A Survey on Knowledge Graph-Based Recommender Systems

Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, *Senior Member, IEEE*, Xing Xie, *Senior Member, IEEE*, Hui Xiong, *Fellow, IEEE*, and Qing He

Abstract—To solve the information explosion problem and enhance user experience in various online applications, recommender systems have been developed to model users' preferences. Although numerous efforts have been made toward more personalized recommendations, recommender systems still suffer from several challenges, such as data sparsity and cold-start problems. In recent years, generating recommendations with the knowledge graph as side information has attracted considerable interest. Such an approach can not only alleviate the above mentioned issues for a more accurate recommendation, but also provide explanations for recommended items. In this paper, we conduct a systematical survey of knowledge graph-based recommender systems. We collect recently published papers in this field, and group them into three categories, i.e., embedding-based methods, connection-based methods, and propagation-based methods. Also, we further subdivide each category according to the characteristics of these approaches. Moreover, we investigate the proposed algorithms by focusing on how the papers utilize the knowledge graph for accurate and explainable recommendation. Finally, we propose several potential research directions in this field.

Index Terms—Knowledge graph, recommender system, explainable recommendation

1 Introduction

With the rapid development of the internet, the volume of data has grown exponentially. Because of the overload of information, it is difficult for users to pick out what interests them among a large number of choices. To improve the user experience, recommender systems have been applied in scenarios such as music recommendation [1], movie recommendation [2], [3], and online shopping [4], [5].

The recommendation algorithm is the core element of recommender systems, which can be categorized into collaborative filtering (CF)-based recommender systems, content-based recommender systems, and hybrid recommender systems [6]. CF-based recommendation models user preference based on the similarity of users or items from the interaction data, while content-based recommendation utilizes item's content features. CF-based recommender systems have been widely applied because they are effective to capture the user preference and can be easily implemented in multiple scenarios, without the efforts of extracting features in content-based recommender systems [7], [8]. However, CF-based recommendation suffers from the data sparsity and cold start problems [8]. To address these issues, hybrid recommender systems have been proposed to unify the interaction-level similarity and content-level similarity. In this process, multiple types of side information have been explored, such as item attributes [9], [10], item reviews [5], [11], and users' social networks [12], [13].

In recent years, introducing a knowledge graph (KG) into the recommender system as side information has attracted the attention of researchers. A KG is a heterogeneous graph, where nodes denote entities, and edges represent relations between entities. Items and their attributes can be mapped into the KG to understand the mutual relations between items [14]. Moreover, users and user side information can also be integrated into the KG, which makes relations between users and items, as well as the user preference, can be captured more accurately [15]. Fig. 1 is an example of KG-based recommendation, where the movie "Avatar" and "Blood Diamond" are recommended to Bob. This KG contains users, movies, actors, directors, and genres as entities, while interaction, belonging, acting, directing, and friendship are relations between entities. With the KG, movies and users are connected with different latent relations, which helps to improve the precision of recommendation. Another benefit of the KG-based recommender system is the explainability of recommendation results [16]. In the same example, reasons for recommending these two movies

- *Qingyu Guo is with the Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences (CAS), Beijing 100190, China, and also with the Hong Kong University of Science and Technology, Clearwater Bay, Kowloon, Hong Kong. E-mail: qguoag@connect.ust.hk.*
- *Fuzhen Zhuang and Qing He are with the Key Lab of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences (CAS), Beijing 100190, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: {zhuangfuzhen, heqing}@ict.ac.cn.*
- *Chuan Qin is with the University of Science and Technology of China, Hefei, Anhui 230052, China, and also with the Baidu Talent Intelligence Center, Baidu Inc, Beijing 100085, China. E-mail: chuanqin0426@gmail.com.*
- *Hengshu Zhu is with the Baidu Talent Intelligence Center, Baidu Inc, Beijing 100085, China. E-mail: zhuhengshu@gmail.com.*
- *Xing Xie is with the Microsoft Research Asia, Beijing 100080, China. E-mail: xingx@microsoft.com.*
- *Hui Xiong is with Rutgers, The State University of New Jersey, New Brunswick, NJ 08901-8554 USA. E-mail: hxiong@rutgers.edu.*

Manuscript received 28 Feb. 2020; revised 2 Sept. 2020; accepted 29 Sept. 2020. Date of publication 7 Oct. 2020; date of current version 7 July 2022. (Corresponding author: Fuzhen Zhuang.) Recommended for acceptance by L. Chen. Digital Object Identifier no. 10.1109/TKDE.2020.3028705

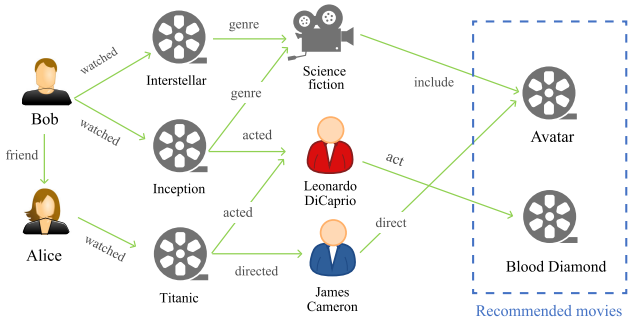

Fig. 1. An illustration of KG-based recommendation.

to Bob can be known by following the relation sequences in the user-item graph. For instance, one reason for recommending "Avatar" is that "Avatar" is the same genre as "Interstellar", which was watched by Bob before. Recently, multiple KGs have been proposed , such as Freebase [17], DBpedia [18], YAGO [19], and Google's Knowledge Graph [20], which makes it convenient to build KGs for recommendation.

Our paper is related to the surveys in the field of graph-based recommendation and applications of KG. On the one hand, Shi *et al.* [21] presented traditional recommendation methods based on the heterogeneous information network, however, latest developed deep learning based models are not covered. Liu *et al.* [22] discussed how to introduce the KG embedding into recommender systems, nevertheless, this is only one implementation of KG-based recommendation. Sun *et al.* [8] reviewed how KGs serve as a type of side information for recommender systems, however, some representative works are missing and the categorization of recommendation approaches are not fine-grained. On the other hand, Zhang and Chen [23] illustrated how KGs bring interpretability to recommender systems, and Wang *et al.* [24] listed some representative methods, however, only a few works are investigated and the inherent relations between algorithms are not mentioned.

Compared with previous works, our survey goes deeper to algorithms and provides a more fine-grained, hierarchical taxonomy. In the first level, we classify these works into three categories, including the embedding-based method, the connection-based method, and the propagation-based method, from the perspective of the leveraged KG-related information. In the second level, we split these three categories into several groups based on their characteristics. Specifically, for embedding-based methods, we group them on the basis of how the KG embedding is learned; while for connection-based methods, we differentiate them according to how to model the connection pattern in the KG; and finally, for propagation-based methods, we distinguish them based on which type of entity is refined in the propagation process. The second contribution is that we elaborate on how different works utilize the KG for explainable recommendation, and summarize common techniques used for explainable recommendation in different methods. In addition, we find that KGs serve as side information in multiple scenarios, including the recommendation for movies, books, news, products, points of interest (POIs), music, and social platform. We gather recent works, categorize them by the application, and collect datasets evaluated in these works.

TABLE 1
Notations Used in This Paper

Notations	Descriptions
u_i	User i
v_j	Item j
e_k	Entity k in the knowledge graph
r_k	Relation between two entities (e_i, e_j) in the knowledge graph
$\hat{y}_{i,j}$	Predicted user u_i's preference for item v_j
$\mathbf{u}_i \in \mathbb{R}^{d\times 1}$	Latent vector of user u_i
$\mathbf{v}_j \in \mathbb{R}^{d\times 1}$	Latent vector of item v_j
$\mathbf{e}_k \in \mathbb{R}^{d\times 1}$	Latent vector of entity e_k in the KG
$\mathbf{r}_k \in \mathbb{R}^{d\times 1}$	Latent vector of relation r_k in the KG
$\mathcal{U} = \{u_1, u_2, \ldots, u_m\}$	User set
$\mathcal{V} = \{v_1, v_2, \ldots, v_n\}$	Item set
$\mathbf{U} \in \mathbb{R}^{d\times m}$	Latent vector of the user set
$\mathbf{V} \in \mathbb{R}^{d\times n}$	Latent vector of the item set
$R \in \mathbb{R}^{m\times n}$	User-Item Interaction matrix
p_k	One path k to connect two entities (e_i, e_j) in the knowledge graph
Φ	Nonlinear Transformation
$\odot$	Element-wise Product
$\oplus$	Vector concatenation

The organization of this survey is as follows: in Section 2, we present notations and concepts used in this paper, as well as foundations of KGs and recommender systems; in Sections 3 and 4, we review KG-based recommender systems from the aspect of approaches and evaluated datasets, respectively; in Section 5, we provide some potential research directions in this field; finally, we conclude this survey in Section 6.

2 Backgrounds

In this section, we introduce the fundamental knowledge and summarize related work in the domain of KG-based recommendation, including KGs and recommender systems. Moreover, before delving into the state-of-the-art approaches exploiting KGs as side information for recommendation, we first present notations and concepts used in the paper to eliminate misunderstanding. For convenience, we list some symbols and their descriptions in Table 1.

• *Recommender Systems.* The recommendation task is to recommend one or a series of unobserved items to a given user, and it can be formulated into the following steps. First, the system learns the representation $\mathbf{u}_i$ and $\mathbf{v}_j$ for the target user u_i and candidate item v_j. Then, it learns a scoring function $f : \mathbf{u}_i \times \mathbf{v}_j \rightarrow \hat{y}_{i,j}$, which models the preference of u_i for v_j. Finally, the recommendation can be generated by sorting the preference scores for items. Recommender systems have been applied in many domains, such as POIs [25], [26], news [16], [27], transportation [28], and education [29], [30]. There have been quite a number of surveys related to recommender systems with different emphases. As recommender systems can be classified into content-based recommender systems, CF-based recommender systems, and hybrid recommender systems [6], Lops *et al.* [31], Su *et al.* [7], and Burke [32] summarized the characteristics of each approach, respectively. Among these three categories, the CF-based recommendation is the most popular strategy,

and Shi *et al.* [33] introduced the latest progress in this direction. Moreover, with the development of deep learning methods, the recommendation architectures have been revolutionized dramatically, therefore, Zhang *et al.* [34] investigated how different deep learning techniques are adopted in recent recommender systems. Readers can check these surveys to learn more fundamental knowledge in this field.

• *Heterogeneous Information Network (HIN).* A HIN is a directed graph $G = (V, E)$ with an entity type mapping function $\phi : V \rightarrow \mathcal{A}$ and a link type mapping function $\psi : E \rightarrow \mathcal{R}$. Each entity $v \in V$ belongs to an entity type $\phi(v) \in \mathcal{A}$, and each link $e \in E$ belongs to a relation type $\psi(e) \in \mathcal{R}$. In addition, the number of entity types $|\mathcal{A}| > 1$ or the number of relation types $|\mathcal{R}| > 1$.

• *Knowledge Graph (KG).* A KG $\mathcal{G} = (V, E)$ is a directed graph whose nodes are entities and edges are subject-property-object triple facts. Each edge of the form (*head entity*, *relation*, *tail entity*) (denoted as $< e_h, r, e_t >$) indicates a relationship of r from entity e_h to entity e_t. A KG can be regarded as an instance of a HIN.

The KG is a practical approach to represent large-scale information from multiple domains [35]. A common way to describe a KG is to follow the Resource Description Framework (RDF) standard [36], in which nodes represent entities, while edges imply the specific relationship between the head entity and tail entity. For example, (Donald Trump, president_of, America) indicates the fact that Donald Trump is the president of America. A KG is a heterogeneous network since it contains multiple types of nodes and relations in the graph. Such a graph has strong representation ability as multiple attributes of an entity can be obtained by following different edges in the graph, and high-level relations of entities can be discovered through these relational links. To date, KGs have been created and applied in multiple scenarios, including search engines, recommender systems, Question Answering system [37], etc. In our collected papers, three open KGs, Freebase [17], DBpedia [18], as well as CN-DBPedia [38], and an enterprise KG, Satori [39], are adopted to provide external knowledge. These KGs contain facts from multiple domains, offering diverse entities and relations for recommender systems. For a more comprehensive review of KGs, we refer readers to [40], [41]. In addition, literature [42] provides practical advice on choosing the KG under different conditions.

In our collected papers, there are two types of KGs, as illustrated below.

- *Item Knowledge Graph.* In the item KG, items and item associated entities, for example, item attributes, serve as nodes. Edges can either stand for item's attribute-level relations, such as brand, category, or user-related relations, such as "co-view", "co-buy".
- *User-Item Knowledge Graph.* In the user-item KG, users, items, and their associated entities serve as nodes. Despite item-related relations in the item KG, relations between the user and the item are also included in the user-item KG, such as "buy", "click", and "mention".
- *Meta-path.* A meta-path $\mathcal{P} = A_0 \xrightarrow{R_1} A_1 \xrightarrow{R_2} \cdots \xrightarrow{R_k} A_k$ is a path defined on the graph of network schema $G_T = (\mathcal{A}, \mathcal{R})$, which defines a new composite relation $R_1 R_2 \ldots R_k$ between type A_0 and A_k, where $A_i \in \mathcal{A}$ and $R_i \in \mathcal{R}$ for $i = 0, \ldots, k$. It is a relation sequence connecting object pairs in a HIN, which can be used to extract connectivity features in the graph.
- *Meta-graph.* Similar to a meta-path, a meta-graph is another meta-structure that connects two entities in a HIN. The difference is that a meta-path only defines one relation sequence, while a meta-graph is a combination of different meta-paths [43]. Compared with a meta-path, a meta-graph can contain more expressive structural information between entities in the graph.
- *Knowledge Graph Embedding (KGE).* KGE is to embed a KG $\mathcal{G} = (V, E)$ into a low dimensional space [44]. After the embedding procedure, each graph component, including the entity and the relation, is represented with a d-dimensional vector. The low dimensional embedding still preserves the inherent property of the graph, which can be quantified by semantic meaning or high-order proximity in the graph. For a more comprehensive understanding of KGE algorithms, we recommend reference [24], [45] to readers.
- *H-hop Neighbor.* Nodes in the graph can be connected with a multi-hop relation path: $e_0 \xrightarrow{r_1} e_1 \xrightarrow{r_2} \cdots \xrightarrow{r_H} e_H$, in this case, e_H is the H-hop neighbor of e_0, which can be represented as $e_H \in \mathcal{N}_{e_0}^H$. Note that $\mathcal{N}_{e_0}^0$ denotes e_0 itself.
- *Entity Triplet Set.* The triplet set of an entity $e \in \mathcal{G}$ is defined as

$$\mathcal{S}_e^k = \left\{ (e_h, r, e_t) | (e_h, r, e_t) \in \mathcal{G} \text{ and } e_h \in \mathcal{N}_e^{k-1} \right\},$$
$$k = 1, 2, \ldots, H.$$

It can be regarded as multiple layers of triplets containing entities from 1-hop neighbors to H-hop neighbors.

