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. In recent years, generating recommendations with the knowledge graph as side information has attracted considerable interest. Such an approach can not only alleviate the abovementioned 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 summarize them from two perspectives. On the one hand, we investigate the proposed algorithms by focusing on how the papers utilize the knowledge graph for accurate and explainable recommendation. On the other hand, we introduce datasets used in these works. 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 for scenarios such as music recommendation [1], movie recommendation [2], and online shopping [3].

The recommendation algorithm is the core element of recommender systems, which are mainly categorized into collaborative filtering (CF)-based recommender systems, content-based recommender systems, and hybrid recommender systems [4]. 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 [5], [6]. However, CF-based recommendation suffers from the data sparsity and cold start problems [6]. 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 [7], [8], item reviews [9], [10], and users' social networks [11], [12].

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 function as entities, and edges represent relations between entities. Items and their attributes can be mapped into the KG to understand the mutual relations between items [2]. 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 [13]. Figure 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 KG-based recommender system is the explainability of recommendation results [14]. In the same example, reasons for recommending these two movies 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 [15], DBpedia [16], YAGO [17], and Google's Knowledge Graph [18], which makes it convenient to build KGs for recommendation.

- *Qingyu Guo, Fuzhen Zhuang, Qing He are with Key Lab of Intelligent Information Processing of Chinese Academy of Sciences (CAS), Institute of Computing Technology, CAS, Beijing 100190, China. Qingyu Guo is also with Hong Kong University of Science and Technology, Clearwater Bay, Kowloon, Hong Kong. Fuzhen Zhuang and Qing He are also with University of Chinese Academy of Sciences, Beijing 100049, China. E-mail: qyguo1996@gmail.com, {zhuangfuzhen, heqing}@ict.ac.cn*
- *Chuan Qin is with University of Science and Technology of China and Baidu Talent Intelligence Center, Baidu Inc. E-mail: chuanqin0426@gmail.com*
- *Hengshu Zhu and Hui Xiong is with Baidu Talent Intelligence Center, Baidu Inc. Hui Xiong is also with Business Intelligence Lab, Baidu Research. E-mail: {zhuhengshu, xionghui}@gmail.com*
- *Xing Xie is with Microsoft Research Asia, Beijing, China. E-mail: xingx@microsoft.com*
- *Fuzhen Zhuang is the corresponding author.*

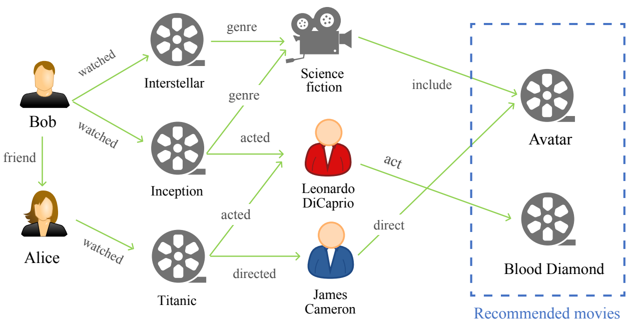

Fig. 1. An illustration of KG-based recommendation.

This survey aims to provide a comprehensive review of the literature utilizing KGs as side information in recommender systems. Throughout our investigation, we discover that existing KG-based recommender systems apply KGs in three ways: the embedding-based method, the path-based method, and the unified method. We illustrate the similarities and differences between these methods in detail. Besides the more accurate recommendation, another benefit of KG-based recommendation is the interpretability. We discuss how different works utilize the KG for explainable recommendation. In addition, based on our survey, 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.

The organization of this survey is as follows: in Section 2, we introduce the foundations of KGs and recommender systems; in Section 3, we present notations and concepts used in this paper; in Section 4 and Section 5, we review KG-based recommender systems from the aspect of approaches and evaluated datasets, respectively; in Section 6, we provide some potential research directions in this field; finally, we conclude this survey in Section 7.

