupt@gmail.com, {zlei, wangpengfei}@bupt.edu.cn

***Abstract*—Knowledge graph captures structured information and relations between a set of entities. Researchers always introduce knowledge graph (KG) into recommender systems for more accurate and explainable recommendation. Recently, many researchers deploy Graph Neural Network (GNN) with knowledge graph in recommender systems. However, they do not consider proper aggregation and ignore the layer limitation of the GNN. To tackle these issues, we propose a novel recommendation framework, named Relation-Enhanced Multiple Graph Attention Network (REMAN for short), which models the heterogeneous and high-order relationships among entities in recommendation. Firstly, we encode user behaviors and item knowledge as a unified relational graph. Then we utilize a relation-specific attention aggregator to aggregate the embeddings of the heterogeneous neighbors. Thirdly, we propose a relation-enhanced user graph in order to make up for the limitations of the GNN layer in recommendation. Finally, we make prediction based on the embeddings we learned in graphs. Extensive experiments on three benchmark datasets demonstrate that our framework significantly outperforms strong recommender methods.**

***Index Terms*—Recommender Systems; Graph Neural Network; Embedding Propagation; Knowledge Graph**

I. Introduction

Recommender systems, which aim to help users find potentially interested items, are playing significant roles in many web sites (e.g., Amazon [1], YouTube [2]) to overcome the information overload problem. A traditional recommendation technique is collaborative filtering (CF) [3], [4], which usually suffers from the sparsity of user-item interactions. Researchers then introduce additional sources of information to address sparsity issue. Some researchers simply using attributes by transforming the information into a generic feature vector, and feeding it into a supervised learning model to predict the score [5], [6]. But these methods ignore the relation between item information, so researchers introduce the knowledge graph (KG) to capture structured data and relations between a set of entities.

KG are heterogeneous graphs in which nodes correspond to entities (e.g., items or products, as well as their properties and characteristics) and edges correspond to relations. KG provide connectivity information between items via different types of relations and thus capture semantic relatedness between the items. In general, existing KG-aware recommender systems models can be categorized into three types: Embedding-based models [7]–[9], path-based models [10]–[12] and hybrid models [13]–[16]. Due to the limitations of the first two methods, many researchers propose the hybrid methods based on the above two methods which learn user/item embeddings by exploiting the structure of KG.

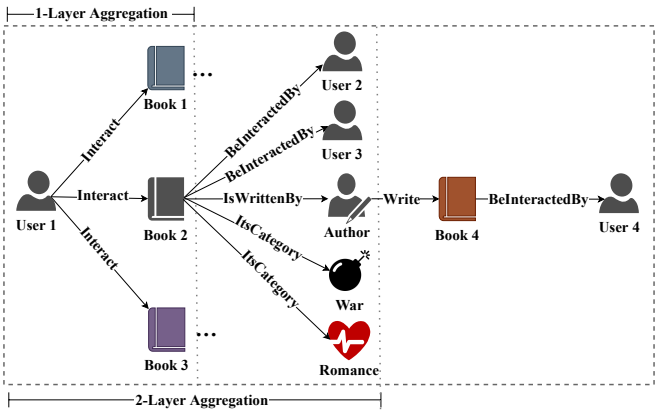

Fig. 1. A toy example of recommendation graph which contains users, books, categories, author as graph nodes.

Recently, the hybrid propagation-based method which introduce GNN to recommendation shows its effectiveness. But their models are not fully adapted to the recommendation. To see this, we present an illustrative example in Fig.1. The picture shows a graph composed of users, products (i.e., books) and attributes and their relations. The neighbor aggregation as Fig.1 is widely used by hybrid models. We take the neighbor aggregation of $User1$ as an example, there are two main problems: (1) In the 2-layer aggregation, a variety of different entities are aggregated. Due to the characteristics of product recommendations, the meaning of entities with different relationships also varies greatly, such as "$User2$" and "$Romance$". But previous methods apply attentive neighborhood aggregation without considering the differences of type between adjacent nodes in Recommendation. (2) There is a limitation of the GNN layer for recommendation. For example, we have a path between "$User1$" and "$User4$" where its length is 4 in Fig.1. But in GNN models for recommendation, 4-layer aggregation will introduce irrelevant information and affect the recommendation result. Therefore the important meta-path information will be ignored causing the effect of

978-1-7281-6579-0/20/$31.00©2020 IEEE

layer number on the GNN performance.