3 Methods of Recommender Systems With Knowledge Graphs

In this section, we collect papers related to KG-based recommender systems. Based on how these works utilize the KG information, we group them into three categories, embedding-based methods, connection-based methods, and propagation-based methods. We further subdivide each category according to the characteristics of these approaches.

Generally, the first step in these methods is to build a KG, either the item KG or the user-item KG. Take the construction of an item KG as an example, items are first mapped to the external KG to find their associated entities, then multi-hop neighbors of associated entities are extracted from the KG and form a subgraph for the recommendation system. The graph can also be built from the side information within the provided data, without the assistant of external KGs. The other steps vary in different methods, which will be elaborated in this section.

The explainable recommendation has been another hot research topic in recent years. On the one hand, it is helpful for users to adopt the suggestions generated by the recommender system if appropriate explanations can be provided

TABLE 2
Table of Collected Papers

Category		Model	KG Type	KG Embed.	Issue Solved
Emb.	TSL.	entity2rec [46]	UIKG	node2vec	–
		DKFM [47]	UIKG	TransE	–
		DKN [27]	IKG	TransD	–
		KSR⋆ [48]	IKG	TransE	explainability
		KTGAN [49]	IKG	Metapath2Vec	improper embedding
		BEM [50]	IKG	TransE, GraphSAGE	improper embedding
	JL.	CKE [14]	IKG	TransR	improper embedding
		SHINE [51]	IKG & UIKG	Autoencoder	improper embedding
		CFKG [15]	UIKG	TransE	improper embedding
		ECFKG⋆ [52]	UIKG	TransE	improper embdding, explainability
	MTL.	MKR [53]	IKG	end-to-end	improper embedding
		KTUP⋆ [54]	IKG	TransH	improper embedding, explainability
		RCF⋆ [55]	IKG	DistMult	improper embedding, explainability
Conn.	MSB.	Hete-MF⋆ [56]	IKG	–	–
		Hete-CF⋆ [57]	UIKG	–	–
		HeteRec⋆ [58]	UIKG	–	–
		HeteRec_p⋆ [59]	UIKG	–	personalization
		ProPPR⋆ [60]	UIKG	–	–
		SemRec⋆ [61]	UIKG	–	–
		HeRec⋆ [62]	UIKG	node2vec	–
		FMG⋆ [63]	UIKG	–	limited semantics of the meta-path
	PEB.	MCRec⋆ [1]	UIKG	end-to-end	meta-path modeling
		RKGE⋆ [64]	UIKG	end-to-end	handcrafted meta-paths
		KPRN⋆ [65]	UIKG	end-to-end	handcrafted meta-paths
		EIUM⋆ [66]	UIKG	end-to-end	handcrafted meta-paths
		RuleRec⋆ [67]	IKG	end-to-end	handcrafted meta-paths
		PGPR⋆ [68]	UIKG	end-to-end	handcrafted meta-paths, post-hoc explanation
		Ekar⋆ [69]	UIKG	ConvE	handcrafted meta-paths, post-hoc explanation
Prop.	RU.	RippleNet⋆ [16]	IKG	end-to-end	–
		AKUPM⋆ [70]	IKG	TransR	–
		RCoLM⋆ [71]	IKG	TransR	–
	RI.	KGCN⋆ [72]	IKG	end-to-end	scalability
		KGCN-LS⋆ [73]	IKG	end-to-end	scalability, generalization
	RUI.	KGAT⋆ [74]	UIKG	TransR	scalability
		KNI [75]	UIKG	end-to-end	early summarization
		IntentGC [76]	UIKG	end-to-end	scalability
		AKGE⋆ [77]	UIKG	TransR	noise in the entire KG

In the table, "Emb." stands for embedding-based method, "Conn." stands for connection-based method, "Prop." stands for propagation-based method, "TSL." stands for two-stage learning method, "JL." stands for joint learning method, "MTL." stands for multi-task learning method, "MSB." stands for meta-structure based method, "PEB." stands for path-embedding based method, "RU." stands for refinement of the user, "RI." stands for refinement of the item, "RUI." stands for refinement of the user and item, "⋆" stands for the recommendation model is explainable, "KG Embed." stands for KGE method, "IKG" stands for item KG, "UIKG" stands for user-item KG, "–" stands for no KGE method is adopted in this model, or no issue of this category is solved by this model.

to them. On the other hand, researchers can gain a deeper understanding of the recommendation algorithm [23]. Compared with traditional recommender systems, KG-based recommender systems bring a variety of entities and relations connecting users and items, and are capable of illustrating the reasoning process. In this section, we will also show how different works leverage KGs for explainable recommendation.

To facilitate readers checking the literature, we summarize and organize these papers in Table 2, which lists the approaches to utilize a KG for recommendation, whether the model is explainable, how the model builds the KG, and what issues are solved in each model.

3.1 Embedding-Based Method

Embedding-based methods leverage fruitful facts in the KG to enrich the representation of items or users. There are two basic modules in these works, one is the graph embedding module to learn representations of entities and relations in the KG; and the other one is the recommendation module, which is used to estimate user u_i's preference for item v_j with learned features. Based on how these two modules are associated in the framework, we categorize embedding-based methods into the two-stage learning method, the joint learning method, and the multi-task learning method. The challenges of this method include: 1) how to obtain the entity embedding with the proper KGE method; 2) how to

integrate the learned entity embedding in the recommendation module.

3.1.1 Two-Stage Learning Method

The two-stage learning method stands for training the graph embedding module and the recommendation module one by one. In the first step, representations of entities and relations are learned with KGE algorithms. Then, the pre-trained graph related embeddings are fed into the recommendation module along with other user features and item features to make predictions.

Wang *et al.* [27] proposed DKN for news recommendation. In the first stage, entities in news titles are extracted and mapped into the Satori KG [39] to mine the knowledge-level relations between news. DKN models the news v_j by combining the textual embedding of sentences learned with Kim CNN [78] and the knowledge-level embedding of entities in news content via TransD [79], and the final news representation is $\mathbf{v}_j$. In order to capture the user's dynamic interest in news, the representation of u_i is learned by aggregating the embedding of historical clicked news $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_N\}$ with an attention mechanism

$$\mathbf{u}_i = \sum_{k=1}^{N} s_{v_k, v_j} \mathbf{v}_k, \tag{1}$$

where s_{v_k,v_j} measures the similarity between the candidate news v_j and the clicked news v_k. In the second stage, user's preference for candidate news v_j can be calculated via $\hat{y}_{i,j} = \mathrm{MLP}(\mathbf{u}_i, \mathbf{v}_j)$.

Huang *et al.* [48] proposed the KSR framework for sequential recommendation. KSR utilizes a GRU network to capture the user's sequential preference, and a KV-MV module to model the user's attribute-level preference with knowledge base information. In detail, given the interaction sequence $\{v_1, v_2, \ldots, v_T\}$, the GRU network models the user representation at time t as the hidden state vector

$$\mathbf{h}_i^t = \mathrm{GRU}(\mathbf{h}_i^{t-1}, \mathbf{q}_i^t), \tag{2}$$

where $\mathbf{q}_i^t$ is the item embedding pre-trained with BPR model [80]. For the KV-MN module, it first learns the entity embedding $\mathbf{e}$ and relation embedding $\mathbf{r}$ with the TransE model [81]. The relation embedding are taken as the attribute key, and the user's attribute level preference $\mathbf{m}_i^t$ is modeled with attention mechanism on vectors of item attributes. The final representation of the user and the candidate item consist of embeddings from the above-mentioned KG and interaction related modules, which can be written as $\mathbf{u}_i^t = \mathbf{h}_i^t \oplus \mathbf{m}_i^t$ and $\mathbf{v}_j = \mathbf{q}_j \oplus \mathbf{e}_j \cdot \mathbf{u}_i^t$, respectively. After transforming $\mathbf{u}_i^t$ and $\mathbf{v}_j$ to the same dimension, the user's preference for items can be estimated via inner product

$$\hat{y}_{i,j}^t = \mathrm{MLP}(\mathbf{u}_i^t)^T \cdot \mathrm{MLP}(\mathbf{v}_j). \tag{3}$$

The KSR framework is interpretable by checking user's attention weight over explicit attributes. For example, the "singer" attribute dominating the attention weight for a recommended song indicates the recommendation is generated based on that feature. Therefore, such feature-level attention weight can reflect the user's explicit preference.

Yang *et al.* [49] introduced a GAN-based recommendation model, KTGAN, with external knowledge learned from item KG. In the first phase, KTGAN learns the knowledge embedding $\mathbf{v}_j^k$ for movie v_j by incorporating the Metapath2Vec model [82] on the movie's KG, and the tag embedding $\mathbf{v}_j^t$ with the Word2Vec model [83] on movie's attributes. The initial latent vector of movie v_j is represented as $\mathbf{v}_j^{initial} = \mathbf{v}_j^k \oplus \mathbf{v}_j^t$. Similarly, the initial latent vector of user u_i is represented as $\mathbf{u}_i^{initial} = \mathbf{u}_i^k \oplus \mathbf{u}_i^t$, where $\mathbf{u}_i^k$ is the average of knowledge embeddings of u_i's favored movies, and $\mathbf{u}_i^t$ is u_i's tag embedding. In the second stage, it learns a generator G and a discriminator D to refine initial representations of users and items. The generator G tries to generate relevant movies for user u_i according the score function $p_\theta(v_j|u_i, r)$, where r denotes the relevance between u_i and v_j. During the training process, G aims to let $p_\theta(v_j|u_i, r)$ approximate u_i's true favorite movie distribution $p_{true}(v_j|u_i, r)$, so that G can select relevant user-movie pairs. The discriminator D is a binary classifier to distinguish relevant user-movie pairs and irrelevant pairs according to the learned score function $f_\phi(u_i, v_j)$. The objective function of the GAN module is written as

$$\begin{aligned}\mathcal{L} = \min_\theta \max_\phi \sum_{i=1}^{M} \{ & \mathbb{E}_{v_j \sim p_{\mathrm{true}}(v_j|u_i,r)} \left[\log P(v_j|u_i)\right] \\ & + \mathbb{E}_{v_j \sim p_\theta(v_j|u_i,r)} \left[\log\left(1 - P(v_j|u_i)\right)\right]\}, \\ P(v_j|u_i) = & \frac{1}{1 + \exp\left(-f_\phi(u_i, v_j)\right)},\end{aligned} \tag{4}$$

where $P(v_j|u_i)$ stands for the probability of movie v_j being preferred by user u_i. After the adversarial training, optimal representations of u_i and v_j are learned. Recommendation can be generated for the target user u_i by ranking movies according to the generator's score function $p_\theta(v_j|u_i, r)$. Experiments show that the pre-trained KG embedding can improve the model performance compared with randomly initialized embeddings.

Ye *et al.* [50] proposed BEM, which uses information from two item KGs, one with item-attribute level knowledge, and the other one is named as behavior graph, containing user-related relations, such as "co-view", "co-buy". BEM first learns the initial embeddings from two KGs with the TransE model [81] and the GraphSAGE model [84], respectively. Next, they designed a Bayesian generative model to mutually refine these two representations and preserve item's structural information in each graph. Finally, the recommendation can be generated by finding closest items of the interacted items in the behavior graph under the relation of "co-buy" or "co-click".

The two-stage learning method is easy to implement, where the KG embeddings are generally treated as extra features for the following recommendation module. Another benefit is that KG embeddings can be learned without the interaction data, therefore, large-scale interaction datasets will not increase the computational complexity. Moreover, since the KG is usually stable, it is unnecessary to update embeddings frequently once they are learned. However, the entity embedding optimized by KGE models is more suitable for in-graph applications, like KG completion. Since the KGE module and the recommendation module are loosely coupled, the learned embeddings may not be suitable for recommendation tasks.

3.1.2 Joint Learning Method

Another trend is to jointly learn the graph embedding module and the recommendation module in the end-to-end training fashion. In this way, the recommendation module can guide the feature learning process in the graph embedding module.

Zhang *et al*. [14] proposed CKE, which unifies various types of side information in the CF framework, including item's attribute-level feature, textual feature, and visual feature. The attribute-level feature $\mathbf{x}_j$ is encoded with the TransR [85] to learn structural knowledge from KG, while the textual feature $\mathbf{z}_{t,j}$ and the visual feature $\mathbf{z}_{v,j}$ are extracted with the autoencoder. The objective function of these three feature learning modules are added with the recommendation module to learn parameters jointly

$$\mathcal{L} = \mathcal{L}_{Rec} + \lambda_1 \mathcal{L}_{att} + \lambda_2 \mathcal{L}_{text} + \lambda_3 \mathcal{L}_{vis} + \lambda_4 \mathcal{L}_{reg}, \tag{5}$$

where $\mathcal{L}_{Rec}$, $\mathcal{L}_{att}$, $\mathcal{L}_{text}$, $\mathcal{L}_{vis}$, and $\mathcal{L}_{reg}$ are the objective function of the recommendation module, the attribute-level feature learning module, textual feature learning module, visual feature learning module, and the regularization term, respectively. The final representation of item $\mathbf{v}_j$ is obtained by aggregating the item feature from each part along with the offset vector $\boldsymbol{\eta}_j$ extracted from the user-item interaction matrix through $\mathbf{v}_j = \boldsymbol{\eta}_j + \mathbf{x}_j + \mathbf{z}_{t,j} + \mathbf{z}_{v,j}$. After obtaining the latent vector $\mathbf{u}_i$ of the user u_i, the preference score is estimated via the inner product $\hat{y}_{i,j} = \mathbf{u}_i^T \mathbf{v}_j$. Experiments show that incorporating structural knowledge can boost the performance of recommendation.