2 Related Work

This section introduces the fundamental knowledge and summarizes related work in the domain of KG-based recommendation, including KGs and recommender systems.

2.1 Knowledge Graphs

The KG is a practical approach to represent large-scale information from multiple domains [19]. A common way to describe a KG is to follow the Resource Description Framework (RDF) standard [20], in which nodes represent entities, while edges in the graph function as relations between entities. Each edge is represented in the form of a triple (head entity, relation, tail entity), also known as a fact in the graph, implying 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 since 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. The concept of knowledge graph was developed in the 1980s [21], when KGs were integrated into the framework of expert systems for medical and social sciences. Later, the application was broadened to linguistic and logical domains. In 2012, Google introduced the KG into the framework of search, for a better understanding of the query and to make search results more user-friendly [18]. To date, KGs have been created and applied in multiple scenarios, including search engines, recommender systems, Question Answering system [22], relation detection [23], etc.

We list some popular KGs in Table 1. Based on the scope of the knowledge covered, these KGs can be divided into two classes. The first group are cross-domain KGs, such as Freebase [15], DBpedia [16], YAGO [17], and NELL [24], while the second are domain-specific KGs, like Bio2RDF [25]. Six of the cross-domain KGs are utilized in recommender systems in this survey, and we briefly introduce them as follows: Freebase [15] was launched in 2007 by Metaweb and was acquired by Google in 2010. It contains more than 3 billion facts and almost 50 million entities by 2015 [26]. Though it is a cross-domain KG, around 77% of its information is in the domain of media [27]. Currently, the data is available at Google's Data Dumps [28]. DBpedia [16] is an open community project, which was started by researchers from the Free University of Berlin and Leipzig University, in cooperation with OpenLink Software. The first version was released in 2007 and is updated yearly. The main knowledge is extracted from different language versions of Wikipedia, and DBpedia combines them in a large-scale graph structure. YAGO [17] (Yet Another Great Ontology) was introduced by the Max Planck Institute in 2007. It contains more than 5 million facts, such as people, locations, and organizations. It automatically extracts and unifies knowledge from Wikipedia and multiple sources, including WordNet [29] and GeoNames [30], then unifies them into an RDF graph. Satori [31] is a KG proposed by Microsoft. Similarly to Google's Knowledge Graph, which empowers the Google search engine, Satori has been integrated into the search engine Bing. Though publicly accessible documents about the Satori KG are limited, it is known that Satori consisted of 300 million entities and 800 million relations in 2012 [32]. CN-DBPedia [33] is the largest Chinese KG. Published by Fudan University in 2015, it has over 16 million entities and over 220 million relations. It automatically extracts knowledge from Baidu Baike, Hudong Baike, and Chinese Wikipedia, then integrates them into a Chinese database. The system is updated continuously with little human effort needed.

2.2 Recommender Systems

Recommender systems have been applied in many domains, including movies [2], [44], music [1], [45], POIs [46], [47], news [14], [48], education [49], [50], etc. 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 a representation $\mathbf{u}_i$ and $\mathbf{v}_j$ for the given user u_i and an 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

TABLE 1
A collection of commonly used knowledge graphs.

KG Name	Domain Type	Main Knowledge Source
YAGO [17]	Cross-Domain	Wikipedia [34]
Freebase [15]	Cross-Domain	Wikipedia, NNDB [35]FMD [36], MusicBrainz [37]
DBpedia [16]	Cross-Domain	Wikipedia
Satori [31]	Cross-Domain	Web Data
CN-DBPedia [33]	Cross-Domain	Baidu Baike [38], Hudong Baike [39], Wikipedia (Chinese)
NELL [24]	Cross-Domain	Web Data
Wikidata [40]	Cross-Domain	Wikipedia, Freebase
Google's Knowledge Graph [18]	Cross-Domain	Web data
Facebooks Entities Graph [41]	Cross-Domain	Wikipedia, Facebook data [42]
Bio2RDF [25]	Biological Domain	Public bioinformatics databases, NCBIs databases
KnowLife [43]	Biomedical Domain	Scientific literature, Web portals

recommendation can be generated by sorting the preference scores for items. To learn the user/item representation and the scoring function, there are three main approaches, as described below.