To remedy these problems, we propose a **R**elation-**E**nhanced **M**ultiple Graph **A**ttention **N**etwork (REMAN for short), which models the heterogeneous and high-order relationships among entities in recommendation. Firstly, we encode user behaviors and item knowledge as a unified relational graph. Secondly, we utilize a relation-specific attention aggregator to aggregate the embeddings of the heterogeneous neighbors. Thirdly, we propose a relation-enhanced user graph in order to make up for the shortcomings of previous methods when mining the users' personal preferences and the limitations of the GNN layer for recommendation. Finally, we make prediction based on the embeddings we learned in graphs. By doing so, our model is able to capture relation-aware high-order connectivity and introduce user profile information. Extensive experiments on three benchmark datasets demonstrate that our framework significantly outperforms strong recommender methods.

II. Related Work

In this section we provide a brief overview on two related research areas, which are knowledge-aware recommendation and graph neural networks respectively.

A. Knowledge-Aware Recommendation

Traditional recommender systems that are based on collaborative filtering (CF), which models user-item interactions by inner product or neural networks such as BPRMF [17]. Then researchers introduce the knowledge graph (KG) to capture structured data and relations between a set of entities.

KGs are heterogeneous graphs which provide connectivity information between items via different types of relations and thus capture semantic relatedness between the items. In general, existing KG-aware recommender systems can be categorized into three types: Embedding-based methods [7]–[9], path-based methods [10]–[12] and hybrid methods [13]–[16].

Embedding-based methods pre-process a KG with knowledge graph embedding (KGE) algorithms which model rigorous semantic relatedness (e.g., TransE [18] assumes head +relation = tail), such as CKE [8] model KG completion by TransR and recommendation with shared item embeddings. But these methods lack high-order modeling and have poor scalability. Path-based methods [10]–[12] exploit the KG structure by design meta-path patterns or path selection algorithms to extract latent features in heterogeneous knowledge graph. But path selection has a large impact on the which relies heavily on manually designed meta-paths/meta-graphs. Many researchers have proposed the hybrid methods based on the above two methods. RippleNet [13] is a memory-network-like model that propagates users' potential preferences in the KG and explores their hierarchical interests. Recently, the hybrid propagation-based methods are utilized to combine GNN and KG for recommendation. For example, the KGNN-LS model [14] applies GCN on the user-specific graph to learn item embedding. In KGAT [16], the graph attention mechanism is adopted to aggregate and propagate local neighborhood information of an entity, without considering users' personalized preferences on entities. On summary, these hybrid propagation-based methods implicitly aggregate the high-order neighborhood information via layer by layer propagation. Our proposed model can be seen as an instance of hybrid propagation-based methods.

B. Graph Neural Networks

Recently, Graph Neural Networks [19]–[21] have achieved appealing performance in various application by modeling both the edges and the node attributes simultaneously. For example, Graph Convolutional Networks (GCN) [19] presents an efficient layer-wise propagation rule based on the first-order approximation of the spectral convolutions on graphs. It achieves outstanding performance for semi-supervised classification on the graph-structured data. GAT [20] leverages masked self-attentional layers to calculate the hidden representations of each node in the graph by attending over its neighbors. GraphSage sampling a fixed-size set of neighbors as the support size. PinSage [22] provides a data-efficient GCN algorithm based on GraphSAGE [23] for effective Web-scale recommender systems. Several recent efforts have attempted to leverage Heterogeneous Graph Neural Networks [14], [24], [25] and Knowledge Graph [14], [16] for recommendations. KGNN-LS [14] extended GCN to the knowledge graph by aggregating the neighborhood information selectively and biasedly. KGAT [16] combined knowledge graph embedding with graph attention networks to model both the user-item interactions and the knowledge graph for recommendations. Though these two methods can model the knowledge graph information, they simply employ GNN to predict items and did not fully consider designing for the recommended scenarios.

III. Task Formulation

In this section, we first introduce CKG based on the heterogeneous graph, then we introduce relation-enhanced user graph. Finally, the task description is given.

Heterogeneous Graph. A heterogeneous graph, donated as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$, where $\mathcal{V}$, $\mathcal{E}$, $\mathcal{R}$ are the set of nodes, edges and relations respectively. We define the edge from node h to node t by relation r is constructed by triplet (h, r, t). So we define $\mathcal{E} = \{(h, r, t)|h, t \in \mathcal{V}, r \in \mathcal{R}\}$.

