Trust-Aware Multi-Task Knowledge Graph for Recommendation

Yan Zhou, Jie Guo, *Member, IEEE*, Bin Song, *Senior Member, IEEE*, Chen Chen, Jianglong Chang, and Fei Richard Yu, *Fellow, IEEE*

Abstract—Data sparsity and cold start problems are common in recommender systems. Adding some side information, such as knowledge graph and users' trust relationship, is an effective method to alleviate these problems. However, few work jointly explore the fine-grained implicit relationships between the external heterogeneous graphs to enhance the recommendation accuracy. To address this issue, in this paper, we propose a new method named Trust-aware Multi-task Knowledge Graph (TMKG), which uses multi-task learning to integrate two kinds of side information of trust graph and knowledge graph in an end-to-end manner. First, we mine the intra-graph and inter-graph high-order connections through the node propagation and aggregation, and optimize the embedding of nodes through the implicit relationships obtained. Furthermore, through the shared cross unit, the connection relationships between each layer is mined, and the high-order interaction of nodes of different layers is obtained. We conduct extensive experiments on real-world datasets and prove that our model has the superior performance compared with the state-of-the-art models.

Index Terms—Knowledge graph, multi-task learning, recommender systems, trust graph

1 Introduction

In recent years, with the explosive growth of information, it has become extremely difficult for people to obtain useable information. Recommendation Systems (RS) have been widely used in e-commerce and life service websites, which can effectively solve the information overload problem in the field of movies, music, and food recommendation.

Traditional recommendation algorithms include content-based filtering [1], [2], [3], collaborative filtering [4], [5], [6] and hybrid recommendation approach [7], [8]. Content-based filtering uses the item's explicit attributes defined in advance by human experts to match similar items for recommendation, with high accuracy and strong interpretability. Collaborative Filtering (CF) uses the user and item interactions to find similar users or items with common preferences. Hybrid recommendation algorithm is a combination of content-based filtering and collaborative filtering methods. However, they have a powerless performance when the data is sparse due to data dependence. With the successful application of machine learning [9] in many fields, researchers are committed to applying machine learning in recommendation systems to address this problem. Matrix Factorization (MF) [10] is a learning-based matrix factorization model, which performs recommendation by predicting missing values in the rating matrix. Then, Rendle et al. [11] propose the Factorization Machine (FM), which adds auxiliary vectors on the linear regression model to obtain combined feature vectors. In order to further enhance the representation ability of the model, researchers use Deep Neural Network (DNN) [12] to extract the features of users and items, such as Wide & Deep [13], DeepCrossing [14] and NFM [15]. The recommendation models based on deep neural network have good performance, but they treat the user and item interactions as independent data, without considering their complex connection relationships. The cold start problem still exists. Adding some side information to assist recommendation is an effective way to solve this problem.

Knowledge Graph (KG) [16] and social network [17] can both be regarded as a kind of side information to enhance recommendation. For the knowledge graph, Zhang et al. [18] combine CF and KG for joint training, which apply the item embedding of KG to assist the item information in CF to obtain more accurate recommendation results. Ai et al. [19] construct a user-item knowledge graph, and use the item side information to learn the entities and relationships embedding effectively. For the social network, Wu et al. [20] use the hierarchical attention network to study the effect of different aspects of user preference, and to mine the complex relationships between users and items. Fu et al. [21] use the social relationship as side information, and use the

- *Yan Zhou, Jie Guo, Bin Song, and Chen Chen are with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'An, Shaanxi 710071, China. E-mail: {zhouyan123y, chenchen_123}@stu.xidian.edu.cn, jguo@xidian.edu.cn, bsong@mail.xidian.edu.cn.*
- *Jianglong Chang is with the Guangdong OPPO Mobile Telecommunications Corp., Ltd, Dongguan, Guangdong 523860, China. E-mail: changjianglong@oppo.com.*
- *Fei Richard Yu is with the Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ), Shenzhen University, Shenzhen, Guangdong 518060, China. E-mail: Richard.Yu@Carleton.ca.*

Manuscript received 5 September 2021; revised 23 September 2022; accepted 4 November 2022. Date of publication 10 November 2022; date of current version 21 June 2023.
This work was supported by the National Natural Science Foundation of China under Grants 62071354, 62201424, and 62271324, in part by the China Postdoctoral Science Foundation under Grants 2017M620438, in part by the Natural Science Foundation of Shaanxi Province under Grant 2019ZDLGY03-03, and also supported by the ISN State Key Laboratory and High-Performance Computing Platform of Xidian University.
(Corresponding authors: Jie Guo and Bin Song.)
Recommended for acceptance by E. Chen.
Digital Object Identifier no. 10.1109/TKDE.2022.3221160

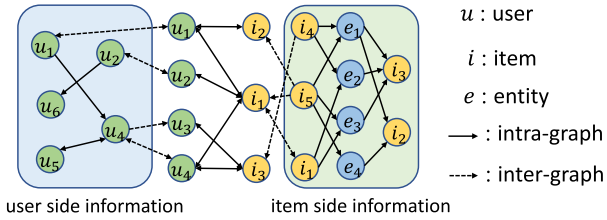


Fig. 1. An illustration of user-item interaction graph and the side information of user and item. Trust relationship is the user side information, and the knowledge graph is the item side information.

graph neural network to model the collaborative relationship to capture user's friend relationship and item information. In addition, Multi-Task Learning (MTL) is also a way to make full use of the side information. Wang et al. [22] use the Knowledge Graph Embedding (KGE) task to assist the recommendation task, and learn the two tasks alternately through the correlation between the items in the interaction graph and the entities in the knowledge graph to optimize the recommendation task.

A single type of side information cannot jointly consider multi-type information from multiple aspects. For example, KG-based recommendation strengthens the item attributes, but does not consider the relationships between users, which will also have a great impact on the recommendation. Recent work attempt to combine the social relationship with the knowledge graph to deeply explore the user and item relationships. Guo et al. [23] fuse trust graph, interaction graph and knowledge graph, and refine user and item feature vectors through the fused heterogeneous multi-relation graph to obtain better recommendation results. However, this method only explore the connection of the user relationship, and does not fully consider the relationships between the user-user graph and the user-item graph. In short, there are few researches jointly considering the inter-graph relationship among user-user trust graph, user-item interaction graph, and knowledge graph.

In general, the KG-based recommendation system uses historical interactions and the knowledge graph to mine the user-item relationships. It does not consider the impact of user attributes and does not utilize the relationships between users. Social network recommendation mainly uses the social relationships and historical interactions to predict, regardless of the influence of item attributes and the item relationships. The user and item representations, which are obtained from the social-network-based and KG-based recommendation, are incomplete. In this paper, we aim to utilize multiple pieces of side information simultaneously in a multi-task manner. Specifically, trust-aware recommendation subtask and knowledge graph recommendation subtask, are designed. We use multi-task learning to simultaneously make user-user relationship and item-item knowledge graph act on recommendation, thus perform feature augmentation on users and items. In addition, each task can implement recommendation independently, so when one of the subtasks lacks data, the favorable recommendation performance can be achieved through the other one. As shown in Fig. 1, the social relationship and the knowledge graph can be used as side information of users and items respectively in user-item interaction graph to assist in modeling. Traditional models aggregate local neighbors for node representation learning, while ignoring fine-grained inter-graph relationships and making knowledge extraction inadequate. Here, inter-graph relations refer to the implicit associations that exist between the interaction graph and the user nodes of the trust graph or knowledge graph, and it is challenging to fully mine the relationships between nodes.

Based on the above problems, we propose the *Trust-aware Multi-task Knowledge Graph (TMKG)* to further explore the relationships of different graphs and to overcome the gap between KG-based recommendation and social network recommendation. Specifically, we formulate two tasks, the trust-aware recommendation task and the KG-based recommendation task to jointly implement preference prediction. In the KG-based recommendation task, we design the shared cross unit to allow the knowledge graph to assist the items in the interaction graph. It is similar in the trust-aware recommendation task. Finally, the preference prediction is performed by concatenating users and the item embeddings obtained from different tasks. In summary, from the perspective of feature level, take the user as an example, we use the shared cross unit to fuse the user features in the user-item interaction graph and user-user trust graph. Besides preserving the original features, potential associations between users are deeply excavated to obtain more accurate user representations. From the perspective of model level, we design two recommendation subtasks. On the one hand, data augmentation is performed on users and items. On the other hand, each subtask can be done individually even when a subtask does not work due to data limitation or other reasons, improving the recommendation performance and scalability of the model. Our main contributions can be summarized in the following four points:

1) The method we proposed is the first work to train the KG-based recommendation and the trust-aware recommendation at the same time in a multi-task manner, as far as we know.
2) We utilize trust graphs and knowledge graphs as external auxiliary information to capture implicit relationships in a finer-grained manner.
3) We use a multi-layer graph network for feature propagation and design a shared cross-unit at each layer for feature aggregation to better model intra-graph and inter-graph relationships.
4) We conduct extensive experiments on real-world datasets and prove that the performance of our method is better than the existing baseline models.

The rest of this paper is organized as follows. Section 2 introduces the related work, including multi-task learning, KG-based recommendation and trust-aware recommendation. In Section 3, we introduce the preliminaries of our model, including the problem statement and notations, graph construction and multi-task recommendation. The proposed method is described in detail in Section 4. Then we show all the details of the experiments in Section 5. Finally, the conclusion is shown in Section 6.

2 Related Work

In this part, we will further introduce the work related to multi-task learning, KG-based recommendation, and trust-aware recommendation.