• **Collaborative Filtering.** CF assumes that users may be interested in items selected by people who share similar interaction records with them. The interaction can either be explicit interaction [51], [52], like ratings, or implicit interaction [53], [54], such as click and view. To implement CF-based recommendation, interaction data from multiple users and items are required, which further forms the user-item interaction matrix. The CF-based approach contains two main techniques, memory-based CF and model-based CF [5]. In detail, memory-based CF first learns the user-user similarity from the user-item interaction data. Then, unobserved items are recommended to a given user based on the interaction records of people similar to the specific user. Alternatively, some models learn the similarity among items, and recommend similar items for a user based on the user's purchase history. The model-based CF approach attempts to alleviate the sparsity issue by building an inference model. One common implementation is the latent factor model [55], [56], which extracts the latent representation of the user and item from the high dimensional user-item interaction matrix, and then computes the similarity between the user and item with the inner product or other methods.

• **Content-based Filtering.** Compared with the CF-based model, which learns the representation of user and item from global user-item interaction data, content-based methods depict the user and item from the content of items. The assumption of content-based filtering is that users may be interested in items that are similar to their past interacted items. The item representation is obtained by extracting attributes from the item's auxiliary information, including texts, images, etc., while the user representation is based on the features of personal interacted items. The procedure of comparing candidate items with the user profile is essentially matching them with the user's previous records. Therefore, this approach tends to recommend items that are similar to items liked by a user in the past [57].

• **Hybrid Method.** Hybrid method is to leverage multiple recommendation techniques in order to overcome the limitation of using only one type of method. One major issue of CF-based recommendation is the sparsity of user-item interaction data, which makes it difficult to find similar items or users from the perspective of interaction. A special case for this issue is the cold-start problem, which means the recommendation for new user or item is difficult, since the user-user and item-item similarity cannot be determined without any interaction records. By incorporating content information of users and items, also known as user side information and item side information, into the CF-based framework, better recommendation performance can be achieved [6]. Some commonly used item side information include item attributes [7], [8], [58], [58], like brands, categories; item multimedia information, like textual description [59], image features [60], audio signals [61], video features [62]; and item reviews [9], [10]. Common options for user side information involve user's demographic information [63], including occupation, gender, and hobbies; and user network [11], [12]. In this survey, KG-based recommender systems leverage the KG as the side information, combining the CF-based technique for more accurate recommendation.

3 Overview

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 2.

• **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 $|\mathcal{A}| > 1$ and/or the number of relation types $|\mathcal{R}| > 1$.

• **Knowledge Graph (KG).** A KG $\mathcal{G}_{know} = (V, E)$ 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. It can be regarded as an instance of a HIN.

• **Meta-path.** A meta-path $\mathcal{P} = A_0 \xrightarrow{R_1} A_1 \xrightarrow{R_2} \cdots \xrightarrow{R_k} A_k$

TABLE 2
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
$\mathcal{U} = \{u_1, u_2, \cdots, u_m\}$	User set
$\mathcal{V} = \{v_1, v_2, \cdots, v_n\}$	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
$\mathcal{P}(e_i, e_j) = \{p_1, p_2, \cdots, p_s\}$	Path set between entity pair (e_i, e_j)
Φ	Nonlinear Transformation
$\odot$	Element-wise Product
$\oplus$	Vector concatenation operation

is a path defined on the graph of network schema $G_T = (\mathcal{A}, \mathcal{R})$, which defines a new composite relation $R_1 R_2 \cdots R_k$ between type A_0 and A_k, where $A_i \in \mathcal{A}$ and $R_i \in \mathcal{R}$ for $i = 0, \cdots, k$. 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 [64]. 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 $\mathcal{G}_{know} = (V, E)$ into a low dimensional space [65]. 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.