Collaborative Knowledge Graph. We construct collaborative knowledge graph (CKG), which encodes user behaviors and item knowledge as a unified relational graph. We define CKG as $\mathcal{G}_{CK} = (\mathcal{V}_{CK}, \mathcal{E}_{CK}, \mathcal{R}_{CK})$, and $\mathcal{V}_{CK} = \mathcal{U} \bigcup \mathcal{I} \bigcup \mathcal{A}$ where $\mathcal{U}, \mathcal{I}, \mathcal{A}$ are the set of users, items and attributes respectively.

(1) For each user behavior, we represent it as (u, y_{ui}, i), where $u \in \mathcal{U}, i \in \mathcal{I}$ and $y_{ui} = Interact = 1$ is a relation between user u and item i. (2) For each item-attribute edge in the knowledge graph, we represent it as (h, r, t) where $h \in \mathcal{I}, t \in \mathcal{A}$. For example, *(Harry Potter, WrittenBy, J.K. Rowling)* states the fact that J.K. Rowling

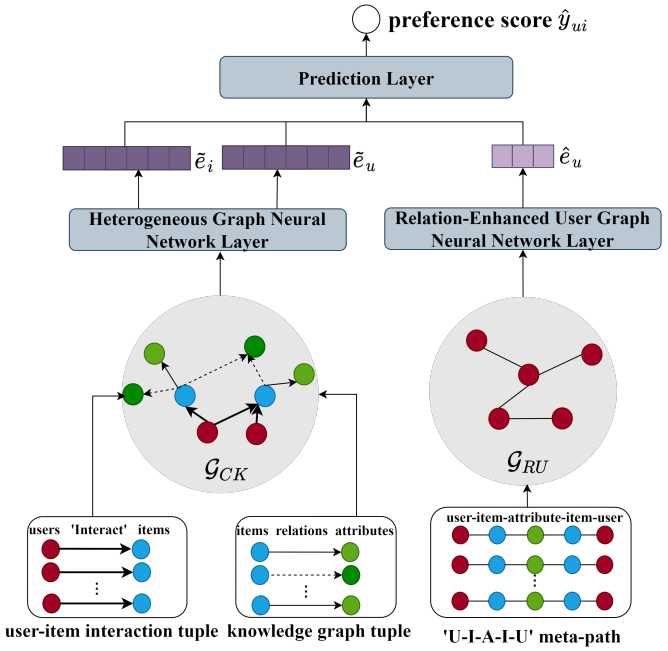

Fig. 2. The overall architecture of Relation-Enhanced Multiple Graph Attention Network (REMAN). For simplicity, we have omitted reverse relations in $\mathcal{G}_{CK}$.

wrote Harry Potter. Based on the item-entity alignment set, we seamlessly integrate the user-item graph with KG as a heterogeneous graph. (3) In addition, for each tuple (h, r, t) in the $\mathcal{G}_{CK}$, we define a corresponding tuple $(t, -r, h)$. For example, the tuples $(User1, Interact, Book1)$ and $(Book1, BeInteractedBy, User1)$ mean the user interact with the book while the book is interacted by the user. The tuples $(Book2, IsWrittenBy, Author)$ and $(Author, Write, Book2)$ are also the meaning of relation and reverse relation. We construct $\mathcal{G}_{CK}$ through the above definitions.

Relation-Enhanced User Graph. Due to limitations of layer length, we lack the ability to explore long-path information. Besides, In the $\mathcal{G}_{CK}$, the item neighbor aggregated information is more abundant than the user neighbor aggregated, because only items directly connected to the user in one-hop for users. Therefore, insufficient exploration of user information in the aggregation of $\mathcal{G}_{CK}$ results in the deficiency of modeling users' personal preference.

To solve these problems, we apply the conventional method of constructing meta-path graph to our user graph. We design the relation-enhanced user graph based on metapath "user-item-attribute-item-user", and represent it as $\mathcal{G}_{RU} = (\mathcal{V}_{RU}, \mathcal{E}_{RU})$ where $\mathcal{E}_{RU} = \{(h, r_u, t)|h, t \in \mathcal{U}\}$ where (h, r_u, t) indicates there is a "user-item-attribute-item-user" path between user h and user t. $\mathcal{G}_{RU}$ is an undirected graph.