2.1 Multi-Task Learning

The multi-task model optimizes several different target tasks at the same time, or uses similar tasks to assist in optimizing the same target task. In the recommendation systems, researchers usually use similar tasks to assist each other to improve the performance. Misra et al. [24] propose a cross-stitch network to learn shared representations from multiple tasks. It uses the cross-stitch unit to fuse the parameters learned by multiple networks to obtain the best combination of task-specific. In the recommender systems, Ma et al. [25] draw on the idea of multi-task learning, introduce two similar auxiliary learning tasks, and predict Click Value Rate (CVR) through multi-task learning and shared embedding. This method solved the sample bias and data sparsity problem through the correlation between multiple tasks. Cao et al. [26] utilize KG to enrich the item attribute information in the user-item interaction graph, and utilize the user-item interaction graph to complement the relationships that may be missing in the knowledge graph at the same time. It realizes the mutual enhancement of the two tasks through joint learning. Xin et al. [27] treat the multi-level relationships of multiple items as different tasks and use the attention mechanism to integrate them to obtain user preferences and to improve item feature representations.

TABLE 1
Definitions and Description of Notation

Notation	Definitions and Description
$\mathcal{U}$	the set of users
$\mathcal{I}$	the set of items
$\mathcal{R}$	the set of relationships
G_h	user-item historical interaction graph
G_u	user-user trust graph
G_i	item-item knowledge graph
M_r	projection matrix in transR
E_{uu}	the embedding look-up table
e_u	the embedding of node u
M_{uu}	message embedding of node itself
M_{ui}	propagation embedding of two nodes
N_e	one-hop neighbors of nodes
S	shared cross function
S_w	weighted shared cross function
$h(\cdot)$	information propagate function
W_l, b_l	weight and bias of network
$\odot$	element-wise product
$[\cdot,\cdot]$	concatenation operation
$\mathcal{L}$	loss function
Θ	model parameters
${G_h}^u$	user aggregation interaction graph
${G_h}^i$	item aggregation interaction graph
σ, $LeakyReLU$	activate function

2.2 KG-Based Recommendation

The basic unit of the knowledge graph is the triple, such as *head-relation-tail*. The knowledge graph is a directed heterogeneous graph containing multiple relationships, which can reflect the complex connection information between items. Recommendations based on the knowledge graph can make full use of the items relationships and get better item representation. Some recent work have conducted sufficient research on KG-based recommendation. Wang et al. [28] perform preference propagation to users on the knowledge graph, and automatically discover all possible paths from users to candidate items. Wang et al. [29] sample the entities in the knowledge graph, propagate and aggregate neighbor information of nodes through the graph convolution network (GCN). They exploit the rich semantic information of entities to mine the relationships between entities to improve the prediction accuracy. He et al. [30] introduce the attention mechanism into the graph convolution network, which treats the knowledge graph as a directed graph to capture high-order features between nodes, and improve the performance of the recommendation system.

2.3 Trust-Aware Recommendation

A person's preferences are largely similar to or influenced by those whom he has a social relationship with. Social recommendation makes full use of the user's attributes, interaction and other information to enhance the user's representations and improve the performance of the recommendation. Trust-aware recommendation is a type of social recommendation. The trust relationship can mine high-order connection between users, which improves the performance and the trustworthiness of recommendation results. Lin et al. [31] use the social network and trust relationship to evaluate the social trust between any two users, especially users who are not directly connected. Fan et al. [32] propose a social network recommendation architecture based on the graph neural network (GNN). They explore the more complex relationships between users and items by integrating user-user social graph and user-item graph to get better user and item embeddings. Further work [33] try to use the relationships between items to assist item modeling. But the authors explore the relationships of items through collaborative similarity from the interaction records instead of using the item-item knowledge graph.

3 Preliminaries

In this part, we first introduce the problem statement and notations, and then introduce the construction of graphs, including user-item interaction graph, user-user trust graph and knowledge graph, and finally briefly state the two tasks and prediction method we formulated of our multi-task recommendation.

3.1 Problem Statement and Notations

The purpose of Top-N recommendation is to predict the items that users may be interested in. The items are sorted according to the user's preference scores for the items, and the top N items are shown to the user. Specifically, given the user-item interaction $y_{ui} = (u, i)$, the implicit feedback $y_{ui} = (u, i) = 1$ is used to train the model to obtain the user and item representations, and top-N items are recommended for the target user according to the collaborative similarity between users and items. Besides, we use the additional user and item information to assist in obtaining the user and item feature representations, such as the knowledge graph and social network. As shown in Table 1, we introduce the notations used.

3.2 Graph Construction

In the recommendation system, the intra-relationship of users and items, as well as the inter-relationship of users and items are complicated. In order to extract complex related information, it is a recent trend to describe the data in the form of graphs. The relationships between users and items, users and users, and items and items are respectively constructed into graphs, and the high-order connections of nodes (including users and items) can be obtained and explained in the graphs.

3.2.1 User-Item Interaction Graph

The user-item historical interaction is important information for recommendation. Through the interaction information, we can mine the high-order relationships of users and items to get the users' and items' representations respectively. The interaction between user and item with preference score is called explicit interaction. We use implicit interaction to construct interaction graph where the scores of users to items are represented by 0 and 1, and 1 indicates that the user and the item have an interactive relationship, otherwise it is 0. We define the user-item interaction graph as $G_h = \{(u, y_{ui}, i)|(u \in \mathcal{U}, i \in \mathcal{I})\}$, where $\mathcal{U}$ represents user set and $\mathcal{I}$ is the item set, and y_{ui} represents the connection relationships between nodes of u and i. It is easy to get $y_{ui} = 0$ or $y_{ui} = 1$.

3.2.2 User-User Trust Graph

In addition to the user-item interaction information, the social network can be treated as the side information to assist users, including user attributes and the social relationships. Obviously, the influence of the trust relationship on the user cannot be underestimated, because the user's preferences are often similar to or easily affected by the person he trusts. Similarly, we use implicit interaction of users to construct the user-user trust graph. We define the user-user trust graph as $G_u = \{(o, y_{uu}, e)|(o, e \in U)\}$, where o is trustor user and e is trustee user, and $\mathcal{U}$ represents user set, and y_{uu} represents the trust between o and e. When o and e have a trust relationship, $y_{uu} = 1$, otherwise $y_{uu} = 0$.

3.2.3 Knowledge Graph

The knowledge graph is a heterogeneous graph containing multiple relationships. The relationships connect different entities and provide a comprehensive description of the attributes and relationships of items. For example, in movie recommendation, "Interstellar is directed by Christopher Edward Nolan" is a factual and semantic piece of information and also a triplet *(Interstellar, directed, Christopher Edward Nolan)*. "Directed" is a relationship from head entity "Interstellar" to tail entity "Christopher Edward Nolan". The semantic information is usually stored in the graph structure in the form of triples, and multiple triples containing fact relationships can form a knowledge graph. Specially, in the Last.FM dataset, item refers to the music artist, such as Taylor Swift, Halsey, etc. The genre of Taylor Swift is the pop music, and Halsey's music genre is also the pop music. So we can obtain the relationship between Taylor Swift and Halsey, "$Taylor\ Swift \xrightarrow{genre} pop \xleftarrow{genre} Halsey$," from the triples *(Taylor Swift, genre, pop)* and *(Halsey, genre, pop)* to form the fact that Taylor Swift and Halsey are both pop musicians. Besides the above example, other nodes and relations in the dataset jointly construct the knowledge graph. As in [34], [35], we denote the item-item knowledge graph as $G_i = \{(h, r, t)|(h, t \in \mathcal{I}, r \in \mathcal{R})\}$, where h and t represent the head and tail entity respectively, and r represents the relationship from h to t. Similarly, the $\mathcal{I}$ represents the user set and the $\mathcal{R}$ represents relationship set.

3.3 Multi-Task Recommendation

Traditional recommender systems use the social relationship or the knowledge graph alone. However, we aim to use the social relationship and the knowledge graph to assist users and items at the same time.

The user-item interaction graph is bidirectional and can be seen as two graphs: $u - i$ interaction graph ${G_h}^u$ and $i - u$ interaction graph ${G_h}^i$. Relying on the relationships between the $u - i$ interaction graph and the $i - u$ interaction graph, we formulate two tasks, the trust-aware recommendation task **Task1** and the knowledge graph recommendation task **Task2**.

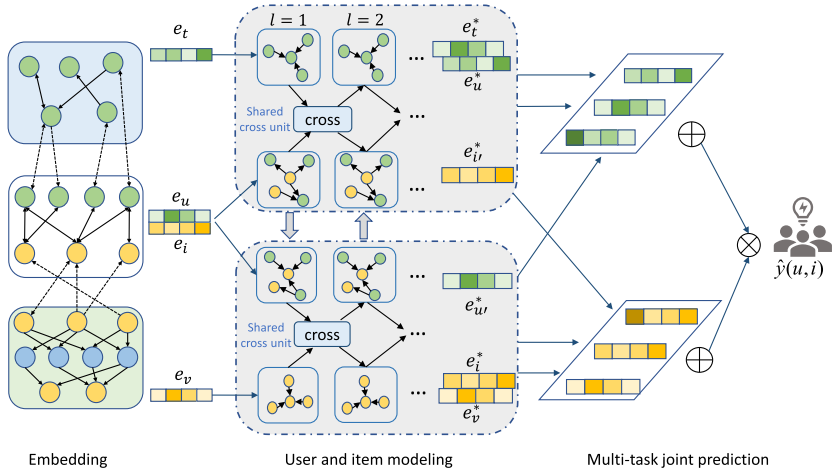
- **Task1**: Given the $u - i$ interaction graph ${G_h}^u$ and trust graph G_u, implement the assistance of the G_u to the ${G_h}^u$ on optimizing the user embedding, and the item embedding is propagated in the $u - i$ interaction graph.
- **Task2**: Given the $i - u$ interaction graph ${G_h}^i$ and knowledge graph G_i, implement the assistance of the G_i to the ${G_h}^i$ on optimizing the item embedding, and the user embedding is propagated in the $i - u$ interaction graph.
- **Joint prediction**: Obtained e_t^*, e_u^* and $e_{i'}^*$ from Task1 and e_v^*, e_i^* and $e_{u'}^*$ from Task2, these embedding from two tasks are used for joint training.