Wang *et al*. [51] proposed SHINE, which formulates the user recommendation task as the sentiment link prediction task between entities in the graph. SHINE utilizes information from multiple sources, including the sentiment network G_s which represents attitude among users, the social network G_r which contains user relations, and the profile network G_p with user attribute-level knowledge. User features $\mathbf{u}_i^s, \mathbf{u}_i^r, \mathbf{u}_i^p$ are learned from G_s, G_r, G_p with the autoencoder model, respectively, then are aggregated for the final user representation $\mathbf{u}_i$. The preference between user u_i and u_j can be modeled from their corresponding representations in the prediction model

$$\mathbf{u}_i = \mathrm{AGG}\big(\mathbf{u}_i^s, \mathbf{u}_i^r, \mathbf{u}_i^p\big), \quad \hat{y}_{i,j} = f(\mathbf{u}_i, \mathbf{u}_j), \tag{6}$$

where $\mathrm{AGG}(\cdot)$ is the aggregation operator. This model is also optimized end-to-end by adding loss terms together.

Zhang *et al*. [15] proposed CFKG, which constructs a user-item KG for recommendation. CFKG adopts the TransE model [81] to encode the graph, and learns the embedding of entities and relations in the graph with a hinge loss

$$\mathcal{L} = \sum_{(e_h,r,e_t)\in\mathcal{G}} \Bigg\{ \sum_{(e_h,r,e_t')\in\mathcal{G}^t} \big[\gamma + \|\mathbf{e}_h + \mathbf{r} - \mathbf{e}_t\|_2 - \|\mathbf{e}_h + \mathbf{r} - \mathbf{e}_{t'}\|_2\big]_+ + \sum_{(e_h',r,e_t)\in\mathcal{G}^h} \big[\gamma + \|\mathbf{e}_h + \mathbf{r} - \mathbf{e}_t\|_2 - \|\mathbf{e}_{h'} + \mathbf{r} - \mathbf{e}_t\|_2\big]_+ \Bigg\}, \tag{7}$$

where γ is the margin, $\mathcal{G}^t$ and $\mathcal{G}^h$ are triplets obtained by replacing the tail entity and the head entity in correct triplets $(e_h, r, e_t) \in \mathcal{G}$, respectively. In the recommendation phase, the system ranks candidate items according to the euclidean distance between u_i and v_j measured by the "buy" relation

$$d_{ij} = \big\|\mathbf{u}_i + \mathbf{r}_{buy} - \mathbf{v}_j\big\|_2, \tag{8}$$

where $\mathbf{r}_{buy}$ is the learned embedding for the relation type "buy". A smaller distance between u_i and v_j measured by the "buy" relation refers to a higher preference score $\hat{y}_{i,j}$. In the follow-up work ECFKG proposed by Ai *et al*. [52], they illustrated that explanations for the recommendation can be provided by extracting relation paths between the target user and the candidate item.

The joint learning method can be trained end-to-end, and it can use the KG structure to regularize the recommender system. Nevertheless, the combination of different objective functions needs to be fine-tuned.

3.1.3 Multi-Task Learning Method

A recent research direction is to adopt the strategy of multi-task learning, to train the recommendation task with the guidance of the KG-related task. The motivation is that items in the user-item interaction bipartite graph and their associated entities in the KG are likely to share similar structures. Therefore, the transferring of low-level features between items and entities is helpful for facilitating the improvement of recommender system.

Wang *et al*. [53] proposed MKR, which consists of a recommendation module and a KGE module. Instead of feeding KG embeddings into the recommendation module, these two modules are independent and are connected with a cross & compress unit to share knowledge. The recommendation module is trained to estimate user's preference for candidate items, while the KGE module is trained to estimate the tail entity representation given the head entity and the relation in a triplet (e_h, r, e_t). In detail, the recommendation module feeds initial user representation $\mathbf{u}$ into L-layer MLP to obtain final user representation $\mathbf{u}^L$. The final item representation $\mathbf{v}^L$ is refined by L-layer cross & compress unit with their associated entities $e \in \mathcal{S}_v$ in the KG. The user preference for the candidate item is estimated with a nonlinear function

$$\begin{aligned} &\mathbf{u}^L = f_u^L(\mathbf{u}), \quad \mathbf{v}^L = \mathbb{E}_{e\in\mathcal{S}_v}\big[C^L(\mathbf{v}, \mathbf{e})[\mathbf{v}]\big], \\ &\hat{y}_{u,v} = \Phi\Big(\big(\mathbf{u}^L\big)^T \mathbf{v}^L\Big). \end{aligned} \tag{9}$$

Similarly, the KGE module learns final relation representation $\mathbf{r}^L$ with the L-layer MLP, and the head entity representation is refined through the L-layer cross & compress unit with their associated items $v \in \mathcal{S}_{e_h}$. Then, the model predicts tail entity embedding $\hat{\mathbf{e}}_t$ by concatenating these two embeddings, followed by a K-layer MLP. The similarity between $\hat{\mathbf{e}}_t$ and the real tail entity embedding $\mathbf{e}_t$ is measured by a score function

$$\begin{aligned} &\mathbf{r}^L = f_r^L(\mathbf{r}), \quad \mathbf{e}_h^L = \mathbb{E}_{v\in\mathcal{S}_{e_h}}\big[C^L(\mathbf{v}, \mathbf{e}_h)[\mathbf{e}]\big], \\ &\hat{\mathbf{e}}_t = f_t^K\big(\mathbf{r}^L \oplus \mathbf{e}_h^L\big), \\ &s(e_h, r, e_t) = \Phi\big(\mathbf{e}_t^T \hat{\mathbf{e}}_t\big). \end{aligned} \tag{10}$$

These two modules share low-level features in each part, and are trained alternatively. The final objective function is

$$\begin{aligned}\mathcal{L} =& \mathcal{L}_{Rec} + \lambda_1 \mathcal{L}_{KG} + \lambda_2 \mathcal{L}_{reg} \\ =& \sum_{u \in \mathcal{U}, v \in \mathcal{V}} \mathcal{J}\left(\hat{y}_{u,v}, y_{uv}\right) \\ & - \lambda_1 \left(\sum_{(e_h, r, e_t) \in \mathcal{G}} s(e_h, r, e_t) - \sum_{\left(e'_h, r, e'_t\right) \notin \mathcal{G}} s(e'_h, r, e'_t) \right) \\ & + \lambda_2 \|\mathbf{W}\|_2^2,\end{aligned} \tag{11}$$

where $\mathcal{J}$ is the cross-entropy function, and $\mathbf{W}$ is the trainable parameters, $\mathcal{L}_{Rec}$, $\mathcal{L}_{KG}$, and $\mathcal{L}_{reg}$ are the objective function of the recommendation task, the KGE task, and the regularization term, respectively.

Cao *et al.* [54] proposed KTUP to jointly learn the task of recommendation and knowledge graph completion. They further considered the user's preference can be reflected by relations among items in the KG. Since some facts are missed in the KG, by transferring low-level features of items and relations from the recommendation task, better representation of entities and relations in the KG can be learned in the KG completion task, which mutually improves performance in two tasks. KTUP adopts the TransH [86] to learn the entity and relation embedding via

$$\begin{aligned}&\mathbf{e}_h^{\perp} = \mathbf{e}_h - \mathbf{w}_r^T \mathbf{e}_h \mathbf{w}_r, \quad \mathbf{e}_t^{\perp} = \mathbf{e}_t - \mathbf{w}_r^T \mathbf{e}_t \mathbf{w}_r, \\ &s(e_h, r, e_t) = \left\| \mathbf{e}_h^{\perp} + \mathbf{r} - \mathbf{e}_t^{\perp} \right\|.\end{aligned} \tag{12}$$

In the recommendation module, KTUP first induces user preferences from the interaction history. Then, it projects representations of the user and the item to the preference hyperplane, and adopts a score function similar to the TransH. The representation of the item is enhanced by related entities in the KG, while the preference vector is enriched by predefined relation mapping in the KG

$$\begin{aligned}&\mathbf{u}_i^{\perp} = \mathbf{u}_i - \mathbf{w}_p^T \mathbf{u}_i \mathbf{w}_p, \quad \hat{\mathbf{p}} = \mathbf{p} + \mathbf{r}, \\ &\hat{\mathbf{v}}_j = \mathbf{v}_j + \mathbf{e}, \quad \hat{\mathbf{w}}_p = \mathbf{w}_p + \mathbf{w}_r, \quad \hat{\mathbf{v}}_j^{\perp} = \hat{\mathbf{v}}_j - \hat{\mathbf{w}}_p^T \hat{\mathbf{v}}_j \hat{\mathbf{w}}_p, \\ &\hat{y}_{i,j} = \left\| \mathbf{u}_i^{\perp} + \hat{\mathbf{p}} - \hat{\mathbf{v}}_j^{\perp} \right\|.\end{aligned} \tag{13}$$

These two modules are jointly trained, and the objective function is

$$\begin{aligned}\mathcal{L} =& \mathcal{L}_{Rec} + \lambda \mathcal{L}_{KG} \\ =& \sum_{(i,j) \in R} \sum_{(i,j') \notin R} -\log \Phi\left(\hat{y}_{i,j} - \hat{y}_{i,j'}\right) \\ & + \lambda \sum_{(e_h, r, e_t) \in \mathcal{G}} \sum_{\left(e'_h, r', e'_t\right) \notin \mathcal{G}} \left[s(e_h, r, e_h) + \gamma - s(e'_h, r', e'_h) \right]_+,\end{aligned} \tag{14}$$

where (i, j) and (i, j') are positive samples and negative samples in the user-item interaction matrix, respectively; $[\cdot]_+$ is the hinge loss function, γ is the margin, $\mathcal{L}_{Rec}$ denotes the objective function of the recommendation task, and $\mathcal{L}_{KG}$ represents the objective function of the KG completion task. Representations of items and preferences can be enriched by transferring knowledge of entities, relations and preferences in each module under the framework of KTUP. By extracting the attention weight of the explicitly modeled user preference, relations that the user concern most are available in the system, which forms the preference-level explanation. Moreover, the model can detect items in the interaction history and entities in the KG that satisfy those salient relations to offer more solid explanations.

By applying the multi-task learning strategy, it is helpful to prevent the recommendation system from overfitting, and improve the generalization ability of the model. However, similar to the joint learning method, it requires efforts to integrate different tasks under one framework.

TABLE 3
Comparisons Between Embedding-Based Methods

	Main advantages	Main shortcomings
TSL.	∗ easy to implement ∗ scalability	∗ improper embedding
JL.	∗ end-to-end training	∗ fine-tune weight of different objective functions
MTL.	∗ end-to-end training ∗ generalization	∗ fine-tune weight of different objective functions

In the table, "TSL." stands for the two-stage learning method, "JL." stands for the joint learning method, and "MTL." stands for the multi-task learning method.

3.1.4 Summary for Embedding-Based Method

In this section, we summarize the advantages and shortcomings of each type of the embedding-based method, which are listed in Table 3.

Although the two-stage learning method is easy to implement, the learned entity embedding may not be suitable for the recommendation task. To solve this issue, initial entity embeddings are refined by the GAN model in KTGAN [49], and the Bayesian generative model in BEM [50]. The joint learning method learns optimized entity embedding through end-to-end training, and the multi-task learning method further improves the generalization of the model by transferring knowledge from KG-related tasks. However, it requires plenty of experiments to find the optimal combination of different objective functions.

3.2 Connection-Based Method

Connection-based method utilizes the connection patterns in the graph to guide the recommendation. Most works in this group utilize the user-item KG to mine the relationships among entities in the graph. There are two main approaches in exploring the connective information in the KG. The first direction is to utilize the meta-structure in the graph, including meta-path and the meta-graph, to calculate the similarity between entities. The meta-structure based similarity can serve as the constraint for the representations of users and items, or can be used to predict users' interests from similar users or similar items in the interaction history. The second solution is to encode the connection pattern between user-item pair or item-item pair into vectors, which can be integrated into the recommendation framework. We call such a method as the path-embedding based method. The challenges of this method include: 1) how to design proper meta-paths for different tasks; 2) how to model the connection patterns between entities.

3.2.1 Meta-Structure Based Method

One implementation of the meta-structure based method is to utilize the connective similarities of entities in different

meta-paths as the graph regularization to constrain the representation of users and items. The motivation is that entities with high meta-path based similarity should be close in the latent space. The objective function can be written as

$$\mathcal{L} = \mathcal{L}_{Rec} + \lambda \mathcal{L}_{Sim}, \tag{15}$$

where $\mathcal{L}_{Rec}$ represents the objective function of recommendation, and the common selection is the Matrix Factorization $\sum_{i=0}^{m}\sum_{j=0}^{n}\left(U_i^T V_j - \mathbb{R}_{i,j}\right)^2$. The similarity constraint $\mathcal{L}_{Sim}$ guides the learning of the user embedding and the item embedding. To measure the connectivity similarity between entities in the graph, *PathSim* [87] is commonly used. It is defined as

$$s_{x,y} = \frac{2 \times \left|\left\{p_{x \rightsquigarrow y} : p_{x \rightsquigarrow y} \in \mathcal{P}\right\}\right|}{\left|\left\{p_{x \rightsquigarrow x} : p_{x \rightsquigarrow x} \in \mathcal{P}\right\}\right| + \left|\left\{p_{y \rightsquigarrow y} : p_{y \rightsquigarrow y} \in \mathcal{P}\right\}\right|}, \tag{16}$$

where $p_{m \rightsquigarrow n}$ is a path between the entity m and n. Three types of entity similarities are commonly utilized,

• *User-User Similarity*. The objective function becomes

$$\min_{\mathbf{U},\mathbf{\Theta}} \sum_{l=1}^{L} \theta_l \sum_{i=1}^{m}\sum_{j=1}^{m} s_{i,j}^{l} \left\|\mathbf{u}_i - \mathbf{u}_j\right\|_F^2, \tag{17}$$

where $\|\cdot\|_F$ denotes the matrix Frobenius norm, $\mathbf{\Theta} = [\theta_1, \theta_2, \ldots, \theta_L]$ denotes the weight for each meta-path, $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m]$ denotes latent vectors of all users, and $s_{i,j}^{l}$ denotes the similarity score of user i and j in meta-path l. The user-user similarity forces the embeddings of users to be close in the latent space if users share high meta-path-based similarity.

• *Item-Item Similarity*. The objective function is

$$\min_{\mathbf{V},\mathbf{\Theta}} \sum_{l=1}^{L} \theta_l \sum_{i=1}^{n}\sum_{j=1}^{n} s_{i,j}^{l} \left\|\mathbf{v}_i - \mathbf{v}_j\right\|_F^2, \tag{18}$$

where $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n]$ denotes latent vectors of all items. Similar to the user-user similarity, the low-rank representations of items should be close if their meta-path-based similarity is high.