Task Description. We now formulate the recommendation task to be addressed in this paper: Given the CKG $\mathcal{G}_{CK}$ and Relation-Enhanced User Graph $\mathcal{G}_{RU}$, our task is to generate a ranked list of items that will be of interest to user u.

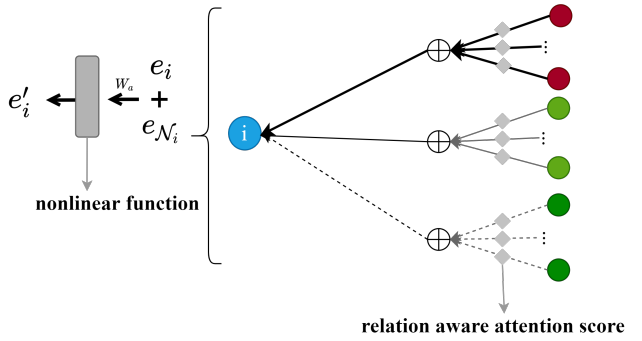

Fig. 3. Relation-specific attention aggregator.We take the item i as an example.

IV. The Proposed Model

To solve the personal recommendation problem, we propose a Relation-Enhanced Multiple Graph Attention Network (REMAN) framework. Figure 2 shows the architecture of the method. Given the graphs $\mathcal{G}_{CK}$ and $\mathcal{G}_{RU}$ shown on the left side of the figure, REMAN makes predictions with three components: Heterogeneous Graph Neural Network layer, Relation-Enhanced User Graph Neural Network layer and Prediction layer.

A. Heterogeneous Graph Neural Network Layer

To capture the heterogeneous and high-order relationships between nodes in CKG $\mathcal{G}_{CK}$, this layer exploits our proposed relation-specific attention aggregator to aggregate information from heterogeneous neighbors, which come from different edges.

1) Relation-specific attention aggregator: We propose the relation-specific attention aggregator to deal with the various types of edges in the heterogeneous graph. In the heterogeneous graph, each node can connect to other nodes with different type of edges.

The key idea of the relation-specific attention aggregator is that: When we are calculate the attention scores between a node and its neighbors, neighbors coming from different types of edges should be treated differently. It is not reasonable to simulatenously apply the softmax function on different relations because they have significantly different characteristics. Taking an item node i in the heterogeneous graph $\mathcal{G}_{CK}$ as an example, Fig.3 illustrates the aggregation process in our method. For a target node h, firstly, we group its neighbors by the edge types. Let $\mathcal{N}_h = \{t|(h, r, t) \in \mathcal{G}_{CK}; h\}$ be the neighbors of h with all types of relations, and $\mathcal{N}_{h,r} = \{t|(h, r, t) \in \mathcal{G}_{CK}; h, r\}$ be the neighbors of h with a given relation type $r \in R$. Secondly, for each relation type r, as shown in Equation 1, we aggregate the information of

neighbors $\mathcal{N}_{h,r}$, which have relation r with the target node, into embedding $\mathbf{e}^r_{\mathcal{N}_h}$:

$$\mathbf{e}^r_{\mathcal{N}_h} = \sum_{t \in \mathcal{N}_{h,r}} \alpha(h,r,t)\mathbf{e_t} \tag{1}$$

where $\alpha(h,r,t)$ is the attention score between h and t in relation r. In this paper, we calculate the attention scores using Equations 2 and 3:

$$e(h,r,t) = \mathbf{e_h}\mathbf{w}_r{\mathbf{e_t}}^T \tag{2}$$

$$\alpha(h,r,t) = \frac{\exp(e(h,r,t))}{\sum_{(h,r,t') \in \mathcal{N}_{h,r}} \exp(e(h,r,t'))} \tag{3}$$

where $e(h,r,t)$ is the attention coefficients between h and t, $\mathbf{w}_r \in \mathbb{R}^{d \times d}$ is the trainable parameters of relation r and the attention scores is normalized across the neighbors in the same relation using the softmax function. Thirdly, we aggregate the neighbor embeddings of all the relation types using Equation 4:

$$\mathbf{e}_{\mathcal{N}_h} = \sum_{r \in R} \mathbf{e}^r_{\mathcal{N}_h} = \sum_{r \in R} \sum_{t \in \mathcal{N}_{h,r}} \alpha(h,r,t)\mathbf{e_t} \tag{4}$$

Finally, we update the embedding of h into $\mathbf{e_h}'$ based on the aggregated information of its neighbors (i.e. $\mathbf{e}_{\mathcal{N}_h}$) and h's current embedding $\mathbf{e_h}$ as follows:

$$\mathbf{e_h}' = agg(\mathbf{e_h}, \mathbf{e}_{\mathcal{N}_h}) = \sigma(\mathbf{W}_a(\mathbf{e}_{\mathcal{N}_h} + \mathbf{e_h})) \tag{5}$$

where $\mathbf{W}_a \in \mathbb{R}^{d \times d}$ are trainable parameters, σ is nonlinear function (e.g., ReLU).