4 Proposed Method

In this part, we introduce the proposed TMKG model in detail. The architecture of the TMKG model is illustrated in Fig. 2. Specifically, the proposed model consists of three parts: (1) Embedding: Initializing the trust graph, interaction graph and knowledge graph to get the initial embeddings of users and items. (2) User and item modeling: Through the trust-aware user data modeling and knowledge graph item data modeling to optimize the user and item embeddings. The shared cross unit captures the high-order connections of nodes between the interaction graph and the trust graph (or the knowledge graph). (3) Multi-task joint prediction: Concatenating user and item feature vectors obtained from two tasks to do prediction, and optimizing the two tasks at the same time.

4.1 Embedding

Following the mainstream methods of recommendation [26], [36], [37], we initialize users, items and entities to obtain the embeddings. Specifically, according to users, items, trust graph, interaction graph, knowledge graph, the embedding look-up tables can be built:



(a) Architecture of TMKG. (b) Shared cross unit.

Fig. 2. (a) The overall architecture of the proposed TMKG model. It contains three major components: Embedded initialization, User and item modeling, and Multi-task joint prediction. (b) The illustration of the share cross unit of our model.

$$E_{uu} = \left[e_{o_1}, \ldots, e_{o_N}, e_{e_1}, \ldots, e_{e_N}\right]$$
$$E_{ui} = \left[e_{u_1}, \ldots, e_{u_N}, e_{i_1}, \ldots, e_{i_M}\right]$$
$$E_{ii} = \left[e_{i_1}, \ldots, e_{i_M}, e_{v_1}, \ldots, e_{v_p}\right], \quad (1)$$

where $e_u = [e_o, e_e], e_u \in R^d$, e_o, e_e represent trustor and trustee of user embedding, $[\cdot, \cdot]$ represents concatenation operation, d represents the embedding size. And $e_i \in R^d$, $e_v \in R^d$ represents item embedding and entity embedding respectively.

Different from the trust graph and the interaction graph, the knowledge graph is a heterogeneous graph that contains multiple types of entities and relationships. The knowledge graph embedding (KGE) [38], [39], [40] method is used to preprocess the knowledge graph. Specifically, the entities and relationships are embedded into the low-dimensional vector space while retaining its inherent structure. TransR [40] is an effective method for the knowledge graph representation learning, which models entities and relationships in different spaces. For a triple (h, r, t), learn the embedding of entities and relationships by optimizing the translation principle $e_h + e_r \approx e_t$, where e_h, $e_t \in R^d$ and $e_r \in R^r$ represent the embedding of h, t and r respectively. The score function is defined as:

$$f(h, r, t) = \left\|e_h^r + e_r - e_t^r\right\|_2^2, \quad (2)$$

where $e_h^r = M_r e_h$, $e_h^t = M_r e_t$, and $M_r \in R^{r \times d}$ is projection matrix to project the entity into the relational space. In our TMKG method, we optimize user and item embeddings by propagating the corresponding embeddings in the trust graph, the interaction graph and the knowledge graph.

4.2 User and Item Modeling

In this part, we focus on the shared cross unit, the trust-aware user data modeling and the knowledge graph item data modeling, which can also be called the multi-task embedding propagation layer.

4.2.1 Shared Cross Unit

In order to mine the feature interaction between users, items, and entities, inspired by DCN [41], we designed the shared cross unit in the proposed TMKG model. Specifically, in the trust-aware recommendation task, we optimize the feature vectors of users and items by trust graphs and $u - i$ interaction graphs. Through high-order feature extraction, we can easily obtain the intra-graph high-order relationships of the trust graph and the interaction graph. Actually, the user information in the trust graph or the interaction graph are closely related. In order to capture the intra-graph and the inter-graph high-order connection, we construct the shared cross unit to align the user o in the trust graph with the user u in the interaction graph. In the real world datasets, the connection relationships between the trustor and the trustee is complicated. For example, each user has someone he trusts and is trusted by others. First, we define a shared cross function S:

$$(e_{u_{l+1}}, e_{o_{l+1}}) = S(e_{u_l}, e_{o_l}) = e_{u_l} \otimes e_{o_l}^\top, \quad (3)$$

where $e_{u_l} \in R^d$ and $e_{o_l} \in R^d$ represent the user embedding of the l layer in the interaction graph and trust graph. To obtain the cross feature matrix, the outer product operator $\otimes$ is employed, which favors the feature interaction of users. Through the S function, the feature vectors obtained in the l layer is crossed and shared, and then the weights and biases are added in vector form. The two vectors can be merged in a more fine-grained form through the element product, and we can get the feature vector representation of the $l+1$ layer:

$$e_{u_{l+1}} = S_l W_l^{uu} + S_l^\top W_l^{ou} + b_l^u$$
$$e_{o_{l+1}} = S_l^\top W_l^{uo} + S_l W_l^{oo} + b_l^o, \quad (4)$$

where $W_l \in R^d$, $b_l \in R^d$ denote the weight and bias. $S_l \in R^{d \times d}$ is the cross feature matrix of layer l. Then we can get the weighted S function S_w:

$$\begin{bmatrix} e_{u_{l+1}} \\ e_{o_{l+1}} \end{bmatrix} = \begin{bmatrix} e_{o_l}^\top W_l^{uu} & e_{u_l}^\top W_l^{ou} \\ e_{o_l}^\top W_l^{oo} & e_{u_l}^\top W_l^{uo} \end{bmatrix} \begin{bmatrix} e_{u_l} \\ e_{o_l} \end{bmatrix} + \begin{bmatrix} b_l^u \\ b_l^o \end{bmatrix}. \quad (5)$$

According to S_w function, we can get the feature vector representation of the $l+1$ layer that has been shared and crossed. The obtained feature vector combines the user

features and relationships of the first l layers, and the user features are fused in each layer. Similarly, we can get the S_w function of the item features in the $i-u$ interaction graph and the knowledge graph:

$$\begin{bmatrix} e_{i_{l+1}} \\ e_{v_{l+1}} \end{bmatrix} = \begin{bmatrix} e_{v_l}^\top W_l^{ii} & e_{i_l}^\top W_l^{vi} \\ e_{v_l}^\top W_l^{vv} & e_{i_l}^\top W_l^{iv} \end{bmatrix} \begin{bmatrix} e_{i_l} \\ e_{v_l} \end{bmatrix} + \begin{bmatrix} b_l^i \\ b_l^v \end{bmatrix}. \tag{6}$$

And through the S_w function, we can fuse the feature representations of the items in each layer to obtain high-order item feature representations in the interaction graph and the knowledge graph.

4.2.2 Trust-Aware Recommendation Data Modeling

User data modeling aims to learn the potential relationships between users and optimize the feature vector representations. In our TMKG model, the nodes' representation is optimized through the propagation and aggregation. Specifically, we optimize user embedding through the trust graph and $u-i$ interaction graph, and at the same time optimize item embedding through $u-i$ interaction graph.

In the trust graph, a user may be the trustor, the trustee person, or both. Through the trust graph, we can get the user feature vector $e_t = [e_o, e_e], e_t \in R^d$. The feature propagation in the trust graph includes two parts, trustor propagation and trustee propagation. Specifically, for a user trust pair $(trustor, trustee)$, we define the information propagation from trustee e to trustor o:

$$M_{oe} = h(e_o, e_e, L_{ap}), \tag{7}$$

where $h(\cdot)$ is the information propagation function, and e_o, e_e is the feature vector of nodes. L_{ap} is the graph laplacian norm, as the delay factor of nodes propagation. M_{oe} is the propagation embedding included in the propagation from trustee e to trustor o. Specifically, we define the first-order trustor propagation:

$$\begin{aligned} M_{oo} &= W_t^1 e_o \\ M_{oe} &= \frac{1}{\sqrt{|N_o||N_e|}} \left(W_t^1 e_e + W_t^2 (e_o \odot e_e)\right), \end{aligned} \tag{8}$$

where W_t is the weight matrix of the trust graph. W_t^1 is shared in the two formulas of Equation (8), and W_t^2 extracts useful information in the process of propagation. M_{oo} represents the message embedding contained in o itself, and M_{oe} represents the propagation embedding obtained from the propagation of e to o. N_o and N_e represent the one-hop neighbors of trustor o and trustee e. And $\odot$ represents the element-wise product between e_o and e_e.