• *User-Item Similarity*. The objective function of this term can be written as

$$\min_{\mathbf{U},\mathbf{V},\mathbf{\Theta}} \sum_{l=1}^{L} \theta_l \sum_{i=1}^{m}\sum_{j=1}^{n} \left(\mathbf{u}_i^T \mathbf{v}_j - s_{i,j}^{l}\right)^2. \tag{19}$$

The user-item similarity term will force the latent vector of users and items to be close to each other if their meta-path-based similarity is high.

Yu *et al*. [56] proposed the Hete-MF, which extracts L different meta-paths and calculates item-item similarity in each path. The item-item regularization is integrated with the matrix factorization method [88] to refine low-rank representation of users and items for better recommendation. Later, Luo *et al*. [57] proposed Hete-CF to find the user's affinity to unrated items by taking the user-user similarity, item-item similarity, and user-item similarity together as regularization terms. Therefore, the Hete-CF outperforms the Hete-MF model.

Another type of meta-structure based methods utilize the entity similarity to predict users interests for unrated items, which can be regarded as preference fusion in the KG. Yu *et al*. [58] proposed HeteRec, which leverages the meta-path similarities to enrich representations of users and items. The motivation is that users history interaction can reflect user's preference, and recommendations to this user can be made by finding similar items to the ones that the target user has interacted before. HeteRec first defines L different types of meta-paths that connect items in the graph, which further forms L item-item similarity matrices $S^{(l)} \in \mathbb{R}^{n \times n} (l = 1, 2, \ldots, L)$, where $S_{i,j}^{(l)}$ is the similarity between item i and j under the relation defined by the lth meta-path. The target user's preference matrix under lth type of meta-path relation can be obtained via $\tilde{R}^{(l)} = RS^{(l)}$, where R is the user-item interaction matrix. By applying non-negative matrix factorization technique [89] on these L user preference matrices, a series of refined latent vectors of users and items can be obtained

$$\left(\hat{\mathbf{U}}^{(l)}, \hat{\mathbf{V}}^{(l)}\right) = \text{argmin}_{\mathbf{U},\mathbf{V}} \left\|\tilde{R}^{(l)} - \mathbf{U}^T\mathbf{V}\right\|_F^2 \text{ s.t. } \mathbf{U} \geq 0, \quad \mathbf{V} \geq 0. \tag{20}$$

Finally, the recommendation can be generated by linear combination of user's preference on each path, with the scoring function

$$\hat{y}_{i,j} = \sum_{l=1}^{L} \theta_l \cdot \hat{\mathbf{u}}_i^{(l)T} \hat{\mathbf{v}}_j^{(l)}, \tag{21}$$

where θ_l is the weight for the user-item latent vector pair in the lth path. One limitation for HeteRec is that the learned weight θ_l for each path is the same for all users, which impedes the degree of personalized recommendation.

Later, Yu *et al*. [59] proposed HeteRec-p, which further considers the importance of different meta-paths should vary for different users. HeteRec-p first clusters users into c groups based on their past behaviors, then generates personalized recommendation with the clustering information, instead of applying a global preference model. The modified scoring function becomes

$$\hat{y}_{i,j} = \sum_{k=1}^{c} \text{sim}(\mathbf{C}_k, \mathbf{u}_i) \sum_{l=1}^{L} \theta_l^k \cdot \hat{\mathbf{u}}_i^{(l)T} \hat{\mathbf{v}}_j^{(l)}, \tag{22}$$

where $\text{sim}(\mathbf{C}_k, \mathbf{u}_i)$ denotes the cosine similarity between user u_i and the target user group C_k, and θ_l^k denotes the importance of meta-path l for the user group k. Therefore, recommendations for target users are more personalized compared with HeteRec.

The aforementioned methods first learn latent vectors of users and items from the interaction matrix and their mutual meta-structure based similarities, then make predictions based on enhanced representations. Another approach is to predict the preference for unrated items with the weighted ensemble of similar user's ratings directly. Shi *et al*.[61] proposed the SemRec which considers the explicit rating scores in the range of 0 to N, instead of implicit interaction records. Ratings for candidate items are predicted by the weighted ensemble of similar user's rating, where user similarities are learned from different meta-paths. In each

meta-path, SemRec defines rating intensity matrix $Q^{(l)} \in \mathbf{R}^{|U|\times|I|\times N}$, where $Q^{(l)}_{u,v,n} = \sum_{u'} E_{u',v,n} \times S^{(l)}_{u,u'}$, $E_{u',v,n}$ is an indicator function that shows whether user u' has rated the item v with score n or not, and $S^{(l)}_{u,u'}$ is the similarity between target user u and user u' along the meta-path l. The target user's score for item v along the meta-path l can be calculated via

$$\hat{R}^{(l)}_{u,v} = \sum_{n=1}^{N} n \times \frac{Q^{(l)}_{u,v,n}}{\sum_{k=1}^{N} Q^{(l)}_{u,v,k}}. \tag{23}$$

The final estimated score $\hat{R}_{u,v}$ can be obtained by considering the importance of different path for the user

$$\hat{R}_{u,v} = \sum_{l=1}^{L} W^{(l)}_u \times \hat{R}^{(l)}_{u,v}, \tag{24}$$

where $W^{(l)}_u$ is the preference weight on the lth meta-path.

One limitation of previous methods is that meta-path is not able to characterize rich semantics. For example, the relation of "two users both buy and review the same product" cannot be depicted by a meta-path. Zhao *et al.* [63] designed FMG by replacing the meta-path with the meta-graph to capture complicated relations between entities in the heterogeneous graph. Similar to HeteRec, by designing L meta-graphs, they got L different user-item similarity matrices, which are further decomposed to L latent vectors of user-item pairs, denoted as $\left(\hat{\mathbf{U}}^{(l)}, \hat{\mathbf{V}}^{(l)}\right), l = 1, 2, \ldots, L$. Next, the factorization machine (FM) is applied to fuse the features of users i and items j across different meta-graphs for computing preference score $\hat{y}_{i,j}$. The FM considers the interaction of entities along different meta-graphs, which can further exploit connectivity patterns.

Meta-structure based methods are explainable since these manually designed meta-structures provide the reference for the recommendation by matching the meta-structure between the candidate item and the interacted item or the target user. For example, if the recommended item shares connectivity pattern $\text{User} \xrightarrow{\text{Buy}} \text{Item}_1 \xrightarrow{\text{BelongTo}} \text{Category} \xleftarrow{\text{BelongTo}} \text{Item}_2$ with an item in the interaction history, it can be indicated that the item is recommended because it is similar to historical interactions.

The meta-structure based method is easy to implement, and most works are based on MF techniques with relatively low model complexity. However, the selection of meta-path or meta-graph requires domain knowledge, and these meta-structures may vary significantly for different datasets. Moreover, it may not be suitable to apply the meta-structure based method under some specific scenarios. For example, in the news recommendation task, entities that belong to one news may belong to different domains, which makes it difficult to design meta-paths.

3.2.2 Path-Embedding Based Methods

One issue in the meta-structure based method is that the connection pattern is not explicitly modeled, which makes it hard to learn the mutual effect between the user-item pair and the connection pattern. Recently, the path-embedding based method has been proposed to explicitly learn the embedding of connection patterns. Some frameworks learn the explicit embedding of paths that connect user-item pairs in the user-item KG, or item-item pairs in the item KG, in order to directly model the user-item or item-item relations. Take the relation modeling in the user-item KG as an example, assume there are K paths that connect u_i and v_j in the KG, the embedding of path p is represented as $\mathbf{h}_p$. Then, the final representation of the interaction between u_i and v_j can be obtained via

$$\mathbf{h} = g(\mathbf{h}_p), p = 1, 2, \ldots, K, \tag{25}$$

where $g(\cdot)$ is the function to summarize the information from each path embedding, and the common choice is a max-pooling operation or weighted sum operation. Then, u_i's preference for the v_j can be modeled via

$$\hat{y}_{i,j} = f(\mathbf{u}_i, \mathbf{v}_j, \mathbf{h}), \tag{26}$$

where $f(\cdot)$ is the function to map the representation of the interaction between the user-item pair as well as the embedding of the user-item pair to a preference score.

For instance, Hu *et al.* [1] proposed MCRec, which learns the explicit representations of meta-paths to depict the interaction context of user-item pairs. For each u_i and v_j, MCRec defines L meta-paths that connects u_i and v_j and samples K path instances for each meta-path, where each path instance $\mathbf{X}^p \in \mathbb{R}^{L\times d}$ consists of L entities $e_i \in \mathbb{R}^d$. It first learns the embedding for each path instance with CNN, then calculates the meta-path embedding by applying the max-pooling operation on embeddings of K path instances that belong to its class. Next, the interaction embedding $\mathbf{h}$ between the user and item is obtained with the weighted average of these meta-path embeddings. Another novelty of MCRec is that the user embedding and the item embedding are updated with the interaction embedding:

$$\begin{aligned} \beta_{u_i} &= \text{ReLU}\left(\mathbf{W}_1\mathbf{u}_i + \mathbf{W}_2\mathbf{h} + \mathbf{b}_{u_i}\right), \quad \tilde{\mathbf{u}}_i = \beta_{u_i} \odot \mathbf{u}_i, \\ \beta_{v_j} &= \text{ReLU}\left(\mathbf{W}'_1\mathbf{v}_j + \mathbf{W}'_2\mathbf{h} + \mathbf{b}'_{v_j}\right), \quad \tilde{\mathbf{v}}_j = \beta_{v_j} \odot \mathbf{v}_j. \end{aligned} \tag{27}$$

Finally, the preference score is calculated by following $\hat{y}_{i,j} = \text{MLP}(\tilde{\mathbf{u}}_i \oplus \tilde{\mathbf{v}}_j \oplus \mathbf{h})$. The recommendations can be interpreted by checking the weight of each meta-path. A higher meta-path weight means such a relation between the target user and the candidate item is more important in making the decision. However, MCRec still needs to manually define the type and number of meta-paths, which is tedious and requires domain knowledge.

Sun *et al.* [64] proposed a recurrent knowledge graph embedding (RKGE) approach that mines the path relation between user u_i and item v_j automatically, without predefined meta-paths. Specifically, RKGE first enumerates user-to-item paths $\mathcal{P}(u_i, v_j)$ that connects u_i and v_j with different semantic relations under a sequence length constraint. They designed a recurrent network which takes as input the path instance p consists of entity embeddings, and used the final hidden state $\mathbf{h}_p$ as the representation of the entire path. Next, following Equation (25), final hidden states $\mathbf{h}_p$ of all these paths are aggregated via the average-pooling operation to model the semantic relation $\mathbf{h}$ between u_i and v_j. Finally, the preference of u_i for v_j is estimated with $\mathbf{h}$, and Equation (26) becomes $\hat{y}_{i,j} = \sigma(\mathbf{W}_h\mathbf{h} + b)$, where $\sigma(\cdot)$ is the

sigmoid function. After the training stage, better representations of users and items can be achieved. In the prediction stage, they used the inner product of the user embedding and the item embedding, since the inner product is more efficient. Entities in the KG serve as the bridge connecting items in the recommendation list and the interaction history, which can provide explanations for recommendation from different angles. Similarly, Wang *et al.* [65] proposed a knowledge-aware path recurrent network (KPRN) solution. The major difference is that KPRN constructs the path sequence with both the entity embedding and the relation embedding. Moreover, they first calculated the preference score based on each path, then aggregated these scores for final preference estimation. The preference score on each path can reflect the relative importance of each path that connects the target user and the candidate item. Therefore, it can provide the precise relation-path-level explanation for each item.

Besides the relation modeling between the user-item pair, the relation embedding between the item-item pair can also be utilized. Ma *et al.* [67] proposed RuleRec to model connection patterns of associated items (co-buy, co-view, etc.) in an external item KG into rule features. RuleRec jointly trains a rule learning module and an item recommendation module. The rule learning module first links items with associated entities in an external KG, which forms an item KG. Then this module summarizes common connection patterns that connect associated items, which are represented as several rules. The corresponding weight for each rule is further learned with the recommendation module jointly, which forms a feature vector where each entry is the rule value given two items in the KG. The final recommendation is obtained by considering the user representation, the candidate item representation, and the rule feature vector between the candidate item and interacted items. The explanation can be offered from the human-understandable rules and corresponding rule weights.

Xian *et al.* [68] proposed Policy-Guided Path Reasoning (PGPR) to use reinforcement learning (RL) to search for reasonable paths between user-item pairs automatically. They formulated the recommendation problem as a Markov decision process to find a reasonable path connecting the user-item pair in the KG. They trained an agent to walk from users to items by designing the path searching algorithm, the transition strategy, terminal conditions, and RL rewards, so that high rewards will be given to the correct user-item pairs. The benefit of adopting RL is that the actual path that connects the user and the recommended item is available, which makes the system more transparent.

The path-embedding based method encodes the connection pattern of user-item pair or item-item pair into latent vectors, which makes it possible to consider the mutual effect of the target user, the candidate item, and the connection pattern. In addition, most models are able to mine the connection patterns automatically by numerating qualified paths and selecting salient ones, without the assistance of predefined meta-structures. Therefore, it is likely to capture expressive connection patterns. However, the number of possible paths in the graph can grow to a large number if relations in the graph are complex. In reality, it is impossible to exploit all the paths for each entity pair in large-scale KGs, which may hinder the performance of the model.

3.2.3 Summary for Connection-Based Method

In this section, we compare the meta-structure based method and the path-embedding based method, as shown in Table 4.

The connection-based method relies heavily on the connection patterns. However, the representation ability of the meta-path is limited, which hinders the performance of traditional meta-structure based methods. FMG [63] solves this issue by replacing the meta-path with the meta-graph to capture richer semantics in the graph. Path-embedding based methods further overcome another shortcoming of meta-structure based methods that domain knowledge and handcrafted paths are required. These methods enumerate possible paths and explicitly model the relation between user-item pairs or item-item pairs. However, the scalability is sacrificed to some extent in the path-embedding based method, since these models are relatively complex, and more computations are required in enumerating paths and learning representations.