2) Multiple-hops Aggregation: To model high-order relationships between nodes, we perform the relation-specific attention aggregator for n hops. For the node h with an initial embedding $\mathbf{e_h}^{(0)}$, the i-th layer aggregates the embeddings of its i-hop neighborhood nodes in the graph. The new representations of h from different hops are denoted as $\mathbf{e_h}^{(1)}, \mathbf{e_h}^{(2)}, \ldots, \mathbf{e_h}^{(n)}$ respenctively. The final embedding of h is defined as $\widetilde{\mathbf{e}}_\mathbf{h}$:

$$\widetilde{\mathbf{e}}_\mathbf{h} = \mathbf{e_h}^{(0)} || \mathbf{e_h}^{(1)} ... || \mathbf{e_h}^{(n)} \tag{6}$$

The vector $\widetilde{\mathbf{e}}_\mathbf{h} \in \mathbb{R}^{(n+1)d}$ is the concatenation of the embeddings from different aggregation layers and n is the maximum number of hops.

B. Relation-Enhanced User Graph Neural Network Layer

Given the graph $\mathcal{G}_{RU}$, we adopt the same attentive aggregator and propagation rule with heterogeneous graph neural network layer. Notice that $\mathcal{G}_{RU}$ is homogenous graph, the linear combination equation of node h's neighborhood will change from Equation 4 to following equation:

$$\mathbf{e}_{\mathcal{N}_h} = \sum_{t \in \mathcal{N}_h} \alpha(h, r_u, t)\mathbf{e_t} \tag{7}$$

Finally, we get users embedding through layer propagation from $\mathcal{G}_{RU}$. For the user u We represent its output embedding as $\hat{\mathbf{e}}_\mathbf{u} \in \mathbb{R}^d$.

TABLE I
STATISTICS OF DATASETS FOR EXPERIMENTS.

Dataset	#interactions	#users	#items	#entities	#relations
Beauty	198,502	22,363	12,101	42,355	20
Books	1,856,747	52,406	41,264	313,956	49
LastFM	203,975	7,694	30,658	214,524	19

C. Prediction Layer

For the user u and the item i, we can get the representation $\widetilde{\mathbf{e}}_\mathbf{u}$ and $\widetilde{\mathbf{e}}_i$ from heterogeneous graph neural network layer, and the representation $\hat{\mathbf{e}}_\mathbf{u}$ from relation-enhanced user graph layer. We integrate the user u's representations and project u and i representation into a dimensional space:

$$\mathbf{e}^*_\mathbf{u} = \text{MLP}(\widetilde{\mathbf{e}}_\mathbf{u} || \hat{\mathbf{e}}_\mathbf{u}; \Phi^u_{mlp}) \tag{8}$$

$$\mathbf{e}^*_\mathbf{i} = \text{MLP}(\widetilde{\mathbf{e}}_\mathbf{i}; \Phi^i_{mlp}) \tag{9}$$

where we use Φ_{mlp} to represent parameters used to project u and i representation into the same d-dimension space. User representation $\mathbf{e}^*_\mathbf{u}$ and item representation $\mathbf{e}^*_\mathbf{i}$ are fed into a function $f : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^d$ for u's preference score:

$$\hat{\mathbf{y}}_\mathbf{ui} = f(\mathbf{e}^*_\mathbf{u}, \mathbf{e}^*_\mathbf{i}). \tag{10}$$

D. Optimization

To optimize the recommendation model, we opt for the BPR loss. Specifcally, it assumes that the observed interactions, which indicate more user preferences, should be assigned higher prediction values than unobserved ones:

$$\mathcal{L} = \sum_{(u,i,i') \in T} -\ln(\hat{\mathbf{y}}_\mathbf{ui} - \hat{\mathbf{y}}_\mathbf{ui'}) + \lambda ||\Theta||^2_2 \tag{11}$$

where T is the set of the ground-truth triplets, and i' is a random sampled negative item for replacing true item i, Θ is the parameters of our model, and λ is the coefficients of the L_2 regularization.