Similarly, we can get the first-order trustee propagation:

$$\begin{aligned} M_{ee} &= W_t^1 e_e \\ M_{eo} &= \frac{1}{\sqrt{|N_o||N_e|}} \left(W_t^1 e_o + W_t^2 (e_e \odot e_o)\right), \end{aligned} \tag{9}$$

where W_t^1 and W_t^2 is shared with Equation (8), M_{ee} represents the message embedding contained in u_e itself, and M_{eo} represents the propagation embedding obtained from the propagation of o to e. Then, the obtained message embedding and propagation embedding are aggregated, and we can get the optimized e_o and e_e:

$$\begin{aligned} e_o^{(1)} &= LeakyReLU(M_{oo} + \sum_{e \in N_o} M_{oe}) \\ e_e^{(1)} &= LeakyReLU(M_{ee} + \sum_{o \in N_e} M_{eo}), \end{aligned} \tag{10}$$

where $LeakyReLU$ represents the activation function. Concatenating e_o and e_e, we can get e_t:

$$e_t^{(1)} = \left[e_o^{(1)}, e_e^{(1)}\right], e_t^{(l)} = \left[e_o^{(l)}, e_e^{(l)}\right], \tag{11}$$

where $e_t^{(l)}$ represents the l-order aggregation user embedding in the trust graph.

In the $u-i$ interaction graph, the propagation from i to u also includes the message embedding and propagation embedding. Through the information propagation function, we can get the M_{uu} and M_{ui}:

$$\begin{aligned} M_{uu} &= W_n^1 e_u \\ M_{ui} &= \frac{1}{\sqrt{|N_u||N_i|}} \left(W_n^1 e_i + W_n^2 (e_u \odot e_i)\right), \end{aligned} \tag{12}$$

where W_n is the weight matrix of the interaction graph. Similarly, in order to obtain the user representation in the $u-i$ interaction graph, we obtain the first-order and high-order user embeddings through the information aggregation:

$$\begin{aligned} e_u^{(1)} &= LeakyReLU(M_{uu} + \sum_{i \in N_u} M_{ui}) \\ e_u^{(l+1)} &= LeakyReLU(M_{uu}^{(l)} + \sum_{i \in N_u} M_{ui}^{(l)}). \end{aligned} \tag{13}$$

At the same time, multiple network layers are connected between different graphs through the S_w in the shared cross unit. And we can obtain the user high-order connection between the trust graph and the $u-i$ interaction graph:

$$\begin{aligned} e_{u_{l+1}} &= e_{u_l} e_{o_l}^\top W_l^{uu} + e_{o_l} e_{u_l}^\top W_l^{ou} + b_l^u \\ e_{o_{l+1}} &= e_{o_l} e_{u_l}^\top W_l^{uo} + e_{u_l} e_{o_l}^\top W_l^{oo} + b_l^o, \end{aligned} \tag{14}$$

where W_l, b_l are the weight and bias we set in the shared cross unit. Simply, through the $u-i$ interaction graph, we can also get the high-order item feature vector:

$$e_{i'}^{(l+1)} = LeakyReLU(M_{uu}^{(l)} + \sum_{i \in N_u} M_{ui}^{(l)}), \tag{15}$$

where M_{ii} represents the message embedding contained in i itself, and M_{iu} represents the propagation embedding obtained from the propagation of u to i. The obtained item feature vector can be used as the auxiliary vector of the knowledge graph item data modeling part to strengthen the item modeling.

4.2.3 Knowledge Graph Recommendation Data Modeling

Unlike the trust graph and the interaction graph, only one kind of relationship is included, and the type of edges are the same. The knowledge graph has more complex structures and relationships. For example, an item may connect multiple items through one relationship, or connect the

same item through multiple relationships. The item connection relationships can be expressed as $i_1 \xrightarrow{r1} e \xrightarrow{r2} i_2$. In the knowledge graph, items are connected through entities and relationships. In order to mine the potential feature of entities and relationships in the knowledge graph, we use graph convolution network to deal with the different relationships between entities. We define a cumulative passing function:

$$e_i^{(l+1)} = \sigma\left(\sum_{m \in \mathcal{M}_i} g_m\left(e_i^{(l)}, e_j^{(l)}\right)\right), \tag{16}$$

where $g_m(e_i, e_j) = We_j$ is the message passing function, W is the weight matrix, and $\mathcal{M}_i$ is the message passing set of node i. $e_i^{(l)}$ represents the embedding vector of node i through l-order propagation, and $e_j^{(l)}$ represents the embedding vector of the l-order neighbor j of node i. And, σ is the activation function.

Different from the traditional GCN, which shares weight W for all edges, we assign different weights to each type of edge. In the knowledge graph, we can get the first-order and high-order item embeddings that have mined the relationships between entities.

$$\begin{aligned} e_i^{(1)} &= \sigma\left(W_0 e_i + \sum_{r \in R} \sum_{j \in N_i} \frac{1}{b_{i,r}} W_r e_j\right) \\ e_i^{(l+1)} &= \sigma\left(W_0^{(l)} e_i^{(l)} + \sum_{r \in R} \sum_{j \in N_i} \frac{1}{b_{i,r}} W_r^{(l)} e_j^{(l)}\right), \end{aligned} \tag{17}$$

where W_0 is the initial weight matrix, W_r is the weight matrix for relation r, and $b_{i,r}$ is the relation-specific regularization constant. e_i represents the embedding vector of node i, and j is the neighbor node of node i.

However, assigning different weights to different relationships may lead to overfitting problems on rare relations, because the knowledge graph contains a large number of different types of entities and relationships. To solve this problem, we decompose $W_r^{(l)}$ to redefine the weight:

$$W_r^{(l)} = \sum_{k=1}^{K} \alpha_{rk}^{(l)} V_k^{(l)}, \tag{18}$$

We representing $W_r^{(l)}$ in the form of linear combination, where $\alpha_{rk}^{(l)}$ is the decomposition factor, $V_k^{(l)} \in R^{d^{(l)}}$ is the basis transformation, $d^{(l)}$ is the dimensionality of this l-layer's representations. In summary, Equation (18) is an effective weight sharing method between edges of different types, which reduces the number of parameters needs to learn for knowledge graph by decomposing the weight matrix $W_r^{(l)}$.

Similar to user data modeling, knowledge graph item data modeling optimizes user and item embeddings through knowledge graph and $i-u$ interaction graph. In the $i-u$ interaction graph, we can get M_{ii} and M_{iu} through the information propagate function $h(\cdot)$:

$$\begin{aligned} M_{ii} &= W_n^1 e_i \\ M_{iu} &= \frac{1}{\sqrt{|N_u||N_i|}}\left(W_n^1 e_u + W_n^2 (e_i \odot e_u)\right), \end{aligned} \tag{19}$$

where W_n^1 and W_n^2 is shared with Equation (12). Through information aggregation, we can get the first-order and high-order item embeddings:

$$\begin{aligned} e_i^{(1)} &= LeakyReLU(M_{ii} + \sum_{u \in N_i} M_{iu}) \\ e_i^{(l+1)} &= LeakyReLU(M_{ii}^{(l)} + \sum_{u \in N_i} M_{iu}^{(l)}). \end{aligned} \tag{20}$$

Finally, multiple network layers are connected between the knowledge graph and the $i-u$ interaction graph through the S_w in the shared cross unit to obtain the item high-order connection:

$$\begin{aligned} e_{i_{l+1}} &= e_{i_l} e_{v_l}^\top W_l^{ii} + e_{v_l} e_{i_l}^\top W_l^{vi} + b_l^i \\ e_{v_{l+1}} &= e_{v_l} e_{i_l}^\top W_l^{iv} + e_{i_l} e_{v_l}^\top W_l^{vv} + b_l^v. \end{aligned} \tag{21}$$

Similarly, we can also get high-order user embedding from the $i-u$ interaction graph as auxiliary information of trust-aware user data modeling,

$$e_{u'}^{(l+1)} = LeakyReLU(M_{ii}^{(l)} + \sum_{u \in N_i} M_{iu}^{(l)}), \tag{22}$$

4.3 Multi-Task Joint Prediction and Loss Function

In the trust-aware user data modeling and knowledge graph item data modeling, we obtain multiple user embeddings and item embeddings in different ways. The embedding obtained at each layer represents different connection relationships. Then, we concatenate these embeddings to get the final user and item embeddings.