3.3 Propagation-Based Method

Embedding-based methods leverage the semantic relations in the KG to enrich the representations of users and items, or to regularize the recommendation, but it is difficult to capture high-order relations between entities. Connection-based methods use the connectivity information in the graph to guide recommendation, however, it is unavoidable to lose information by decomposing the sophisticated user-item connection pattern into separate linear paths. To fully exploit the information in the KG, propagation-based methods have been proposed to integrate both the representation of entities and relations, and the high-order connection patterns for a more personalized recommendation. The propagation-based method is based on the idea of embedding propagation, where the common implementation is based on the GNN technique. These methods refine the entity representation by aggregating embeddings of multi-hop neighbors in the KG. Then, the user's preference can be predicted with the enriched representations of user u_i and the potential item v_j. We subdivide this category according to which type of entity is refined in the message propagation procedure. The challenges of this method include: 1) how to assign proper weights to different neighbors; 2) how to propagate messages on different relation edges; 3) how to improve the scalability of models.

3.3.1 Refinement of User Representation

The first group of works refine the user representation based on their interaction history. These works build the item KG that connects both interacted items and candidate items with multiple relations. The motivation is that users can be represented as the combination of their interacted items as well as multi-hop neighbors of these items. In detail, items in the interaction history are selected as seeds of the propagation process. Then, multi-hop triplet sets $\mathcal{S}_{u_i}^k (k = 1, 2, \ldots, H)$ are extracted along links in the graph, where $\mathcal{S}_{u_i}^1$ is the triplet set (e_h, r, e_t) with the head entities being the user u_i's engaged items. The process of learning user representation $\mathbf{u_i}$ can be formulated as

TABLE 4
Comparisons Between Connection-Based Methods

	Main advantages	Main shortcomings
MSB.	∗ easy to implement ∗ scalability	∗ handcrafted meta-paths with limited semantics ∗ domain knowledge required
PEB.	∗ richer semantics ∗ mutual effect of path and user-item pair involved	∗ scalability

In the table, "MSB." stands for the meta-structure based method, and "PEB." stands for the path-embedding based method.

1. Calculate the user representation $\mathbf{o}_u^k$ by aggregating entities in each layer of the triplet set $\mathcal{S}_{u_i}^k (k = 1, 2, \ldots, H)$.
2. Combine $\mathbf{o}_u^k (k = 1, 2, \ldots, H)$ for final user representation $\mathbf{o}_u$.

Since the propagation starts from the user's engaged items and ends with distant neighbors, this process can be regarded as propagating the user's preference layer by layer outwardly in the item KG. Therefore, these methods can be interpreted as propagating the user's preference from historical interests along paths in the KG.

Wang *et al.* [16] proposed RippleNet that first introduces the preference propagation mechanism in the KG. It trains a relation matrix $\mathbf{R}_i \in \mathbb{R}^{d \times d}$ to assign weights for neighbors in the graph. In each layer of the triplet set, the aggregation process can be illustrated as follows. First, representations of the head entity $\mathbf{e}_{h_i} \in \mathbb{R}^d$, the relation matrix $\mathbf{R}_i$, and the candidate item $\mathbf{v}_j \in \mathbb{R}^d$ are used to calculate the weight p_i for tail entity in the corresponding triplet by following Equation (28). During this process, the similarities of the candidate item v_j and multi-hop neighbors of interacted items are calculated in the relation space. Second, user's representation $\mathbf{o}_{u_i}^h \in \mathbb{R}^d$ in the hth layer of triplet set can be calculated with the weighted average of tail entity embeddings $\mathbf{e}_{t_i} \in \mathbb{R}^d$ via Equation (29)

$$p_i^h = \frac{\exp\left(\mathbf{v}_j^T \mathbf{R}_i^h \mathbf{e}_{h_i}^h\right)}{\sum_{(e_{h_k}, r_k, e_{t_k}) \in \mathcal{S}_{u_i}^h} \exp\left(\mathbf{v}_j^T \mathbf{R}_k^h \mathbf{e}_{h_k}^h\right)}, \tag{28}$$

$$\mathbf{o}_{u_i}^h = \sum_{\left(e_{h_i}, r_i, e_{t_i}\right) \in \mathcal{S}_{u_i}^h} p_i^h \mathbf{e}_{t_i}^h, \tag{29}$$

where the candidate item embedding is initialized with $\mathbf{v}_j^{initial}$ in the first layer of the triplet set, and is replaced with the $\mathbf{o}_{u_i}^{h-1}$ in the hth layer of the triplet set. By repeating the process in Equations (28)-(29) from $h = 1, 2, \ldots, H$ iteratively, user's preference is propagated from interacted item to distant neighbors in the graph. The final representation of u_i is the combination of user representations in each layer of triplet set with the equation of $\mathbf{u}_i = \mathbf{o}_{u_i}^1 + \mathbf{o}_{u_i}^2 + \cdots + \mathbf{o}_{u_i}^H$. Finally, the preference score can be generated via

$$\hat{y}_{i,j} = \sigma\left(\mathbf{u}_i^T \mathbf{v}_j^{initial}\right), \tag{30}$$

where $\sigma(x)$ is the sigmoid function. However, the preference matrix $\mathbf{R}_i$ is hard to train. As a consequence, the recommendation results suffer from unrelated entities. Moreover, the size of triplet sets can be extremely large as the layer increases, which hinders the scalability.

Tang *et al.* [70] proposed AKUPM with a similar preference propagation mechanism in the RippleNet. The difference is that AKUPM models entities with TransR [85] to assign entities with different representations under various relations. In addition, AKUPM applies the self-attention mechanism [90] to assign weights for entities in the aggregation process, and to concatenate the user representation obtained in each layer. In the entity aggregation process, the Query $\mathbf{Q}_{u_i}^h$, Key $\mathbf{K}_{u_i}^h$, Value $\mathbf{V}_{u_i}^h$ are combined with the representation of the head entity and the corresponding relation, then the user representations $\mathbf{o}_{u_i}^h$ in hth layer is calculated via Equation (31)

$$\begin{aligned} \mathbf{V}_{u_i}^h &= \mathbf{Q}_{u_i}^h = \mathbf{K}_{u_i}^h = \left[\mathbf{R}_1^h \mathbf{e}_{h_1}^h, \mathbf{R}_2^h \mathbf{e}_{h_2}^h, \ldots, \mathbf{R}_N^h \mathbf{e}_{h_N}^h\right], \\ \mathbf{o}_{u_i}^h &= \mathbf{V}_{u_i}^h softmax\left(\frac{\mathbf{Q}_{u_i}{}^h \mathbf{K}_{u_i}^h}{\sqrt{d}}\right), \end{aligned} \tag{31}$$

where d is applied to scale the matrix for stability. To improve the efficiency, the maximum number of entities to be aggregated in each layer is set to N. Finally, user u_i's representation in each layer are concatenated with different importance and forms the final user representation $\mathbf{u}$ via Equation (32)

$$\begin{aligned} \mathbf{Q}_{u_i} &= \mathbf{v}_j, \\ \mathbf{V}_{u_i} &= \mathbf{K}_{u_i} = \left[\mathbf{o}_{u_i}^1, \mathbf{o}_{u_i}^2, \ldots, \mathbf{o}_{u_i}^H\right], \\ \mathbf{u}_i &= \mathbf{V}_{u_i}^h softmax\left(\frac{\mathbf{Q}_{u_i}{}^h \mathbf{K}_{u_i}^h}{\sqrt{d}}\right), \end{aligned} \tag{32}$$

where d is applied to scale the matrix for stability, $\mathbf{v}_j$ is the representation of the candidate item v_j. Then it follows Equation (30) to predict the user preference. With the self-attention mechanism, AKUPM can figure out related items for the user and capture the user's interest better.

In these methods, the edge weight is explicit in the item KG. Therefore, the salient path that connects the candidate item and the interacted item can be selected and serve as the explanation for the recommendation results. Although these works utilize both the entity embedding and the high order connection information, only the user representation gets updated during the propagation process.

3.3.2 Refinement of Item Representation

The above-mentioned works refine user representation by aggregating entities outwardly in the graph. Another solution is to learn high order representation of the candidate item $\mathbf{v}_j$ by aggregating embeddings of item v_j's multi-hop neighbors $\mathcal{N}_v^k (k = 1, 2, \ldots, H)$ inwardly in the item KG. During the inward propagation process, the graph attention mechanism is adopted, where the weight of different neighbors is user-specific and relation-specific. The motivation is that a user will have distinct preferences for different relations, which can guide the information flow in the KG. Each

round of the propagation procedure can be illustrated as follows:

1. Aggregate neighbors of an entity e_i

$$\mathbf{e}_{\mathcal{N}_i}^{h-1} = \mathrm{AGG}(\mathbf{e}_m^{h-1}), \forall m \in \mathcal{N}_i, h = 1, 2, \ldots, H. \tag{33}$$

2. Update the h-order representation of the entity with $h-1$-order neighbor embedding and self embedding

$$\mathbf{e}_i^h = g(\mathbf{e}_{\mathcal{N}_i}^{h-1}, \mathbf{e}_i^{h-1}), h = 1, 2, \ldots, H. \tag{34}$$

Note that $\mathbf{e}_i^0$ stands for the initial representation of the entity, and $\mathbf{e}_i^h$ stands for the h-order representation of entity e_i, which is a mixture of entity initial representation and representations from h-hop neighbors. The aggregation function maps N neighbors to a vector $\in \mathbb{R}^d$, and the update function $g(\cdot)$ is a non-linear function: $\mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^d$. By repeating this process for H times iteratively, the representation of the candidate item contains information from H-hop neighbors.

Wang *et al.* [72] proposed KGCN, in which the weight of each neighbor $m \in \mathcal{N}_i$ is user-specific. It is calculated with the user representation $\mathbf{u}$ and the entity relation $\mathbf{r}$, which are all trainable parameters

$$w_m = \mathbf{u}^T \mathbf{r}. \tag{35}$$

Then, the weight of each neighbor is normalized, and forms the $h-1$-order representation for entity e_i's neighbors via

$$\tilde{w}_m = \frac{\exp(w_m)}{\sum_{m \in \mathcal{N}(i)} \exp(w_m)}, \quad \mathbf{e}_{\mathcal{N}(i)}^{h-1} = \sum_{m \in \mathcal{N}(i)} \tilde{w}_m \mathbf{e}_m^{h-1}. \tag{36}$$

As for the entity representation update part, it designs different update functions to concatenate neighbors. In the hth round of the propagation process, entities within $H-h+1$ hops of the candidate items are updated by following the Equations (33) and (34). In each round of update, the candidate item embedding $\mathbf{e}_{v_i}$ mixes with neighbors one hop further, and finally receives the information from H-hop neighbors. For computational efficiency, KGCN uniformly samples a fixed-size of neighbors for each entity in the constructed item KG, which is favorable for large-scale datasets and KGs.

However, KGCN is prone to overfitting, since the user-item interaction is the only supervised signal for the whole framework. Later, Wang *et al.* [73] proposed a follow-up approach, KGCN-LS, which further adds a label smoothness (LS) regularization on the KGCN model. If the user u interacts with item v in the extracted entities, v should have a label $l_u(v) = 1$, otherwise $l_u(v) = 0$. In this work, they mapped the user preference for each entity pair in the extracted entity sets ε into a preference matrix $\mathbf{A}_u$, where $\mathbf{A}_u[i,j] = s_u(r_{e_i,e_j})$, the user preference score for entity e_i and e_j connected with the relation r_{e_i,e_j}. Then, the framework tries to minimize the following energy function with the assumption that adjacent entities in the KG may have similar labels

$$E(l_u, \mathbf{A}_u) = \frac{1}{2} \sum_{e_i \in \mathcal{E}, e_j \in \mathcal{E}} \mathbf{A}_u[i,j] \big(l_u(e_i) - l_u\big(e_j\big)\big)^2. \tag{37}$$

Next, it propagates the interaction labels in the KG and estimates the label $\hat{l}_u(v)$ for the candidate item v. The loss function for the label propagation part is

$$L = \sum_u \sum_v J\Big(y_{uv}, \hat{l}_u(v)\Big), \tag{38}$$

where $J(\cdot)$ is the cross-entropy loss function, y_{uv} is the true label for user-item pair. The label propagation module and the preference propagation module are jointly trained, and further improve the recommendation results.

Based on the item KG, these papers refine the item representation with inward propagation in the item KG. However, similar to user refinement with outward aggregation in the KG, only one type of entity is refined.

3.3.3 Refinement of Both User and Item Representation

Recently, some papers have explored the propagation mechanism in the user-item KG. Both users, items and their associated entities are connected in one graph, and the interaction between user-item pairs serves as one type of relation. The user embedding and the item embedding can be refined with their corresponding neighbors during the propagation process, as illustrated in Equations (33) and (34).

Wang *et al.* [74] proposed KGAT, which directly models the high order relations between users and items with embedding propagation. KGAT first applies TransR [85] to obtain the initial representation for entities. The neighbor aggregation process is similar to KGCN, though it utilizes the following nonlinear activation equation for each triplet $(h,r,t) \in \mathcal{G}$ to calculate the weight for aggregation:

$$w(h,r,t) = (\mathbf{W}_r \mathbf{e}_t)^T \tanh(\mathbf{W}_r \mathbf{e}_h + \mathbf{e}_r). \tag{39}$$

After H-layer-propagation, different order representations of the user $\mathbf{e}_u^{(0)}, \mathbf{e}_u^{(1)}, \ldots, \mathbf{e}_u^{(H)}$ will be obtained. These representations are concatenated and forms the final user representation $\mathbf{e}_u^*$. The final representation of the candidate item $\mathbf{e}_v^*$ can be calculated in a similar approach. Finally, the user preference can be calculated via $\hat{y}_{u,v} = \mathbf{e}_u^{*T} \mathbf{e}_v^*$.