V. EXPERIMENT

In this section, we evaluate REMAN on the task of personal recommendation. We first introduce the experimental settings, then we compare our model to the baseline methods to demonstrate its effectiveness on recommendation.

A. Experimental Setup

We ran our experiments on three real-world datasets from different domains including two e-commerce recommendation datasets and one music dataset.

- **Amazon**[1] [26] comprises a large corpus of reviews and timestamps on various products. We adopt two categories of diverse size and sparsity, which are Books and Beauty.
- **LastFM**[2] [27] is a music listening dataset released from Last.fm online music system. We take the subset of the

[1]http://jmcauley.ucsd.edu/data/amazon/

[2]http://www.cp.jku.at/datasets/LFM-1b/

TABLE II

PERFORMANCE COMPARISON ON THE THREE DATASETS. THE BEST PERFORMANCE IS BOLD FACED; THE RUNNER UP IS LABELED WITH'$*$'; 'IMPROVE' INDICATES THE IMPROVEMENTS (PAIRED T-TEST WITH PVALUE ¡ 0.01) THAT OUR MODEL ACHIEVES RELATIVE TO THE '$*$' RESULTS.

Dataset	Evaluation metric	CF-based Models	Path-based Models	Embedding-based Models	Hybrid Models				Improve
		BPRMF	PER	CKE	NIPPLE	KGNN-LS	KGAT	REMAN	
Beauty	HR@10	0.321	0.299	0.352	0.392	0.409	0.440*	**0.483**	9.84%
Books		0.502	0.464	0.553	0.601	0.641	0.682*	**0.742**	8.70%
LastFM		0.402	0.363	0.441	0.485	0.508	0.559*	**0.621**	11.10%
Beauty	NDCG@10	0.185	0.146	0.204*	0.235	0.245	0.276*	**0.308**	11.36%
Books		0.302	0.247	0.344	0.394	0.412	0.458*	**0.504**	9.88%
LastFM		0.257	0.213	0.293	0.337	0.353	0.402*	**0.452**	12.51%

dataset where the timestamp is from Jan, 2015 to June, 2015.

For all datasets, we remove users and items with fewer than five interaction records. In our work, we need to obtain the knowledge graph information for items in each dataset. For the LastFM and Amazon datasets of Books and Beauty, similar with [28] we link the items with Freebase entities to enrich the entities and relations. The statistics of two datasets are shown in Table I.

We adopt leave-one-out, which has been widely used in the previous efforts [3], [29]–[31], to evaluate the recommendation performance. For each user, the latest interaction is held out as test set, and the remaining data is utilized as training set, while 5% of data from testing sets are further randomly selected as the validation sets. Aligning with [3], [32], during testing, for each user, we randomly sample 100 items that the user has not interacted with and then rank the test item among the 101 items to avoid heavy computation on all user-item pairs.

B. Evaluation metrics

We provide top-N recommendation list for each item in the testing set, where N=10. Following [3], [32], we adopt Hit-Ratio@N and NDCG@N as the evaluation metrics to compute both metrics for each test user. Here we abbreviate Hit-Ratio@N as HR@10. Generally, higher metric values indicate better ranking accuracy.

C. Parameter Settings

For the baselines, to make a fair comparison, we follow the reported optimal parameter settings and optimize them using the validation set. We implement our method in Tensorflow. We optimize our model using the Adam optimizer. We set the learning rate as 10^{-4}, batch size as 512, embedding size d as 64, dropout ratio as 0.1, coefficient of L2 normalization as 10^{-5} and the number of propagation layer as 3 in CKG $\mathcal{G}_{CK}$ while we apply 2-layer propagation for $\mathcal{G}_{RU}$.

D. Baselines

To evaluate the performance of our methods, we compare our methods with the following methods:

- **BPRMF** [29]: the Bayesian Personalized Ranking based matrix factorization, which is a classic method for learning pairwise personalized rankings from user implicit feedback.
- **PER** [12]: is a representative of path-based methods, which treats the KG as heterogeneous information networks and extracts meta-path based features to represent the connectivity between users and items. Following the previous work [15], [16], We use manually designed "user-item-attribute-item" as meta-paths such as user-book-author-user for Amazon-book and user-musician-country-musician for LastFM.
- **CKE** [8]: is a representative of embedding-based methods, which exploits semantic embeddings derived from TransR [33] to enhance matrix factorization.
- **RippleNet** [13]:is a hybrid method, which is a memory-network-like approach that propagates users' preferences on the KG for recommendation.
- **KGNN-LS** [14]:is a hybrid propagation-based method with KG, which applies GCN on KG to compute the item embedding by propagating and aggregating the neighborhood information on item KG.
- **KGAT** [16]:is a hybrid propagation-based method with KGs, which employs graph attention mechanism on KG to exploit the graph context for recommendation.