$$\begin{aligned} e_t^* &= \left[e_t^{(0)}, e_t^{(1)}, \ldots, e_t^{(l+1)}\right] \\ e_u^* &= \left[e_u^{(0)}, e_u^{(1)}, \ldots, e_u^{(l+1)}\right] \\ e_{u'}^* &= \left[e_{u'}^{(0)}, e_{u'}^{(1)}, \ldots, e_{u'}^{(l+1)}\right], \end{aligned} \tag{23}$$

where $e_t^{(l)} = [e_o^{(l)}, e_e^{(l)}]$, $[\cdot, \cdot]$ represents concatenation operation. e_t^* is user embedding obtained from the trust graph, e_u^* is user embedding obtained from the $u-i$ interaction graph in the trust-aware user data modeling, and $e_{u'}^*$ is the auxiliary user embedding obtained from the $i-u$ interaction graph. We concatenate three user embeddings obtained from different graphs to obtain refined user embedding. Similarly, we can also get item embedding:

$$e'_u = \left[e_t^*, e_u^*, e_{u'}^*\right], e'_i = \left[e_v^*, e_i^*, e_{i'}^*\right], \tag{24}$$

where e'_u is the user embedding obtained from our multi-task embedding propagation layers, and e'_i is the item embedding. e_v^* is the item embedding obtained from the knowledge graph, e_i^* is obtained from the $i-u$ interaction graph, and $e_{i'}^*$ is the auxiliary item embedding obtained from the $u-i$ interaction graph. Then, we can implement joint prediction through auxiliary embeddings. Users' ratings for items can be obtained through the inner product of two vectors:

$$\hat{y}(u,i) = {e'_u}^{\top} e'_i. \tag{25}$$

Then, we use BPR loss [42], which is widely used in recommender systems, to optimize our model. During the training, the observable interactions between the user and the item are taken as positive samples, and the unobserved interactions are taken as negative samples. The observable interaction should theoretically have a higher predictive score:

$$\mathcal{L}_{RS} = \sum_{(u,i,j)\in O} -\ln \sigma(\hat{y}(u,i) - \hat{y}(u,j)), \tag{26}$$

where $O = \{(u,i,j)|(u,i) \in \mathcal{P}, (u,j) \in \mathcal{N}\}$ represents the training set, and $\mathcal{P}$ is the positive sample sets, $\mathcal{N}$ is the negative sample sets. And, σ represents the activate function. For trust-aware user data modeling, we design $\mathcal{L}_{TG}$ to evaluate the similarity of the trustor and trustee feature vectors in the trust graph, and increase the score of trusted user pairs:

$$\mathcal{L}_{TG} = \sum_{\substack{(o,e)\in G_u \\ (o',e') \notin G_u}} -\ln \sigma(\hat{s}(o,e) - \hat{s}(o',e')), \tag{27}$$

where $\hat{s}$ is the score function. For knowledge graph item data modeling, we use $\mathcal{L}_{KG}$ to increase the score of true triples and reduce the score of false triples:

$$\mathcal{L}_{KG} = \sum_{\substack{(h,r,t)\in G_i \\ (h',r,t') \notin G_i}} -\ln \sigma(\hat{s}(h,r,t) - \hat{s}(h',r,t')). \tag{28}$$

Finally, the loss function of our TMKG model can be expressed as:

$$\mathcal{L} = \mathcal{L}_{RS} + \mathcal{L}_{TG} + \mathcal{L}_{KG} + \lambda \|\Theta\|_2^2, \tag{29}$$

where Θ represents the parameters in the model training process, λ is used to control the model parameters.

5 Experiments

In this section, we evaluate the proposed TMKG model. Specifically, we evaluate the performance of the model from the following five aspects.

- **RQ1**: Does our TMKG model perform better than the state-of-the-art KG-based and trust-aware methods?
- **RQ2**: Is the use of trust relationship the key to improving the performance of our model?
- **RQ3**: Is the shared cross unit the key to improving the performance of our model?
- **RQ4**: Do the parameters of the model have a great influence on our model?
- **RQ5**: Is the convergence and effectiveness of our model better than other models?

5.1 Datasets and Data Preprocessing

In order to evaluate the proposed TMKG model, we implement a lot of experiments on three real word benchmark datasets, Last.FM, Amazon-book and Yelp2018. They are different in size, sparsity, and domain. The statistics of the Last.FM dataset is shown in Table 2, and the Amazon-book and Yelp2018 datasets are shown in Table 3.

TABLE 2
The Statistics of the Last.FM Dataset

Last.FM							
Trust graph		Interaction graph			Knowledge graph		
User	Pair	User	Item	Interaction	Entity	Relationship	Triple
1872	25064	1872	3846	42346	9366	60	15518

- Last.FM[1]: This is a music recommendation dataset from Last.fm online music system of real world, which commonly used in recommender systems. This dataset contains the music artist listening information, user trust relationship and artist tagging of two thousand users.
- Amazon-book[2]: Amazon product dataset is a common dataset in product recommendation. We use Amazon-book to train the proposed model. This dataset includes user ratings for items, product matedata and complete review data.
- Yelp2018[3]: This is a dataset from Yelp, the largest review website in the United States, which treats restaurants, bars and other businesses as items, and user comments as interaction data. Yelp2018 is the dataset of the 2018 version of the Yelp Challenge, which is widely used in recommender systems.

For the Last.FM dataset, we follow the method in [23], [34], [35] to construct a trust graph, interaction graph and knowledge graph as the input of our model. For the Amazon-book and Yelp2018 datasets, we follow the methods in [22], [30], [43], [44] to get the interaction graph and knowledge graph. Specifically, the triples of the knowledge graph are obtained by Microsoft Satori[4] which is available to the public. Then, the nodes are aggregated by matching the head and tail entities with their IDs to form a complete knowledge graph. During training, we select 80% of the interaction records as the training set, and the remaining 20% as the test set. For the training set, we take the observable interactions between users and items as positive samples, and negatively sample unobserved user-item interactions randomly.

5.2 Evaluation Metrics

For each user, we evaluate the similarity score for all negative items. Each user has a similarity score for each item. We sort the items according to the score, and achieve top-N recommendations for each user. Following the previous work [30], [45], we select common metrics in recommendation, such as Recall@N, Precision@N, Hit-Ratio@N and NDCG@N to evaluate the performance of recommendation. We set the value of N to 5, 10, 20, 50 respectively.

1. https://grouplens.org/datasets/hetrec-2011
2. http://jmcauley.ucsd.edu/data/amazon
3. https://www.yelp.com/dataset/challenge
4. https://searchengineland.com/library/bing/bing-satori

TABLE 3
The Statistics of the Amazon-Book and Yelp2018 Datasets

		Amazon-book	Yelp2018
User-Item interaction	User	70679	45919
	Item	24915	45538
	Interaction	847733	1185068
Knowledge graph	Entity	88572	90961
	Relationship	39	42
	Triple	2557746	1853704

5.3 Comparable Methods

In order to show the effectiveness of the proposed method, we choose the following models as the baseline models for contrast experiments.

- FM [11]: This is a factorization machine model that introduces the combined characteristics of users and items. Specifically, it is based on the Logistic Regression (LR) model to learn the features' relationships and to solve the feature combination problem of large-scale sparse data.
- NFM [15]: This model introduces DNN into the FM model, which uses multiple hidden layers to capture the nonlinear interactions of features. Using DNN can strengthen the nonlinear ability of the model and help to learn more about the relationships between the combined features.
- CKE [18]: This is a model that combines the knowledge graph and collaborative filtering. This model uses TransR to extract the structured information of the items in the knowledge graph. And then, it's used to take it as the auxiliary information of the item feature vectors in the interaction graph to optimize the recommendation.
- CFKG [19]: This method projects users and items to the same vector space to construct a user-item knowledge graph, which contains users, items, and the item related attributes. Finally, we obtain the recommendation result by calculating the similarity score between users and items.
- KGAT [30]: This is a KG-based recommendation model, which adds users to the entity set and adds interactions to the relationship set to obtain the Collaborative Knowledge Graph (CKG). Modeling graph attention network is based on CKG, the graph convolution is used to mine high-order connections information to optimize user and item embeddings.
- JNSKR [44]: This is a state-of-the-art knowledge graph enhanced recommendation method, which uses a non-sampling strategy instead of negative sampling. In particular, the multi-task learning method is used to jointly optimize the KG embedding task and the recommendation task.
- MGHIF [46]: This is a meta-path-based social recommendation method, which fuses heterogeneous interaction features from user–item bipartite graph and social relation network. In addition, the model fuses multi-hop heterogeneous interaction features via discrete cross-correlations to improve the user and item feature representation.
- T-MRGF [23]: This is a state-of-the-art trust-aware recommendation method, which fuses the trust graph, interaction graph and knowledge graph to explore the user and item relationships, and finally to optimize the user and item feature vectors to get better recommendation results.

5.4 Parameter Settings

Our TMKG model is implemented in tensorflow-GPU-1.14.0 and run on hardware equipment with Inter Core i7-7700 k 4-core 4.20 GHz CPU, NVIDIA GeForce GTX 1080Ti GPU, 32 GB RAM and 1 T SSD.

In order to compare with the baseline models, we set the embedding size of each layer to 64 and the batch size to 1024 in our model. We use Xavier initializer [47] to initialize the model parameters, and use the adaptive moment estimation optimizer (Adam) [48] to optimize these parameters. The learning rate, the parameter of L_2 normalization and the dropout ratio are set to $[0.001, 0.005]$, $[10^{-4}, 10^{-5}]$ and $[0, 0.1, 0.2, \ldots, 0.8]$ respectively. For most baseline models, we follow the settings of the original paper. For example, following [15], we use an MLP network layer with 64 neurons in the NFM model; for KGAT model, we use 3 hidden layers with dimensions of 64, 32, 16 respectively to follow the original paper settings [30]. For our TMKG model, we set the depth of the hidden layer to 2.

5.5 Performance Comparison (RQ1)

In this part, we show the overall performance of our TMKG model when compared with the baseline models.

5.5.1 Overall Comparison

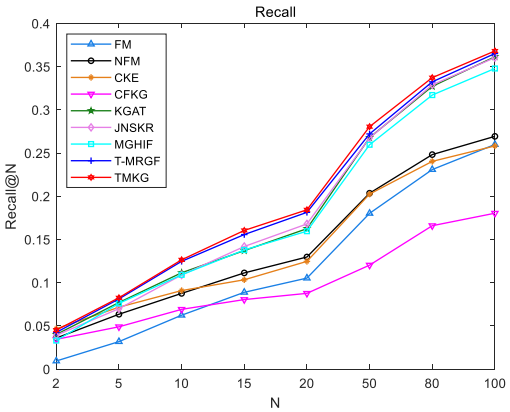
In order to evaluate the performance of the proposed TMKG model, we compare TMKG with all the baseline models. The performance comparison results are shown in Table 4. In general, TMKG performs the best compared with baseline models.