Qu *et al.* [75] proposed KNI, which further considers the interaction between item-side neighbors and user-side neighbors. After obtaining high-order representation $\mathbf{e}^{(H)}$ of entities in the KG, instead of using the enhanced user and item embedding to predict the user preference, KNI leverages the enhanced representation of user neighbors N_u and item neighbors N_v for preference estimation. The motivation is that items $i \in N_u$ and entities $j \in N_v$ share interactive patterns. They designed an attention network to assign a proper weight for each neighbor pair to model the user preference $\hat{y}_{u,v}$

$$\begin{aligned} \alpha_{i,j} &= \mathrm{softmax}_{i,j}\Big(\mathbf{W}\Big[\mathbf{e}_u^{(H)}, \mathbf{e}_i^{(H)}, \mathbf{e}_v^{(H)}, \mathbf{e}_j^{(H)}\Big] + b\Big), \\ \hat{y}_{u,v} &= \sum_{i \in N_u} \sum_{j \in N_v} \alpha_{i,j} \big(\mathbf{e}_i^{(H)}\big)^T \mathbf{e}_j^{(H)}. \end{aligned} \tag{40}$$

Zhao *et al.* [76] proposed IntentGC, a scalable recommendation framework. To improve efficiency, IntentGC translates the original user-item KG into two user-user and item-

TABLE 5
Comparisons Between Propagation-Based Methods

	Main advantages	Main shortcomings
RU.	–	∗ only one type of entity gets refined ∗ scalability
RI.	–	∗ only one type of entity gets refined
RUI.	∗ refine both user and item representations	∗ more irrelevant entities

In the table, "RU." stands for refinement of the user, "RI." stands for refinement of the item, "RUI." stands for refinement of the user and item, and "–" stands for this method has no comparative advantage.

item multi-relational graphs. IntentGC calculates the second-order proximity of two users and items under each type of auxiliary entity in the original graph, and converts it to a relation in the translated graph. The user representation and the item representation are refined in the user graph and item graph separately with the propagation mechanism as illustrated in Equations (33) and (34). Moreover, they designed a faster graph convolution function to update high-order entity representation, with the assumption that the interaction between features in different dimensions of the entity and its neighbors are meaningless. Take the user side update as an example, the update function (34) becomes:

$$\begin{aligned}\mathbf{g}_u^{h-1}(i) &= \Phi\left(w_u^{h-1}(i,1)\cdot\mathbf{e}_u^{h-1} + \sum_{r=2}^{R} w_u^{h-1}(i,r)\cdot\mathbf{e}_{\mathcal{N}_u^r}^{h-1}\right),\\ \mathbf{e}_u^{h} &= \Phi\left(\sum_{i=1}^{D}\theta_i^{h-1}\cdot\mathbf{g}_u^{h-1}(i)\right),\end{aligned} \tag{41}$$

where r is the rth type of relation, $w_u^{h-1}(i,j)$ is the weight for the feature in the ith dimension, $\mathcal{N}_u^r$ is the neighbor with the rth type of relation, θ_i^{h-1} is the importance for the ith feature vector.

One limitation of previous methods is that they may introduce irrelevant neighbors in the propagation process. To overcome this issue, Sha *et al.* [77] proposed AKGE, which learns enhanced representation of user u_i and candidate item v_j by propagating information in a subgraph of this user-item pair. AKGE first pre-trains the embeddings of entities in the graph with TransR [85], then samples several salient paths connecting u_i and v_j based on the pairwise distance in these paths, which forms a subgraph for u_i and v_j. Next, AKGE updates high-order entity representation with a gating mechanism that is similar to the gated recurrent units [91] to better control the information flow in the graph. The construction of the subgraph filters out less related entities in the graph, facilitating mining high-order user-item relations for recommendation.

Similar to propagation in the item graph, the weight of edges in the user-item graph is also user-specific. Therefore, these models can offer explanations for recommendation results by checking the salient paths that connect the target user and the candidate item. As the user is incorporated as one type of node, the explanation is more intuitive, since the contribution of each interacted item is available.

By incorporating users into the KG, the high-order connection pattern can be explored to a greater extent. The downside is that more relations in the graph will bring irrelevant entities, which may mislead the user's preference in the aggregation process.

TABLE 6
Comparisons for Methods Across Categories

	Main advantages	Main shortcomings
Emb.	∗ flexible	∗ ignore high-order relation between entities ∗ hard to interpret
Conn.	∗ interpretable	∗ data-sparsity issue ∗ information loss in decomposing connection pattern to meta-structure
Prop.	∗ interpretable ∗ fully exploit information	∗ scalability

In the table, "Emb." stands for embedding-based method, "Conn." stands for connection-based method, and "Prop." stands for propagation-based method.

3.3.4 Summary for Propagation-Based Method

In Table 5, we list the main advantages and shortcomings of different implementation of propagation-based methods.

Propgation-based methods are usually computational costly. As the graph grows large, it becomes difficult for the model to converge. To improve the efficiency, faster graph convolutional operation has been proposed in [76], and it is common to apply neighbor sampling in each layer [72], [73], [74], [77]. However, the random sampling will inevitably lead to loss of information, failing to fully explore the knowledge in the graph.

3.4 Summary

In this section, we provide qualitative comparisons for methods across categories and necessary explanations to illustrate their advantages and shortcomings in Table 6. Moreover, we summarize common techniques for the explainable recommendation and compare the performance of popular KGE methods, so as to provide readers with practical suggestions.

The embedding-based method is the most flexible approach. On the one hand, it is relatively easy to encode the KG with the KGE module, and the learned embedding can be naturally incorporated into the user representation or item representation. While in the connection-based method, it can be tedious to define meta-path or meta-graph in the graph; as for the propagation-based method, the aggregation and update part need to be carefully designed. On the other hand, the embedding-based method is suitable for most application scenarios, due to the external knowledge is usually available in different tasks. On the contrary, in the meta-structure based method, meta-paths are usually diverse for different application scenarios and cannot generalize to new datasets. Also, for specific scenarios, like news recommendation, it is hard to define meta-path and apply the meta-structure based method. Meanwhile, both the path-embedding based method and the propagation-based method are impropriate for the recommendation scenarios

with large-scale datasets, since the computational complexity may grow large in enumerating paths and neighbors. Moreover, the quality and the quantity of paths are crucial for connection-based methods, therefore, sparse datasets may not provide enough paths to mine relations and model interests for this type of method. Nevertheless, both the embedding-based method and the connection-based method fail to fully explore information in the KG. With the development of GNN techniques, the propagation-based method has become a new research trend in recent years. In addition, both the connection-based method and the propagation-based method can be interpreted with paths in the KG, while the embedding-based method is less intuitive to interpret.

To facilitate readers understanding how to leverage KG for the explainable recommendation, we summarize common techniques in our collected papers.

• *Attention Mechanism on Relation Embedding.* This approach is adopted in embedding-based methods [48], [54], [55]. The attention mechanism is applied to the embedding of relations between entities in the KG. From the attention weight of different relations, the significance of each type of item attribute for the target user is available. Therefore, this technique could provide the preference-level explanation for the recommendation.

• *Defining Meta-Path/Meta-Graph.* The relation between the selected item and the target user or interacted items can be decomposed into the combination of several meta-paths or meta-graphs. By translating the meta-path or meta-graph into understandable rules, the explanation can be offered by the system [56], [57], [58], [59], [60], [61], [62], [63].

• *Attention Mechanism on Path Embedding.* For path-embedding methods, the weight of a specific path that connects the target user and candidate item is available with the attention mechanism. The weight of each path can represent the relative importance of each path for the user. Therefore, explanations can be provided based on salient paths in the graph [1], [64], [65], [66].

• *Reinforcement Learning in User-Item KG.* By training an agent in the user-item KG with the reinforcement learning technique, the actual path that connects the uer-item pair can be mined [68], [69]. It can directly show the reasoning process in the KG instead of finding a post-hoc explanation for the already chosen recommendations. Thus, the reasoning process is precise and trustworthy for the target user.

• *Extracting Edge Weight.* The propagation-based method requires assigning the user-specific weight for each type of neighbor in the aggregation process. The edge weight controls the information flow between entities in the graph, and can reflect the importance of each type of relation in the KG. Moreover, edge weights between entities in the KG are also available from the attention weight or learned relation matrix. Therefore, it is possible to generate explanations by finding salient paths that connect the candidate item and the target user, or the interacted items in multi-hop neighbors [16], [70], [71], [72], [73], [74], [77].

The KGE method is required in most works to encode the semantic relations between entities in the KG. Distinct KGE methods will lead to different performances in models, thus, we summarize and compare various KGE strategies. In the two-stage learning method, the entity embedding is learned in the KGE module separately. While in other categories of works, the entity embedding can be randomly initialized and jointly trained with the recommendation module. However, it has been reported that replacing the randomly initialized vector with pre-trained KG embedding learned from the KGE model can improve the recommendation performance [74]. Here, we collect empirical analysis of the effect of different KGE methods on recommendation results. The most commonly used methods are translational distance models, including TransE [81], TransH [86], TransR [85], and TransD [79]. TransE provides the most strict constraint of entity representations. Given a triplet $(e_h, r, e_t) \in \mathcal{G}$, TransE assumes the entity embedding and the relation embedding have the relation $\mathbf{e}_h + \mathbf{r} \approx \mathbf{e}_t$. Although TransE is simple and efficient, it has flaws in capturing the many-to-many relation. TransH overcomes the issue by allowing the head entity and tail entity to have different representations under various relations. TransR relaxes the entity relation by modeling the entity embedding and the relation embedding in separate spaces connected with a projection matrix. TransD further allows the projection matrix to consider both the relation type and entity type. Based on the empirical results in [27], [77], [92], flexible models such as TransD, TransR, and TransH outperform the TransE model. However, a higher degree of flexibility in KGE methods not necessarily bring improvement [27], [92]. The optimal KGE method may be related to specific datasets and recommendation frameworks. Moreover, these translational distance models are only suitable for a direct graph. For an undirected KG, there may exist the relation $(movie1, share\ director, movie2)$. To learn the embedding for an undirected graph, semantic matching models are commonly used. For example, RCF adopts DistMult [55] to learn the graph embedding.

4 Datasets of Recommender Systems With Knowledge Graph

Besides the benefit of accuracy and interpretability, another advantage of KG-based recommendation is that this type of side information can be naturally incorporated into recommender systems for different applications. To show the effectiveness of the KG as side information, KG-based recommender systems have been evaluated on datasets under different scenarios. In this section, we categorize these works based on the dataset and illustrate the difference among these scenarios. The contributions of this section are two-fold. First, we provide an overview of datasets used in various scenarios. Second, we illustrate how knowledge graphs are constructed for different recommendation tasks. This section can help researchers find suitable datasets to test their recommender systems.

We collect datasets used in investigated papers, and summarize the basic statistical information of the dataset (the number of users, the number of items, the density of the dataset), how papers construct the KG, as well as the commonly selected knowledge base to supplement external knowledge for each dataset in Table 7. For the statistical information of the dataset, we list the information in its original description if it is available. In practice, it is common to select a subset of the large dataset, and filter out users and

TABLE 7
A Collection of Datasets for Different Application Scenarios and Corresponding Papers, Where "⋆" Stands for the Dataset is Public Available, "# U" Stands for the Number of User, "# I" Stands for the Number of Item, "IKG" Stands for Item KG, "UIKG" Stands for User-Item KG, "Exernal KB" Stands for the External Knowledge Base, "–" Stands for the Specific Number is Not Public, and the Blank Entries Stand for no Paper in the Survey Belongs to This Category, or no External Knowledge Base has been Used for This Dataset in Our Collected Papers

Scenario	Data	# U	# I	Density	KG Type		External KB
					IKG	UIKG	
Movie	MovieLens-100K ⋆	943	1682	6.3%	[55], [56]	[1], [58], [59], [60]	DBPedia, Freebase, Satori, IMDb
	MovieLens-1M ⋆	6040	3900	4.25%	[14], [16], [48], [53], [54], [70], [71]	[46], [64], [65], [69], [75], [77]	
	MovieLens-20M ⋆	138493	27278	0.53%	[48], [72], [73]	[66], [75]	
	DoubanMovie ⋆	42000	5000	0.44%	[49]		CN-DBPedia
Book	Book-Crossing ⋆	278858	271379	0.0015%	[16], [53], [70], [71], [72], [73]	[75]	Satori
	Amazon-Book ⋆	8.2M	1.6M	0.0002%	[48]	[74], [75]	Freebase
	DBbook2014	5576	2680	0.44%	[54]	[69]	DBPedia
	IntentBooks	92564	18475	0.053%	[14]		Satori
	DoubanBook	13042	22347	0.27%		[62]	
Music	Last.FM-a ⋆	1872	17632	0.28%	[53], [72], [73]	[69]	Satori
	Last.FM-b ⋆	120322	32.3M	0.028%	[48]	[74], [77]	Freebase
	KKBox ⋆	34403	2.3M	0.0047%	[55]	[65]	
News	Bing-News	141487	535145	0.0014%	[16], [27], [53]		Satori
	MIND ⋆	21M	161K	0.015%			Wikidata
Products	Amazon Review ⋆	21M	5.9M	0.00012%	[67]	[15], [52], [63], [68], [76]	Freebase
	Alibaba Taobao	–	–	–	[50]	[76]	
POI	Yelp Challenge ⋆	43876	11537	0.045%		[1], [58], [59], [60], [62], [63], [64], [74], [77]	
	Dianping-Food	–	–	–	[73]		
	CEM	26019	119	4.93%		[47]	STDs
Social	Weibo	12814	12814	0.043%	[51]		Satori
	DBLP	–	–	–		[57]	
	MeetUp	–	–	–		[57]	

items with less than k records for higher data quality, thus, the statistical information of the dataset in different works might be slightly different. Eventually, these works can be categorized into seven application scenarios, and we will illustrate the characteristics of each scenario.

• *Movie.* The advantage of movie recommendation is that there are multiple external knowledge bases contain knowledge in the movie domain. Moreover, it is easy to map the movie title into the external knowledge base, from which extracts subgraphs with the movie's extra knowledge, such as the genre, the actor, the director, and the country. The most commonly used datasets are the MovieLens benchmark datasets [93] which are collected from the MovieLens website [94], including MovieLens-100K, MovieLens-1M, and MovieLens-20M with a different number of ratings. Each dataset contains ratings, the movie's attributes, and the user's profile. Besides the MovieLens dataset, there is also the DoubanMovie dataset [49] crawled from Douban [95], a popular Chinese social media network. DoubanMovie maintains the movie tag, and can link the movie title with the entity in the Chines KG, CN-DBPedia, to enrich the representation of items. Among these datasets, MovieLens-100K is the smallest one, which is generally evaluated by some early works and works with high computational complexity. MovieLens-1M is the most popular one with a balanced rating number and density, while the MovieLens-20M is the largest one, which is suitable for verifying the scalability of the model. Note that the movie recommendation datasets are much denser than other scenarios.

• *Book.* The book recommendation is another popular task. Similar to the movie scenario, external knowledge of books is also abundant. Two public available datasets are commonly used, including Book-Crossing [96] and Amazon-Book [97]. Book-Crossing contains the user's demographic information and the book's attributes, such as the author, the publisher, the year of publication. The Amazon-Book dataset is the largest subset of the Amazon Review dataset. Compared with the Book-Crossing dataset, the user's review and user's behavior relations, such as "also view", "view and buy", "also buy", and "buy together" are available. Therefore, more relations can be included in the constructed KG. Moreover, the Amazon-Book dataset is much larger. Some papers also adopt the dataset of Douban-Book, DBbook2014, and IntentBooks.