E. Comparison against Baselines

We compare REMAN to the state-of-the-art baseline methods for recommendation. From Table II, we can observe that: (1) REMAN outperforms all of the baselines significantly on three datasets. (2) For hybrid models, graph propagation methods KGNN-LS and KGAT perform better than RippleNet. It shows that graph propagation mechanism is useful to consider for recommendation. Specially, KGAT adopts the graph attention mechanism for high-order connectivity, and outperforms KGNN-LS that uses a simple graph convolutional network. (3) The embedding-based model CKE performs better than BPRMF and PER, because it explores knowledge information to a certain extent while path-based model PER heavily relies on the quality of handcrafted meta-paths, and BPRMF lacks

TABLE III

PERFORMANCE COMPARISON OF REMAN$_{w/oKG}$, REMAN$_{w/oRU}$, AND REMAN OVER THREE DATASETS. BEST PERFORMANCE IS WRITTEN IN BOLD.

Dateset	Metric	REMAN$_{w/oKG}$	REMAN$_{w/oRU}$	REMAN
Beauty	HR@10	0.445	0.464	**0.483**
	NDCG@10	0.278	0.291	**0.308**
Books	HR@10	0.681	0.711	**0.742**
	NDCG@10	0.455	0.475	**0.504**
LastFM	HR@10	0.546	0.602	**0.621**
	NDCG@10	0.390	0.426	**0.446**

the KG information. PER performs worst indicating that this method might not make full use of item knowledge. (4) KGAT is the best baseline on the three datasets, as it considers the attributes of the items and high order relationships between items. But it does not really consider aggregation in the context of different relationships and does not adequately leverages information of the heterogeneous graph which is weak in capturing users' personalized preferences. Our work has improved these deficiencies to achieve a better exploitation on KG, REMAN shows its effectiveness for personalized recommendation.

F. Ablation Study

We conduct ablation studies to evaluate the performances of the following our model variants: (1) REMAN$_{w/oKG}$ ignores the knowledge graph of items' meta attributes. (2)REMAN$_{w/oRU}$ removes the relation-enhanced user graph from model, which only considers the node embedding from CKG $\mathcal{G}_{CK}$ for recommendation. The performance of these variants is shown in Table III. From this table, we have following findings: (1) REMAN overperforms all of the variants, indicating that all of our mechanisms are very important for the superior performance of our model. (2) The performance of REMAN$_{w/oKG}$ is the lowest. Such phenomenon demonstrates the significance of items' meta attributes. (3) REMAN achieves better performance than REMAN$_{w/oRU}$. This demonstrates the relation-enhanced user graph is suitable for personalized recommendation to capture users' personalized preferences.

G. Analysis on the relation-specific attention aggregator

In this section, we analyze the impact of the relation-specific attention aggregator. Specifically, we adopt the normal attention aggregator which has been widely used in the previous efforts and denoted it as REMAN$_{nr}$. REMAN$_{nr}$ does not consider the relationship in Equation 4 of neighbor aggregation. Fig.4 presents the performance comparison with and without the pairwise learning strategy. As we can see, REMAN consistently outperforms REMAN$_{nr}$ on three datasets.

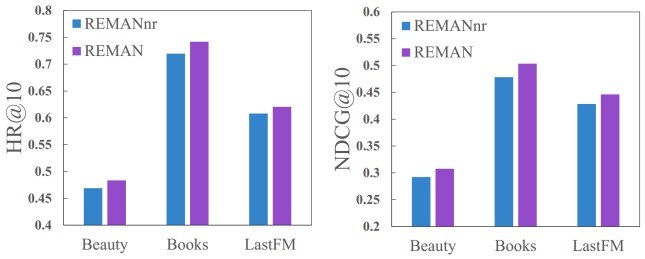

Fig. 4. Performance comparison of the REMAN with REMAN$_{nr}$ on three datasets in terms of HR@10 and NDCG@10.