Specifically, FM and NFM are traditional supervised learning models with collaborative filtering. CKE and CFKG are regularization methods that use knowledge graph to assist item modeling. According to the experiment results, we can observe that the NFM model is generally better than the FM, CKE and CFKG models. For FM, it is easy to understand that the using NFM model with DNN is better than operating without one. For the CKE and CFKG models, although using the knowledge graph as side information, they only consider the embedding of aligned entities and ignore the high-order connections. High-order connections may contain rich semantic information, which have a great impact on the recommendation accuracy. At the same time, we can see that compared with the DNN models, the regularization-based models cannot fully learn item representations.

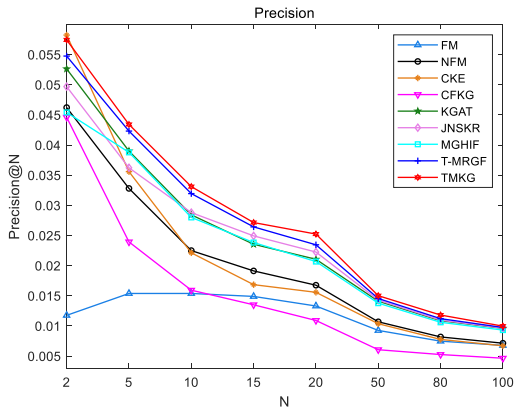
KGAT and JNSKR are relatively advanced knowledge graph enhanced recommendation models, both of which consider high-order connections between nodes. Compared with CKE, the performance of KGAT and JNSKR proves the importance of higher-order connections. T-MRGF is the state-of-the-art recommendation algorithm on the last.FM dataset. It fuses the trust graph, interaction graph and

TABLE 4
Overall Performance Comparison of Different Evaluation Metrics on Last.FM Dataset

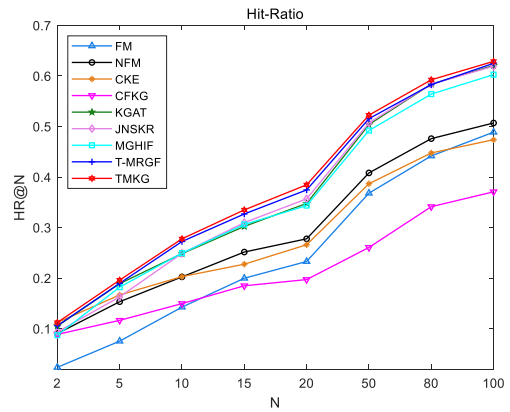
Model	Recall			Precision			Hit-Ratio			NDCG		
	@15	@20	@50	@15	@20	@50	@15	@20	@50	@15	@20	@50
FM	0.0890	0.1054	0.1803	0.0149	0.0133	0.0093	0.1998	0.2329	0.3681	0.0629	0.0707	0.1009
NFM	0.1114	0.1298	0.2036	0.0191	0.0168	0.0107	0.2516	0.2778	0.4081	0.1108	0.1188	0.1496
CKE	0.1035	0.1248	0.2026	0.0168	0.0156	0.0103	0.2276	0.2660	0.3868	0.1169	0.1266	0.1576
CFKG	0.0805	0.0878	0.1204	0.0135	0.0109	0.0061	0.1848	0.1971	0.2607	0.0962	0.0992	0.1124
KGAT	0.1372	0.1626	0.2680	0.0235	0.0210	0.0141	0.3024	0.3478	0.5037	0.1281	0.1393	0.1789
JNSKR	0.1420	0.1681	0.2675	0.0249	0.0223	0.0143	0.3098	0.3568	0.5069	0.1264	0.1385	0.1773
MGHIF	0.1379	0.1596	0.2597	0.0238	0.0207	0.0138	0.3061	0.3435	0.4920	0.1210	0.1302	0.1693
T-MRGF	0.1554	0.1815	0.2722	0.0266	0.0235	0.0145	0.3369	0.3785	0.5125	0.1356	0.1481	0.1842
TMKG	**0.1592**	**0.1844**	**0.2806**	**0.0271**	**0.0252**	**0.0150**	**0.3377**	**0.3846**	**0.5224**	**0.1391**	**0.1521**	**0.1885**
improve%	2.45%	1.60%	3.09%	1.88%	6.81%	3.45%	0.24%	1.61%	1.93%	2.58%	2.70%	2.33%



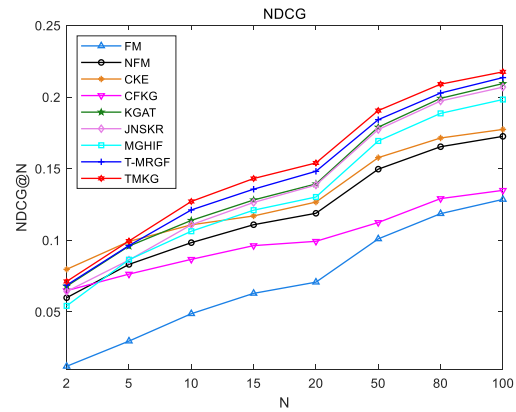
(a) The results of Recall@N in top-N recommendation.



(b) The results of Precision@N in top-N recommendation.



(c) The results of Hit-Ratio@N in top-N recommendation.



(d) The results of NDCG@N in top-N recommendation.

Fig. 3. The performance comparison of different models of different evaluation metrics on Last.FM dataset.

knowledge graph as a heterogeneous graph to refine user and item embeddings. Compared with the strongest baseline T-MRGF, taking the top-20 recommendation as example, the Recall, Precision, Hit-Ratio, and NDCG of TMKG increased by 1.60%, 6.81%, 1.61% and 2.70% respectively. Unlike T-MRGF, which merges three graphs into a whole, TMKG mines the node connections of each graph at each layer, and extracts the high-order relationships between nodes.

In Fig. 3, we can observe that the performance of the knowledge graph methods are better than the traditional methods significantly. In general, the more side information used, the more beneficial it is to the recommendation. In addition, we can see that when N is small, the advantages of JNSKR, T-MRGF, and our TMKG are not obvious, that is, when $N < 10$, the effect of higher-order relationships cannot be fully manifested, and the recommendation results have great uncertainty.

5.5.2 Performance Comparison Without Trust Data

Due to the privacy protection problem of users in the real world, it is difficult to obtain the trust relationship of users. In order to prove the practicability and scalability of our

TABLE 5
The Performance Comparison of top-20 Recommendation on Recall and NDCG on Different Datasets When Trust Relationship is Disabled

Model	Last.FM		Amazon-book		yelp2018	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
FM	0.1054	0.0707	0.1345	0.0886	0.0627	0.0768
NFM	0.1298	0.1159	0.1366	0.0913	0.0660	0.0810
CKE	0.1248	0.1241	0.1343	0.0885	0.0657	0.0805
CFKG	0.0878	0.0992	0.1142	0.0770	0.0522	0.0644
KGAT	0.1626	0.1393	0.1489	0.1006	0.0712	0.0867
JNSKR	0.1681	0.1385	0.1558	0.1068	0.0749	0.0917
MGHIF	0.1596	0.1302	0.1329	0.0907	0.0660	0.0824
N-MRGF	0.1703	0.1408	0.1514	0.1031	0.0734	0.0887
N-TMKG	**0.1802**	**0.1483**	**0.1563**	**0.1071**	**0.0753**	**0.0920**

model, we experiment with its performance in the absence of user trust relationship. N-MRGF means that we have disabled the trust graph in T-MRGF model. N-TMKG means that we have disabled the trust graph in our TMKG model.

It can be seen from Table 5 that in the Last.FM dataset, N-MRGF and N-TMKG are better than other knowledge graph enhancement recommendation models. Because the fusion of the interaction graph and the knowledge graph or the shared cross unit are sufficient to extract the high-order relationships of nodes. Moreover, the non-robustness of negative sampling has little impact on the extraction of relations, and the advantages of non-sampling learning strategy of JNSKR cannot be fully utilized in the last.FM dataset. In the Amazon-book and Yelp2018 datasets, the N-MRGF model is weaker than JNSKR, and our N-TMKG performs better than all baseline models. Unlike Last.FM, Amazon-book and Yelp2018 have more data. And, JNSKR has the best performance in all baseline models because negative sampling may lose a lot of useful information.

Our TMKG model still has the best performance when it does not use the trust graph as side information. This is mainly because we use the shared cross unit to completely mine useful information and optimize the embedding of nodes. In addition, we can see from Table 5 that the advantages of FM and NFM over CKE and CFKG are more obvious in the Amazon-book and Yelp2018 datasets. Generally, on large datasets, the extraction of high-order relationships has a greater impact on the recommendation results.

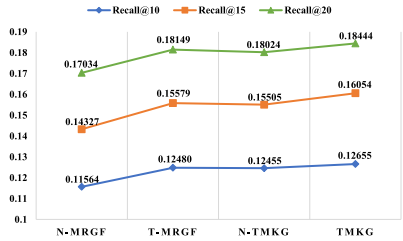
5.6 Model Analysis

In this part, we show the influence of model components and parameters.