• **User Feedback.** With m users $\mathcal{U} = \{u_1, \cdots, u_m\}$ and n items $\mathcal{V} = \{v_1, \cdots, v_n\}$, we define the binary user feedback matrix $R \in \mathbb{R}^{m\times n}$ as follows:

$$R_{ij} = \begin{cases} 1, & \text{if } (u_i, v_j) \text{ interaction is observed;} \\ 0, & \text{otherwise.} \end{cases}$$

Note that a value of 1 for R_{ij} indicates there is an implicit interaction between user u_i and item v_j, such as behaviors of clicking, watching, browsing, etc. Such an implicit interaction does not necessarily imply u_i's preference over v_j. Unless otherwise stated, the user feedback used in this paper means the implicit feedback. However, in some specific scenarios, explicit feedback to show the user's preference can also be available. For example, in movie recommendation, a user explicitly rates a movie in the score range of one to five. Some papers have extracted the data of score ratings of five to indicate the user's preference in such a case.

• **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 $\mathcal{N}_{e_0}^0$ is e_0 itself.

• **Relevant Entity.** Given the interaction matrix R and the knowledge graph $\mathcal{G}_{know}$, the set of k-hop relevant entities for user u can be represented as

$$\mathcal{E}_u^k = \left\{ e_t | (e_h, r, e_t) \in \mathcal{G} \text{ and } e_h \in \mathcal{E}_u^{k-1} \right\}, \\ k = 1, 2, \cdots, H.$$

where $\mathcal{E}_u^0 = \{u | R_{uv} = 1\}$ is the set of the user's historical interacted items.

• **User Ripple Set.** The ripple set of a user is defined as the knowledge triples with the head entities being $(k-1)$-hop relevant entities $\mathcal{E}_u^{k-1}$,

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

• **Entity Ripple Set.** The ripple set of an entity $e \in \mathcal{G}$ is defined as

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

4 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, path-based methods, and unified methods. We will introduce how different methods leverage KGs to improve the recommendation results. To facilitate readers checking the literature, we summarize and organize these papers in Table 3, which lists their publication information, the approach to utilize a KG for recommendation, and the techniques adopted in these works.

Explainable recommendation has been another hot research topic in recent years. It is helpful for users to adopt the suggestions generated by the recommender system if appropriate explanations are provided to them [97]. Compared with traditional recommender systems, KG-based recommendation makes the reasoning process available. In this section, we will also show how different works leverage KGs for explainable recommendation.

4.1 Embedding-based Methods

The embedding-based methods generally use the information from the KG directly to enrich the representation of items or users. In order to exploit the KG information, knowledge graph embedding (KGE) algorithms need to be applied to encode the KG into low-rank embedding. KGE algorithms can be divided into two classes [98]: translation distance models, such as TransE [99], TransH [100], TransR [101], TransD [102], etc., and semantic matching models, such as DistMult [103].

Based on whether users are included in the KG, embedding-based methods can be divided into two classes. In the first type of method, KGs are constructed with items and their related attributes, which are extracted from the dataset or external knowledge bases. We name such a graph as the item graph. Note that users are not included in such an item graph. Papers following this strategy leverage the knowledge graph embedding (KGE) algorithms to encode the graph for a more comprehensive representation of items, and then integrate the item side information into the recommendation framework. The general idea can be illustrated as follows. The latent vector $\mathbf{v}_j$ of each item v_j is obtained by aggregating information from multiple sources, such as the KG, the user-item interaction matrix, item's content, and item's attributes. The latent vector $\mathbf{u}_i$ of each user u_i can either be extracted from the user-item interaction matrix, or the combination of interacted items' embedding. Then, the probability of u_i selecting v_j can be calculated with

$$\hat{y}_{i,j} = f(\mathbf{u}_i, \mathbf{v}_j), \tag{1}$$

where $f(\cdot)$ refers to a function to map the embedding of the user and item into a preference score, which can be the inner product, DNN, etc. In the recommendation stage, results will be generated in descending order of the preference score $\hat{y}_{i,j}$.