• *Music.* Similar to the movie and book scenario, external knowledge of music tracks and artists can be obtained. There are two datasets extracted from the Last.fm online music system [98]. Last.FM-a [99] is a small dataset, which

provides the user's social relation, tags of tracks and artists, and listening records. While Last.FM-b [100] contains demographic information of users, tags of tracks, and user's listening records. Another popular dataset is the KKBox dataset, which was released by the WSDM Cup 2018 Challenge [101]. This dataset contains the listening records and the description of the track, including the genre, artist, composer, and lyricist. These datasets vary by the size, and the KKBox is the most sparse one.

• *News.* Compared with other scenarios, news recommendation is heavily dependent on the textual information, which requires incorporating natural language processing (NLP) techniques. Moreover, news recommendation is challenging [27] because the news itself is time-sensitive, and the content is highly condensed, which requires commonsense to understand. Besides, people are topic-sensitive in choosing news to read and may prefer news from various domains. Traditional news recommendation models fail to discover the high-level connection among the news. Therefore, KGs are introduced into this scenario [16], [27], [53] to find the logical relations between different news for more personalized recommendation. To build the KG for news recommendation, the first step is to recognize entities in the content, then map these extracted entities to the external knowledge base. The most popular dataset is Bing-News, which contains the user click information, news title, etc, however, this dataset is private. While MIND [102] is a recently released large-scale news dataset, containing a title, an abstract, a category label, and a body for each news article, as well as interaction records for each user. Moreover, entities in each news article are recognized and mapped into the Wikidata [103] knowledge base. The entities, their corresponding triplets in Wikidata, as well as entity and relation embeddings learned with the TransE [81] model are all included in the dataset, which makes it convenient for the research of KG-based news recommendation.

• *Product.* Compared with datasets in other scenarios, datasets in this domain are quite large and sparse. The most popular dataset is the Amazon Review dataset [97]. There are multiple subsets of different sizes in the Amazon Review dataset, including books, cell phones, clothing, music, electronics, etc, here we list the statistics of data in total. Except for the user and item attributes, user reviews and user behavior relations are also included. Although external knowledge of products is also available, most works [15], [52], [63], [68], [76] build the user-item KG directly with multiple types of relations within the recommendation dataset, while [67] build the item KG with the assistance of the Freebase to mine rules between associated items. Some papers [50], [76] also use the data from Alibaba Taobao.

• *POI.* Point of Interest (POI) recommendation is the recommendation of new businesses and activities (restaurants, museums, parks, cities, etc.) to users based on their historical check-in data. The most popular dataset is the Yelp Challenge [104], which contains the attributes of businesses and users, check-ins, and reviews. There are multiple versions of the Yelp Challenge dataset, here we present the dataset released in 2013. Similar to the product recommendation, the Yelp Challenge dataset contains rich POI side information and user-related relations, therefore, the user-item KG can be constructed with knowledge within the dataset. There is also paper [47] that utilizes the CEM dataset[1] to recommend next trip, and work [73] that uses the Dianping-Food dataset, which is provided by Dianping.com [105] for restaurant recommendation.

• *Social Platform.* This task is to recommend potentially interested people or meetings to users in the community. Since all the data are crawled by themselves, it is hard to map the user or item to external knowledge bases in some datasets. Therefore, most works build KGs with available relations in the crawled data. One application is to recommend unfollowed users to the target user on the social platform Weibo [106] with the collected Weibo tweets data [51], where an item KG is constructed to map celebrities to Satori. Another application is to recommend offline meetings for users on a social website, MeetUp [107]. The last application lies in the academic domain, to recommend conferences to researchers with the DBLP data [108].

5 Future Directions

In the above sections, we have demonstrated the advantage of KG-based recommender systems from the aspects of more accurate recommendation and explainability. Although many novel models have been proposed to utilize the KG as side information for recommendation, some further opportunities still exist. In this section, we outline and discuss some prospective research directions.

• *Dynamic Recommendation.* Although KG-based recommender systems with GNN or GCN architectures have achieved good performance, the training process is time-consuming. Thus such models can be regarded as static preference recommendation. However, in some scenarios, such as online shopping, news recommendation, Twitter, and forums, a user's interest can be influenced by social events or friends very quickly. In this case, recommendation with a static preference modeling may not be enough to understand real-time interests. In order to capture dynamic preference, leveraging the dynamic graph network can be a solution. Recently, Song *et al.* [109] designed a dynamic-graph-attention network to capture the user's rapidly-changing interests by incorporating long term and short term interests from friends. It is natural to integrate other types of side information and build a KG for dynamic recommendation by following such an approach.

• *Multi-Task Learning.* KG-based recommender systems can be naturally regarded as link prediction in the graph. Therefore, it is potential to improve the performance of graph-based recommendation by considering the nature of the KG. For example, there may exist missing facts in the KG, which leads to missing relations or entities. However, the user's preference may be ignored because these facts are missing, which can deteriorate the recommendation results. Papers [54], [71] have shown it is effective to jointly train the KG completion module and recommendation module for better recommendation. Other works have utilized multi-task learning by jointly training the recommendation module with the KGE task [53] and item relation regulation

1. an Amadeus database containing bookings over a dozen of airlines

task [55]. It would be interesting to exploit transferring knowledge from other KG-related tasks, such as entity classification and resolution, for better recommendation performance.

- *Cross-Domain Recommendation.* Recently, works on cross-domain recommendation have appeared. The motivation is that interaction data is unbalanced across domains. For example, on the Amazon platform, the book subset is larger than other domains. With the transfer learning technique, interaction data from the source domain with relatively rich data can be shared for better recommendation in the target domains. Zhang *et al.* [110] proposed a matrix-based method for cross-domain recommendation. Later, Zhao *et al.* [111] introduced PPGN, which puts users and products from different domains in one graph, and leverages the user-item interaction graph for cross-domain recommendation. Although PPGN outperforms SOTA significantly, the user-item graph contains only interaction relations, and does not consider other relationships among users and items. It could be promising to follow works in this survey, by incorporating different types of user and item side information in the graph to improve the performance of cross-domain recommendation.
- *Knowledge Enhanced Language Representation.* To improve the performance of various NLP tasks, there is a trend to integrate external knowledge into the language representation model, so that the knowledge representation and the text representation can be refined mutually. For instance, Chen *et al.* [112] proposed the STCKA for short text classification, which utilizes the prior knowledge from KGs to enrich the semantic representation of short texts. Zhang *et al.* [113] proposed the ERNIE, which incorporates knowledge from Wikidata to enhance the language representation, and such an approach has proven to be effective in the task of relation classification. Although the DKN model [27] utilizes both the text embedding and the entity embedding in the news, these two types of embeddings are simply concatenated to obtain the final representation of news, instead of considering the information fusion between two vectors. Therefore, it is promising to apply the strategy of knowledge-enhanced text representation in the news recommendation task and other text-based recommendation tasks for better representation learning, facilitating more accurate recommendation.

6 Conclusion

In this survey paper, we investigate KG-based recommender systems and summarize the recent efforts in this domain. This survey illustrates how different approaches utilize the KG as side information to improve the recommendation result as well as providing interpretability in the recommendation process. Moreover, an introduction to datasets used in different scenarios is provided. Finally, future research directions are identified, hoping to promote development in this field. KG-based recommender systems are promising for accurate recommendation and explainable recommendation, benefitting from the fruitful information contained in the KGs. We hope this survey paper can help readers better understand works in this area.

Acknowledgments

The research work was supported by the National Key Research and Development Program of China under Grant No. 2018YFB1004300, the National Natural Science Foundation of China under Grant No. U1836206, U1811461, 61773361, 61836013, 71531001, the Project of Youth Innovation Promotion Association CAS under Grant No. 2017146.

References

[1] 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 *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 1531–1540.

[2] C. Wang, H. Zhu, C. Zhu, C. Qin, and H. Xiong, "SetRank: A setwise Bayesian approach for collaborative ranking from implicit feedback," in *Proc. AAAI Conf. Artif. Intell.*, 2020, pp. 6127–6136.

[3] F. Zhuang, Z. Zhang, M. Qian, C. Shi, X. Xie, and Q. He, "Representation learning via dual-autoencoder for recommendation," *Neural Netw.*, vol. 90, pp. 83–89, 2017.

[4] J. Han *et al.*, "Adaptive deep modeling of users and items using side information for recommendation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 3, pp. 737–748, Mar. 2020.

[5] Y. Xu, Y. Yang, J. Han, E. Wang, F. Zhuang, and H. Xiong, "Exploiting the sentimental bias between ratings and reviews for enhancing recommendation," in *Proc. IEEE Int. Conf. Data Mining*, 2018, pp. 1356–1361.

[6] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.

[7] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances Artif. Intell.*, vol. 2009, 2009, Art. no. 4.

[8] Z. Sun *et al.*, "Research commentary on recommendations with side information: A survey and research directions," *Electron. Commerce Res. Appl.*, vol. 37, 2019, Art. no. 100879.

[9] S. Sen, J. Vig, and J. Riedl, "Tagommenders: Connecting users to items through tags," in *Proc. 18th Int. Conf. World Wide Web*, 2009, pp. 671–680.

[10] Y. Zhen, W.-J. Li, and D.-Y. Yeung, "TagiCoFi: Tag informed collaborative filtering," in *Proc. 3rd ACM Conf. Recommender Syst.*, 2009, pp. 69–76.

[11] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, 2017, pp. 425–434.

[12] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proc. ACM Conf. Recommender Syst.*, 2007, pp. 17–24.

[13] M. Jamali and M. Ester, "TrustWalker: A random walk model for combining trust-based and item-based recommendation," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 397–406.

[14] 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.

[15] Y. Zhang, Q. Ai, X. Chen, and P. Wang, "Learning over knowledge-base embeddings for recommendation," 2018, *arXiv: 1803.06540*.

[16] 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.

[17] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 1247–1250.

[18] J. Lehmann *et al.*, "DBpedia–A large-scale, multilingual knowledge base extracted from Wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.

[19] F. M. Suchanek, G. Kasneci, and G. Weikum, "YAGO: A core of semantic knowledge," in *Proc. 16th Int. Conf. World Wide Web*, 2007, pp. 697–706.

[20] A. Singhal, "Introducing the knowledge graph: Things, not strings," 2012. [Online]. Available: https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html

[21] C. Shi, Y. Li, J. Zhang, Y. Sun, and P. S. Yu, "A survey of heterogeneous information network analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 1, pp. 17–37, Jan. 2017.

[22] C. Liu, L. Li, X. Yao, and L. Tang, "A survey of recommendation algorithms based on knowledge graph embedding," in *Proc. IEEE Int. Conf. Comput. Sci. Educ. Informatization*, 2019, pp. 168–171.

[23] Y. Zhang and X. Chen, "Explainable recommendation: A survey and new perspectives," 2018, *arXiv: 1804.11192*.

[24] 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.

[25] D. Xi, F. Zhuang, Y. Liu, J. Gu, H. Xiong, and Q. He, "Modelling of bi-directional spatio-temporal dependence and users' dynamic preferences for missing POI check-in identification," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 5458–5465.

[26] P. Zhao *et al.*, "Where to go next: A spatio-temporal gated network for next POI recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 5877–5884.

[27] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: Deep knowledge-aware network for news recommendation," in *Proc. World Wide Web Conf.*, 2018, pp. 1835–1844.

[28] H. Liu, Y. Tong, J. Han, P. Zhang, X. Lu, and H. Xiong, "Incorporating multi-source urban data for personalized and context-aware multi-modal transportation recommendation," *IEEE Trans. Knowl. Data Eng.*, to be published, doi: 10.1109/TKDE.2020.2985954.

[29] Z. Huang *et al.*, "Exploring multi-objective exercise recommendations in online education systems," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 1261–1270.

[30] C. Qin *et al.*, "DuerQuiz: A personalized question recommender system for intelligent job interview," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 2165–2173.

[31] P. Lops, M. De Gemmis, and G. Semeraro, "Content-based recommender systems: State of the art and trends," in *Recommender Systems Handbook*. Berlin, Germany: Springer, 2011, pp. 73–105.

[32] R. Burke, "Hybrid recommender systems: Survey and experiments," *User Model. User-Adapted Interact.*, vol. 12, no. 4, pp. 331–370, 2002.

[33] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix: A survey of the state of the art and future challenges," *ACM Comput. Surv.*, vol. 47, no. 1, pp. 1–45, 2014.

[34] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, 2019.

[35] L. Ehrlinger and W. Wöß, "Towards a definition of knowledge graphs," in *Proc. 12th Int. Conf. Semantic Syst.: Posters Demos Track CEUR Workshop*, 2016.

[36] J. M. Gomez-Perez, J. Z. Pan, G. Vetere, and H. Wu, "Enterprise knowledge graph: An introduction," in *Exploiting Linked Data and Knowledge Graphs in Large Organisations*. Berlin, Germany: Springer, 2017, pp. 1–14.

[37] X. Huang, J. Zhang, D. Li, and P. Li, "Knowledge graph embedding based question answering," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, 2019, pp. 105–113.

[38] B. Xu *et al.*, "CN-DBpedia: A never-ending chinese knowledge extraction system," in *Proc. Int. Conf. Ind. Eng. Other Appl. Appl. Intell. Syst.*, 2017, pp. 428–438.

[39] R. Qian, "Understand your world with bing," *Bing Search Blog*, Mar. 2013.

[40] A. Hogan *et al.*, "Knowledge graphs," 2020, *arXiv: 2003.02320*.

[41] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic Web*, vol. 8, no. 3, pp. 489–508, 2017.

[42] M. Färber and A. Rettinger, "Which knowledge graph is best for me?" 2018, *arXiv: 1809.11099*.

[43] Y. Fang, W. Lin, V. W. Zheng, M. Wu, K. C.-C. Chang, and X.-L. Li, "Semantic proximity search on graphs with metagraph-based learning," in *Proc. IEEE 32nd Int. Conf. Data Eng.*, 2016, pp. 277–288.

[44] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, Sep. 2018.

[45] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition and applications," 2020, *arXiv: 2002.00388*.

[46] E. Palumbo, G. Rizzo, and R. Troncy, "Entity2rec: Learning user-item relatedness from knowledge graphs for top-N item recommendation," in *Proc. 11th ACM Conf. Recommender Syst.*, 2017, pp. 32–36.