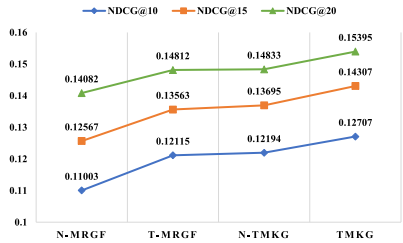
5.6.1 Effect of Trust Graph (RQ2)

To show the influence of the trust relationship on our model, we compare our TMKG model with the T-MRGF model, which also uses the trust relationship as side information on the Last.FM dataset. Fig. 4 shows the T-MRGF and TMKG models that use the trust relationship as side information and the N-MRGF and N-TMKG models that do not use the trust relationship.

In Figs. 4a and 4b reveal the effect of the trust relationship in the T-MRGF and our TMKG models when the test metrics



(a) The test performance of Recall.



(b) The test performance of NDCG.

Fig. 4. The effect of trust relationship on T-MRGF and our TMKG model.

TABLE 6
The Effect of Trust Graph and Shared Cross Unit

Model	Recall			NDCG		
	@15	@20	@50	@15	@20	@50
TMKG	0.1592	0.1844	0.2806	0.1391	0.1521	0.1885
N-TMKG	0.1551	0.1802	0.2708	0.1369	0.1408	0.1833
C-TMKG	0.1558	0.1817	0.2722	0.1356	0.1468	0.1859
NC-TMKG	0.1448	0.1694	0.3670	0.1307	0.1410	0.1792

is Recall and NDCG respectively. From Fig. 4a, we can see that N-TMKG can achieve comparable performance to T-MRGF. When all or none of the trust relationship is used, our model always has the better performance. In Fig. 4b, our TMKG model is always performs better than T-MRGF, regardless of whether the trust relationship is used. At the same time, comparing (a) and (b), we can find that the influence of the trust relationship on T-MRGF is greater than on TMKG. So, our TMKG model is more stable than T-MRGF.

5.6.2 Effect of Shared Cross Unit (RQ3)

In order to show the effect of the shared cross unit in our model, we design C-TMKG, which does not have the shared cross units. N-TMKG represents a no trust relationship, and NC-TMKG represents no trust relationship and shared cross unit. By comparing TMKG, N-TMKG, C-TMKG and NC-TMKG to show the effect of the shared cross unit in Table 6. We experiment with the performance of Recall and NDCG of N=15, 20, 50 respectively. The experiment shows that TMKG performs better than C-TMKG. Therefore, the shared cross unit can improve the recommendation performance. At the same time, when comparing N-TMKG and C-TMKG, we can find the influence of the shared cross unit is weaker than the trust graph. When the trust relationship and the shared cross unit are disabled at the same time, the performance of our model will decrease significantly. All in

TABLE 7
The Effect of Stacked Embedding Propagation Layer Numbers

Model	Recall			NDCG		
	@15	@20	@50	@15	@20	@50
TMKG-2	0.1592	0.1844	0.2806	0.1391	0.1521	0.1885
TMKG-3	0.1567	0.1835	0.2807	0.1382	0.1502	0.1882
TMKG-4	0.1565	0.1832	0.2818	0.1376	0.1498	0.1889

all, in our model, these two components are indispensable and can significantly influence the model performance.

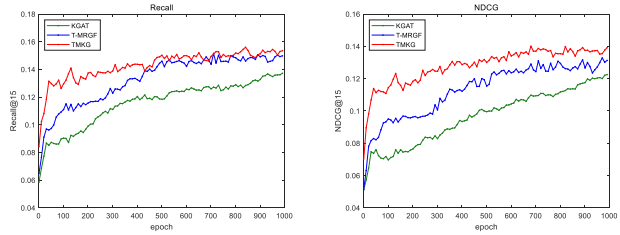
5.6.3 Effect of Embedding Propagation Layer Numbers (RQ4)

In our TMKG model, we use the shared cross unit, so there are at least 2 embedding propagation layers to ensure the transfer of cross data when training. Using different network layers to train the model, we can get different user-item relationships and user-item embeddings. In order to explore the influence of different network layers on the model, we show the performance of TMKG of 2∼4 embedding propagation layers.

TMKG-2, TMKG-3 and TMKG-4 represent using 2, 3 and 4 network layers respectively. From Table 7, we can see that TMKG-2 has the best performance overall. As the number of network layers increases, the model performance gradually decreases. This shows that in our model, the two-layer network is sufficient to mine the relationships between users and items. Increasing the number of network layers will cause an over-smooth of GCN, covering the characteristics of the node itself. Specifically, when stacking more network layers, the node embeddings will become similar, and it becomes difficult to distinguish different nodes. Therefore, excessive propagation and aggregation of nodes is not conducive to the extraction of node features, and it even causes the node features to tend to be consistent, resulting in a decline in model performance.

5.6.4 Training Efficiency Comparison (RQ5)

The effectiveness and convergence of the model are also important aspects of evaluating model performance. To show the effectiveness of our model, we compare the training convergence curves of KGAT, T-MRGF and our TMKG model. For the knowledge graph enhancement recommendation model, we choose KGAT instead of JNSKR. Because JNSKR uses a non-sampling strategy, the training method is different from other baseline models. We observe the changing trend of Recall and NDCG of different epochs. In Figs. 5a and 5b represent the training trend when the test metrics are Recall@15 and NDCG@15 respectively. From Fig. 5, we can see that the convergence speed of our model is faster than KGAT and T-MRGF. Our TMKG model tends to converge at about 300 epochs, while T-MRGF and KGAT need to be trained for at least 500 and 700 epochs respectively.



(a) The trend of Recall@15 of each epoch. (b) The trend of NDCG@15 of each epoch.

Fig. 5. The test performance of each epoch of KGAT, T-MRGF and TMKG.

5.7 Discussion

In most previous works, recommender systems used interaction records to mine the relationships between users and items generally. Nowadays, the knowledge graph is often used as side information to assist the item data modeling in the knowledge graph enhanced recommendation. In social network recommendation, researchers employ the user-user relationships to recommend friends or items for users. However, current work rarely take both social relationships and knowledge graphs as side information to assist recommendation. Our TMKG model treats the knowledge graph enhanced recommendation and the social network recommendation as two sub-tasks, and obtains high-order connections between nodes by extracting the inter-task relationships. Similar to the mutual assistance of two domains in cross-domain recommendation [49], the two tasks in our model assist each other. The difference is that each task can complete the recommended process individually. Essentially, our model belongs to the knowledge graph enhanced recommendation.

The experiment results show that our TMKG model is better than all baseline models. Limited by the dataset, we implement related experiments without a trust relationship on Amazon-book and Yelp2018 datasets. At the same time, the user trust relationship is disabled on the Last.FM dataset for contrast experiments. We use two tasks to assist each other, and use the shared cross unit to mine the high-order intra-task relationships of the nodes. In the absence of the trust relationship, our model still maintains the optimal performance. In addition, based on the knowledge graph enhanced recommendation, we verify the effect of trust relationship and shared cross unit in our model respectively. Through the ablation experiments, we can conclude that both the trust relationship and the shared cross unit significantly improve the performance of the model. Since the trust relationship directly affects the original data, it has a greater impact on the model. Moreover, we find that when the number of embedding propagation layers is two, the model has the best performance. Stacking more embedding propagation layers will cause the node embeddings to become similar, which is harmful to the mining of the node relationships.

Finally, we qualitatively discuss and analyze the effectiveness of addressing data sparsity and cold-start problems. To alleviate the data sparsity problem, we add side information of graph structure on the user and item sides respectively, which is beneficial to mining the potential relationship between users and items, and solves the problem of sparse interaction between users and items. Furthermore, we use trust graph and knowledge graph to mine implicit

interactions between users and items to alleviate the cold start problem.

6 Conclusion

In this paper, we propose a new method named Trust-aware Multi-task Knowledge Graph (TMKG) for recommendation which explores the effect of trust graph and knowledge graph through multi-task learning. We treat the trust graph and the knowledge graph as side information of different sub-tasks to assist recommendation. For the intra-task, we mine the high-order relationships through the propagation and aggregation of nodes, and use shared cross units to mine inter-graph connections. For the inter-task, we also use interaction records and user and item embeddings for mutual assistance. Extensive experiments on real-world datasets show that our TMKG model has the best performance compared with the baseline models.

Our TMKG is the first work that utilizes the trust graph and the knowledge graph in the recommendation system simultaneously in a multi-task manner. The trust graph and knowledge graph are respectively used to enhance the embedding of users and items. More importantly, it integrates social network recommendation and knowledge graph recommendation into a complete network architecture, makes the two recommendation methods assist each other and significantly improves the recommendation accuracy.

In the future, we will consider richer side information, such as user attributes, item contexts and other real-world structured information, to further extend our model to more tasks. In addition, we will consider applying our model architecture to other recommendation scenarios, such as federated learning [50], [51], cross-domain recommendation [52], [53], disentangled representation learning [54], [55], etc.

References

[1] M. Balabanović and Y. Shoham, "Fab: Content-based, collaborative recommendation," *Commun. ACM*, vol. 40, no. 3, pp. 66–72, Mar. 1997.

[2] Y. Zhang, J. Callan, and T. Minka, "Novelty and redundancy detection in adaptive filtering," in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2002, pp. 81–88.

[3] L. Martínez, L. G. Páerez, and M. Barranco, "A multigranular linguistic content-based recommendation model," *Int. J. Intell. Syst.*, vol. 22, no. 5, pp. 419–434, 2007.

[4] R. Van Meteren and M. Van Someren, "Using content-based filtering for recommendation," in *Proc. Mach. Learn. New Inf. Age: MLnet/ECML2000 Workshop*, 2000, pp. 47–56.

[5] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web*, 2001, pp. 285–295.