It indicates the effectiveness of the proposed relation-specific attention aggregator for our approach.

VI. CONCLUSION

In this paper, we propose a novel relation-enhanced multiple graph attention network to models the heterogeneous and high-order relationships among entities in recommendation. We encode user behaviors and item knowledge as a unified relational graph. Then, we innovatively utilize a relation-specific attention aggregator and relation-enhanced user graph to enhance the effectiveness of relation. Relation-specific attention aggregator aims to aggregate the embeddings of the heterogeneous neighbors. Relation-enhanced user graph aims to make up for the limitations of the GNN layer for recommendation and model the user preference. Finally, we make prediction based on the embeddings we learned in graphs. By doing so, our model is able to capture relation-aware high-order connectivity and increase the introduction of user profile information. The empirical results show that our model can significantly outperform strong recommender methods. We also provide detailed analysis on REMAN model. In the future, we aim to explore more effective graph information like relation-enhanced user graph to improve the performance of recommendation.

REFERENCES

[1] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet computing*, no. 1, pp. 76–80, 2003.

[2] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proceedings of the 10th ACM conference on recommender systems*, 2016, pp. 191–198.

[3] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, "Neural collaborative filtering," pp. 173–182, 2017.

[4] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," pp. 285–295, 2001.

[5] S. Rendle, Z. Gantner, C. Freudenthaler, and L. Schmidtthieme, "Fast context-aware recommendations with factorization machines," pp. 635–644, 2011.

[6] X. He and T. Chua, "Neural factorization machines for sparse predictive analytics," pp. 355–364, 2017.

[7] H. Wang, F. Zhang, X. Xie, and M. Guo, "Dkn: Deep knowledge-aware network for news recommendation," pp. 1835–1844, 2018.

[8] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W. Ma, "Collaborative knowledge base embedding for recommender systems," pp. 353–362, 2016.

[9] Q. Ai, V. Azizi, X. Chen, and Y. Zhang, "Learning heterogeneous knowledge base embeddings for explainable recommendation," *Algorithms*, vol. 11, no. 9, p. 137, 2018.

[10] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, "Leveraging meta-path based context for top-n recommendation with a neural co-attention model," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 1531–1540.

[11] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, "Meta-graph based recommendation fusion over heterogeneous information networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 635–644.

[12] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," in *Proceedings of the 7th ACM international conference on Web search and data mining*, 2014, pp. 283–292.

[13] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, "Ripplenet: Propagating user preferences on the knowledge graph for recommender systems," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 417–426.

[14] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, and Z. Wang, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 968–977.

[15] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," pp. 3307–3313, 2019.

[16] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "Kgat: Knowledge graph attention network for recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 950–958.

[17] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidtthieme, "Bpr: Bayesian personalized ranking from implicit feedback," pp. 452–461, 2009.

[18] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Advances in neural information processing systems*, 2013, pp. 2787–2795.

[19] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *International Conference on Learning Representations (ICLR)*, 2016.

[20] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *ICLR*, 2018.

[21] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" *ICLR*, 2019.

[22] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 974–983.

[23] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[24] F. Xu, J. Lian, Z. Han, Y. Li, Y. Xu, and X. Xie, "Relation-aware graph convolutional networks for agent-initiated social e-commerce recommendation," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 529–538.

[25] C. Shi, B. Hu, W. X. Zhao, and P. S. Yu, "Heterogeneous information network embedding for recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 2, pp. 357–370, 2019.

[26] R. He and J. J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada*, 2016, pp. 507–517.

[27] M. Schedl, "The lfm-1b dataset for music retrieval and recommendation," in *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval, New York*, 2016, pp. 103–110.

[28] J. Huang, W. X. Zhao, H. Dou, J. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018*, 2018, pp. 505–514.

[29] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," *arXiv preprint arXiv:1205.2618*, 2012.

[30] I. Bayer, X. He, B. Kanagal, and S. Rendle, "A generic coordinate descent framework for learning from implicit feedback," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 1341–1350.

[31] X. He, H. Zhang, M.-Y. Kan, and T.-S. Chua, "Fast matrix factorization for online recommendation with implicit feedback," in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2016, pp. 549–558.

[32] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable reasoning over knowledge graphs for recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5329–5336.

[33] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Twenty-ninth AAAI conference on artificial intelligence*, 2015.