[47] A. Dadoun, R. Troncy, O. Ratier, and R. Petitti, "Location embeddings for next trip recommendation," in *Companion Proc. World Wide Web Conf.*, 2019, pp. 896–903.

[48] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2018, pp. 505–514.

[49] D. Yang, Z. Guo, Z. Wang, J. Jiang, Y. Xiao, and W. Wang, "A knowledge-enhanced deep recommendation framework incorporating GAN-based models," in *Proc. IEEE Int. Conf. Data Mining*, 2018, pp. 1368–1373.

[50] Y. Ye *et al.*, "Bayes embedding (BEM): Refining representation by integrating knowledge graphs and behavior-specific networks," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 679–688.

[51] H. Wang, F. Zhang, M. Hou, X. Xie, M. Guo, and Q. Liu, "SHINE: Signed heterogeneous information network embedding for sentiment link prediction," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, 2018, pp. 592–600.

[52] 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.

[53] 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.

[54] 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.

[55] 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.

[56] X. Yu, X. Ren, Q. Gu, Y. Sun, and J. Han, "Collaborative filtering with entity similarity regularization in heterogeneous information networks," in *Proc. 1st Int. Joint Conf. Artif. Intell. Workshop Heterogeneous Inf. Netw. Anal.*, 2013.

[57] C. Luo, W. Pang, Z. Wang, and C. Lin, "Hete-CF: Social-based collaborative filtering recommendation using heterogeneous relations," in *Proc. IEEE Int. Conf. Data Mining*, 2014, pp. 917–922.

[58] X. Yu *et al.*, "Recommendation in heterogeneous information networks with implicit user feedback," in *Proc. 7th ACM Conf. Recommender Syst.*, 2013, pp. 347–350.

[59] X. Yu *et al.*, "Personalized entity recommendation: A heterogeneous information network approach," in *Proc. 7th ACM Int. Conf. Web Search Data Mining*, 2014, pp. 283–292.

[60] R. Catherine and W. Cohen, "Personalized recommendations using knowledge graphs: A probabilistic logic programming approach," in *Proc. 10th ACM Conf. Recommender Syst.*, 2016, pp. 325–332.

[61] C. Shi, Z. Zhang, P. Luo, P. S. Yu, Y. Yue, and B. Wu, "Semantic path based personalized recommendation on weighted heterogeneous information networks," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, 2015, pp. 453–462.

[62] C. Shi, B. Hu, W. X. Zhao, and P. S. Yu, "Heterogeneous information network embedding for recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 357–370, Feb. 2019.

[63] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, "Meta-graph based recommendation fusion over heterogeneous information networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 635–644.

[64] Z. Sun, J. Yang, J. Zhang, A. Bozzon, L.-K. Huang, and C. Xu, "Recurrent knowledge graph embedding for effective recommendation," in *Proc. 12th ACM Conf. Recommender Syst.*, 2018, pp. 297–305.

[65] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable reasoning over knowledge graphs for recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 5329–5336.

[66] X. Huang, Q. Fang, S. Qian, J. Sang, Y. Li, and C. Xu, "Explainable interaction-driven user modeling over knowledge graph for sequential recommendation," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 548–556.

[67] W. Ma *et al.*, "Jointly learning explainable rules for recommendation with knowledge graph," in *Proc. World Wide Web Conf.*, 2019, pp. 1210–1221.
[68] Y. Xian, Z. Fu, S. Muthukrishnan, G. de Melo, and Y. Zhang, "Reinforcement knowledge graph reasoning for explainable recommendation," 2019, *arXiv: 1906.05237*.
[69] W. Song, Z. Duan, Z. Yang, H. Zhu, M. Zhang, and J. Tang, "Explainable knowledge graph-based recommendation via deep reinforcement learning," 2019, *arXiv: 1906.09506*.
[70] X. Tang, T. Wang, H. Yang, and H. Song, "AKUPM: Attention-enhanced knowledge-aware user preference model for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 1891–1899.
[71] Q. Li, X. Tang, T. Wang, H. Yang, and H. Song, "Unifying task-oriented knowledge graph learning and recommendation," *IEEE Access*, vol. 7, pp. 115 816–115 828, 2019.
[72] 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.
[73] H. Wang *et al.*, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 968–977.
[74] 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.
[75] Y. Qu, T. Bai, W. Zhang, J. Nie, and J. Tang, "An end-to-end neighborhood-based interaction model for knowledge-enhanced recommendation," 2019, *arXiv: 1908.04032*.
[76] J. Zhao *et al.*, "IntentGC: A scalable graph convolution framework fusing heterogeneous information for recommendation," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 2347–2357.
[77] X. Sha, Z. Sun, and J. Zhang, "Attentive knowledge graph embedding for personalized recommendation," 2019, *arXiv: 1910.08288*.
[78] Y. Kim, "Convolutional neural networks for sentence classification," 2014, *arXiv:1408.5882*.
[79] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguistics 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 687–696.
[80] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," 2012, *arXiv:1205.2618*.
[81] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2787–2795.
[82] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 135–144.
[83] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*.
[84] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.
[85] 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.
[86] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. 28th AAAI Conf. Artif. Intell.*, 2014, pp. 1112–1119.
[87] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "PathSim: Meta path-based top-K similarity search in heterogeneous information networks," *Proc. VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.
[88] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," in *Proc. SIAM Int. Conf. Data Mining*, 2006, pp. 549–553.
[89] C. H. Q. Ding, T. Li, and M. I. Jordan, "Convex and semi-nonnegative matrix factorizations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 45–55, Jan. 2010.
[90] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.
[91] K. Cho *et al.*, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," 2014, *arXiv:1406.1078*.
[92] E. Palumbo, G. Rizzo, R. Troncy, E. Baralis, M. Osella, and E. Ferro, "An empirical comparison of knowledge graph embeddings for item recommendation," in *Proc. 1st Workshop Deep Learn. Knowl. Graphs Semantic Technol. 15th Extended Semantic Web Conf.*, 2018, pp. 14–20.
[93] GroupLens, "Movielens dataset," 1997. [Online]. Available: https://grouplens.org/datasets/movielens/
[94] MovieLens, "Movielens website," 1997. [Online]. Available: https://movielens.org/
[95] Douban, "Douban movie," 2005. [Online]. Available: http://movie.douban.com/
[96] C.-N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, "Improving recommendation lists through topic diversification," in *Proc. 14th Int. Conf. World Wide Web*, 2005, pp. 22–32.
[97] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2015, pp. 43–52.
[98] Last.fm, "Last.fm online music system," 2002. [Online]. Available: http://www.last.fm/
[99] GroupLens, "Last.fm-A dataset," 2011. [Online]. Available: https://grouplens.org/datasets/hetrec-2011/
[100] M. Schedl, "The LFM-1b dataset for music retrieval and recommendation," in *Proc. ACM Int. Conf. Multimedia Retrieval*, 2016, pp. 103–110.
[101] KKBox, "Kkbox dataset," 2018. [Online]. Available: https://wsdm-cup-2018.kkbox.events/
[102] F. Wu *et al.*, "MIND: A large-scale dataset for news recommendation," in *Proc. 58th Annu. Meeting Assoc. Computat. Linguistics*, 2020, pp. 3597–3606.
[103] D. Vrandečić and M. Krötzsch, "Wikidata: A free collaborative knowledgebase," *Commun. ACM*, vol. 57, no. 10, pp. 78–85, 2014.
[104] Yelp, "Yelp challenge dataset," 2013. [Online]. Available: https://www.kaggle.com/c/yelp-recsys-2013/
[105] Meituan-Dianping, "Dianping.com," 2009. [Online]. Available: https://www.dianping.com/
[106] Sina, "Sina weibo," 2009. [Online]. Available: http://weibo.com
[107] Meetup, "Meetup," 2002. [Online]. Available: http://www.meetup.com/
[108] DBLP, "DBLP dataset," 2013. [Online]. Available: https://dblp.uni-trier.de/xml/
[109] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, 2019, pp. 555–563.
[110] Q. Zhang, J. Lu, D. Wu, and G. Zhang, "A cross-domain recommender system with kernel-induced knowledge transfer for overlapping entities," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 7, pp. 1998–2012, Jul. 2019.
[111] C. Zhao, C. Li, and C. Fu, "Cross-domain recommendation via preference propagation GraphNet," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 2165–2168.
[112] J. Chen, Y. Hu, J. Liu, Y. Xiao, and H. Jiang, "Deep short text classification with knowledge powered attention," in *Proc. AAAI Conf. Artif. Intell.*, 2019, pp. 6252–6259.
[113] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "ERNIE: Enhanced language representation with informative entities," in *Proc. Annu. Meeting Assoc. Comput. Linguistics*, 2019, pp. 1441–1451.

Qingyu Guo received the BE degree from the University of Chinese Academy of Sciences (UCAS), Beijing, China, in 2018. He is currently working toward the PhD degree in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology (HKUST), Hong Kong. His current research interests include recommender systems, social media, and human-computer interaction.

Fuzhen Zhuang received the BE and PhD degrees in computer science from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2006 and 2011, respectively. He is currently an associate professor with the Institute of Computing Technology, Chinese Academy of Sciences. He has published more than 100 papers in some prestigious refereed journals and conference proceedings such as the *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Cybernetics*, *IEEE Transactions on Neural Networks and Learning Systems*, *ACM Transactions on Knowledge Discovery from Data*, *ACM Transactions on Intelligent Systems and Technology*, *Information Sciences*, *Neural Networks*, SIGKDD, IJCAI, AAAI, TheWebConf, SIGIR, ICDE, ACM CIKM, ACM WSDM, SIAM SDM, and IEEE ICDM. His research interests include transfer learning, machine learning, data mining, multitask learning, and recommendation systems. He is a senior member of CCF. He was a recipient of the Distinguished Dissertation Award of CAAI in 2013.

Chuan Qin received the BS degree in computer science and technology from the University of Science and Technology of China (USTC), Hefei, China, in 2015. He is currently working toward the PhD degree in the School of Computer Science and Technology, USTC, Hefei, China. He has authored more than 15 journal and conference papers in the fields of natural language processing and recommender system, including the *ACM Transactions on Information Systems*, KDD, SIGIR, WWW, AAAI, IJCAI, ICDM, etc.

Hengshu Zhu (Senior Member, IEEE) received the BE and PhD degrees in computer science from the University of Science and Technology of China (USTC), Hefei, China, in 2009 and 2014, respectively. He is currently a principal data scientist & architect with Baidu Inc. His general area of research is data mining and machine learning, with a focus on developing advanced data analysis techniques for emerging applied business research. He has published prolifically in refereed journals and conference proceedings, including the *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, the *IEEE Transactions on Mobile Computing (TMC)*, the *ACM Transactions on Information Systems (ACM TOIS)*, the *ACM Transactions on Knowledge Discovery from Data (TKDD)*, ACM SIGKDD, ACM SIGIR, WWW, IJCAI, and AAAI. He has served regularly on the organization and program committees of numerous conferences, including as a program co-chair of the KDD Cup-2019 Regular ML Track, and a founding co-chair of the first International Workshop on Organizational Behavior and Talent Analytics (OBTA-2018) and the International Workshop on Talent and Management Computing (TMC-2019), which were held in conjunction with ACM SIGKDD-2018 and ACM SIGKDD-2019 respectively. He was the recipient of the Distinguished Dissertation Award of CAS (2016), the Distinguished Dissertation Award of CAAI (2016), the Special Prize of President Scholarship for Postgraduate Students of CAS (2014), the Best Student Paper Award of KSEM-2011, WAIM-2013, CCDM-2014, and the Best Paper Nomination of ICDM-2014. He is a senior member of the ACM and CCF, and the committee member of the CCF Task Force on Big Data.

Xing Xie (Senior Member, IEEE) is currently a senior principal research manager with Microsoft Research Asia, working on data mining, social computing, and ubiquitous computing. During the past years, he has published more than 300 referred journal and conference papers, won the 10-year impact Award in ACM SIGSPATIAL 2019, the best student paper award in KDD 2016, and the Best Paper Awards in ICDM 2013 and UIC 2010. He has more than 50 patents filed or granted. He currently serves on the editorial boards of the *ACM Transactions on Social Computing*, *ACM Transactions on Intelligent Systems and Technology*, *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, *GeoInformatica*, *Pervasive and Mobile Computing*, and *CCF Transactions on Pervasive Computing and Interaction*. In recent years, he was involved in the program or organizing committees of more than 70 conferences and workshops. Especially, he served as program co-chair of ACM Ubicomp 2011, PCC 2012, UIC 2015, SMP 2017, and will be a program co-chair of ACM SIGSPATIAL 2020. He is a distinguished member of the ACM and China Computer Federation (CCF).

Hui Xiong (Fellow, IEEE) received the PhD degree from the University of Minnesota (UMN), Minneapolis, Minnesota. He is currently a full professor with the Rutgers, the State University of New Jersey, where he received the 2018 Ram Charan Management Practice Award as the Grand Prix winner from the Harvard Business Review, RBS dean's research professorship (2016), the Rutgers University Board of Trustees Research Fellowship for Scholarly Excellence (2009), the ICDM Best Research Paper Award (2011), and the IEEE ICDM Outstanding Service Award (2017). He is a co-editor-in-chief of the *Encyclopedia of GIS*, an associate editor of the *IEEE Transactions on Big Data (TBD)*, *ACM Transactions on Knowledge Discovery from Data (TKDD)*, and *ACM Transactions on Management Information Systems (TMIS)*. He has served regularly on the organization and program committees of numerous conferences, including as a program co-chair of the Industrial and Government Track for the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), a program co-chair for the IEEE 2013 International Conference on Data Mining (ICDM), a general co-chair for the IEEE 2015 International Conference on Data Mining (ICDM), and a program co-chair of the Research Track for the 2018 ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. He is an ACM distinguished scientist.

Qing He received the BS degree in mathematics from Hebei Normal University, Shijiazhang, China, in 1985, the MS degree in mathematics from Zhengzhou University, Zhengzhou, China, in 1987, and the PhD degree in fuzzy mathematics and artificial intelligence from Beijing Normal University, Beijing, China, in 2000. He is a professor with the Institute of Computing Technology, Chinese Academy of Science (CAS), and he is a professor with the Graduate University of Chinese (GUCAS). Since 1987 to 1997, he has been with the Hebei University of Science and Technology. He is currently a doctoral tutor with the Institute of Computing and Technology, CAS. His interests include data mining, machine learning, classification, fuzzy clustering. add here.