[6] Z. Zhao and M. Shang, "User-based collaborative-filtering recommendation algorithms on hadoop," in *Proc. IEEE 3rd Int. Conf. Knowl. Discov. Data Mining*, 2010, pp. 478–481.

[7] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock, "Methods and metrics for cold-start recommendations," in *Proc. 25th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2002, pp. 253–260.

[8] G. Lekakos and P. Caravelas, "A hybrid approach for movie recommendation," *Multimedia Tools Appl.*, vol. 36, no. 1, pp. 55–70, 2008.

[9] S. B. Kotsiantis et al., "Supervised machine learning: A review of classification techniques," *Emerg. Artif. Intell. Appl. Comput. Eng.*, vol. 160, no. 1, pp. 3–24, 2007.

[10] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[11] S. Rendle, "Factorization machines," in *Proc. IEEE Int. Conf. Data Mining*, 2010, pp. 995–1000.

[12] P. Covington, J. Adams, and E. Sargin, "Deep neural networks for youtube recommendations," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 191–198.

[13] H.-T. Cheng et al., "Wide & deep learning for recommender systems," in *Proc. 1st Workshop Deep Learn. Recommender Syst.*, 2016, pp. 7–10.

[14] Y. Shan, T. R. Hoens, J. Jiao, H. Wang, D. Yu, and J. Mao, "Deep crossing: Web-scale modeling without manually crafted combinatorial features," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 255–262.

[15] X. He and T.-S. Chua, "Neural factorization machines for sparse predictive analytics," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2017, pp. 355–364.

[16] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 12, pp. 2724–2743, Dec. 2017.

[17] J. Tang, X. Hu, and H. Liu, "Social recommendation: A review," *Social Netw. Anal. Mining*, vol. 3, no. 4, pp. 1113–1133, 2013.

[18] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2016, pp. 353–362.

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

[20] L. Wu, L. Chen, R. Hong, Y. Fu, X. Xie, and M. Wang, "A hierarchical attention model for social contextual image recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 32, no. 10, pp. 1854–1867, Oct. 2020.

[21] B. Fu, W. Zhang, G. Hu, X. Dai, S. Huang, and J. Chen, "Dual side deep context-aware modulation for social recommendation," in *Proc. Web Conf.*, 2021, pp. 2524–2534.

[22] H. Wang, F. Zhang, M. Zhao, W. Li, X. Xie, and M. Guo, "Multi-task feature learning for knowledge graph enhanced recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 2000–2010.

[23] J. Guo, Y. Zhou, P. Zhang, B. Song, and C. Chen, "Trust-aware recommendation based on heterogeneous multi-relational graphs fusion," *Inf. Fusion*, vol. 74, pp. 87–95, 2021.

[24] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3994–4003.

[25] X. Ma et al., "Entire space multi-task model: An effective approach for estimating post-click conversion rate," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 1137–1140.

[26] Y. Cao, X. Wang, X. He, Z. Hu, and T.-S. Chua, "Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences," in *Proc. World Wide Web Conf.*, 2019, pp. 151–161.

[27] X. Xin, X. He, Y. Zhang, Y. Zhang, and J. Jose, "Relational collaborative filtering: Modeling multiple item relations for recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 125–134.

[28] H. Wang et al., "RippleNet: Propagating user preferences on the knowledge graph for recommender systems," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, 2018, pp. 417–426.

[29] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *Proc. World Wide Web Conf.*, 2019, pp. 3307–3313.

[30] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "KGAT: Knowledge graph attention network for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 950–958.

[31] W. Lin, Z. Gao, and B. Li, "Guardian: Evaluating trust in online social networks with graph convolutional networks," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 914–923.

[32] W. Fan et al., "Graph neural networks for social recommendation," in *Proc. World Wide Web Conf.*, 2019, pp. 417–426.

[33] W. Fan et al., "A graph neural network framework for social recommendations," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 5, pp. 2033–2047, May 2022.

[34] H. Wang et al., "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proc. 25th ACM SIGKDD Int. Conf.*, 2019, pp. 968–977.

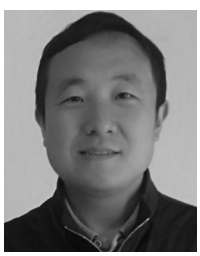
[35] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *Proc. World Wide Web Conf.*, 2019, pp. 3307–3313.
[36] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 173–182.
[37] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 165–174.
[38] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.
[39] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. AAAI Conf. Artif. Intell.*, 2014, pp. 1112–1119.
[40] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, "Learning entity and relation embeddings for knowledge graph completion," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 2181–2187.
[41] R. Wang, B. Fu, G. Fu, and M. Wang, "Deep & cross network for ad click predictions," in *Proc. ADKDD'17*, 2017, pp. 1–7.
[42] R. Steffen, F. Christoph, G. Zeno, and S. Lars, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.
[43] X. Wang, Y. Xu, X. He, Y. Cao, M. Wang, and T.-S. Chua, "Reinforced negative sampling over knowledge graph for recommendation," in *Proc. Web Conf.*, 2020, pp. 99–109.
[44] C. Chen, M. Zhang, W. Ma, Y. Liu, and S. Ma, "Jointly non-sampling learning for knowledge graph enhanced recommendation," in *Proc. Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 189–198.
[45] D. Yang, Z. Song, L. Xue, and Y. Xiao, "A knowledge-enhanced recommendation model with attribute-level co-attention," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 1909–1912.
[46] C. Zhang, Y. Wang, L. Zhu, J. Song, and H. Yin, "Multi-graph heterogeneous interaction fusion for social recommendation," *ACM Trans. Inf. Syst.*, vol. 40, no. 2, pp. 1–26, 2021.
[47] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.
[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[49] P. Li and A. Tuzhilin, "DDTCDR: Deep dual transfer cross domain recommendation," in *Proc. 13th Int. Conf. Web Search Data Mining*, 2020, pp. 331–339.
[50] F. Liang, W. Pan, and Z. Ming, "FedRec++: Lossless federated recommendation with explicit feedback," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 4224–4231.
[51] S. Liu, S. Xu, W. Yu, Z. Fu, Y. Zhang, and A. Marian, "FedCT: Federated collaborative transfer for recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 716–725.
[52] C. Chen, J. Guo, and B. Song, "Dual attention transfer in session-based recommendation with multi-dimensional integration," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 869–878.
[53] Q. Do, W. Liu, J. Fan, and D. Tao, "Unveiling hidden implicit similarities for cross-domain recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 1, pp. 302–315, Jan. 2019.
[54] Y. Wang, S. Tang, Y. Lei, W. Song, S. Wang, and M. Zhang, "DisenHAN: Disentangled heterogeneous graph attention network for recommendation," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 1605–1614.
[55] Y. Zheng, C. Gao, X. Li, X. He, Y. Li, and D. Jin, "Disentangling user interest and conformity for recommendation with causal embedding," in *Proc. Web Conf.*, 2021, pp. 2980–2991.

Yan Zhou received the BE degree in electronic information engineering from the Henan University of Science and Technology, Luoyang, China, in 2020. She is currently working toward the MS degree in information and telecommunication engineering with the School of Telecommunications Engineering, Xidian University, Xi'an, China. Her research interests include deep learning, information retrieval, and recommender systems.

Jie Guo (Member, IEEE) received the BE degree in communication engineering from Zhengzhou University, Zhengzhou, China, in 2011, and the PhD degree from Xidian University, Xian, China, in 2017. She is currently an Associate Professor with Xidian University. From 2015 to 2016, she got the state scholarship fund from China Scholarship Council to be an exchange Ph.D. Student with Carleton University, Canada. Her research interests include information fusion, deep learning, and recommender systems.

Bin Song (Senior Member, IEEE) received the BS, MS, and PhD degrees in communication and information systems from Xidian University, Xi'an, China, in 1996, 1999, and 2002, respectively. He is currently a professor in information and telecommunication engineering with the Xidian University, Xi'an, China. He has authored more than 80 journal papers or conference papers and 40 patents. His research interests include and areas of publication include multimedia communication, multimodal data fusion, content-based image recognition and machine learning, reinforcement learning, Internet of Things, Big Data, and recommender systems.

Chen Chen received the BE degree in communication engineering from the Lanzhou University of Technology, Lanzhou, China, in 2019. He is currently working toward the PhD degree in information and telecommunication engineering with the School of Telecommunications Engineering, Xidian University, Xi'an, China. His current research interests include deep learning, recommender systems, and cross modal retrieval.



Jianglong Chang received the BE and PhD degrees in pattern recognition and intelligent system from the University of Science and Technology of China, Hefei, China, in 2004 and 2009, respectively. He is currently a multi-media expert with Guangdong OPPO Mobile Telecommunications Corp., Ltd. His research interests include media understanding, media retrieval, and image/video editing.

Fei Richard Yu (Fellow, IEEE) received the PhD degree in electrical engineering from the University of British Columbia (UBC), in 2003. His research interests include connected/autonomous vehicles, artificial intelligence, blockchain, and wireless systems. He has been named in the Clarivate Analytics list of "Highly Cited Researchers" since 2019, and received several Best Paper Awards from some first-tier conferences. He is an elected member of the Board of Governors of the IEEE VTS and editor-in-chief for IEEE VTS Mobile World newsletter. He is a fellow of the Canadian Academy of Engineering (CAE), Engineering Institute of Canada (EIC), and IET. He is a distinguished lecturer of IEEE in both VTS and ComSoc.