For instance, Zhang *et al.* [2] proposed CKE, which unifies various types of side information in the CF framework. They fed the item's structural knowledge (item's attributes represented with knowledge graph) and content (textual and visual) knowledge into a knowledge base embedding module. The latent vector of the item's structural knowledge $\mathbf{x}_j$ is encoded with the TransR algorithm, while the textual feature $\mathbf{z}_{t,j}$ and the visual feature $\mathbf{z}_{v,j}$ are extracted with the autoencoder architecture. Then these representations are aggregated along with the offset vector $\boldsymbol{\eta}_j$ extracted from the user-item interaction matrix. The final representation of each item v_j can be written as

$$\mathbf{v}_j = \boldsymbol{\eta}_j + \mathbf{x}_j + \mathbf{z}_{t,j} + \mathbf{z}_{v,j}. \tag{2}$$

After obtaining the latent vector $\mathbf{u}_i$ of the user u_i, the preference score $\hat{y}_{i,j}$ is obtained via the inner product $\mathbf{u}_i^T\mathbf{v}_j$. Finally, in the prediction stage, items are recommended to u_i by the following ranking criteria:

$$v_{j1} > v_{j2} > \cdots > v_{jn} \rightarrow \mathbf{u}_i^T\mathbf{v}_{j1} > \mathbf{u}_i^T\mathbf{v}_{j2} > \cdots > \mathbf{u}_i^T\mathbf{v}_{jn}. \tag{3}$$

Experiments show that incorporating structural knowledge can boost the performance of recommendation.

Wang *et al.* [48] proposed DKN for news recommendation. It models the news by combining the textual embedding of sentences learned with Kim CNN [104] and the knowledge-level embedding of entities in news content via TransD. With the incorporation of a KG for entities, high-level semantic relations of news can be depicted in the final embedding $\mathbf{v}_j$ of news 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 $\{\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_{N_i}\}$ with an attention mechanism. The attention weight for each news $v_k (k = 1, 2, \cdots, N_i)$ in the clicked news set is calculated via

$$s_{v_k,v_j} = \frac{\exp\left(g\left(\mathbf{v}_k, \mathbf{v}_j\right)\right)}{\sum_{k=1}^{N_i} \exp\left(g\left(\mathbf{v}_k, \mathbf{v}_j\right)\right)}, \tag{4}$$

where $g(\cdot)$ is a DNN layer, v_j is the candidate news. Then, the final user embedding $\mathbf{u}_i$ is calculated via the weighted sum of clicked news embeddings:

$$\mathbf{u}_i = \sum_{k=1}^{N_i} s_{v_k,v_j}\mathbf{v}_k. \tag{5}$$

Finally, user's preference for candidate news v_j can be calculated with Equation 1, where $f(\cdot)$ is a DNN layer. Huang *et al.* [44] proposed the KSR framework for sequential recommendation. KSR uses a GRU network with a knowledge-enhanced key-value memory network (KV-MN) to model comprehensive user preference from the sequential interaction. The GRU network captures the user's sequential preference, while the KV-MN module utilizes knowledge base information (learned with TransE) to model the user's attribute-level preference. In this way, fine-grained user preference can be captured for recommendation. In detail, at time t, the latent vector of u_i is represented as $\mathbf{u}_i^t = \mathbf{h}_i^t \oplus \mathbf{m}_i^t$, where $\mathbf{h}_i^t$ and $\mathbf{m}_i^t$ stands for the representation of user's interaction-level preference and attribute-level preference, respectively. The latent vector of v_j is represented as $\mathbf{v}_j = \mathbf{q}_j \oplus \mathbf{e}_j \cdot \mathbf{u}_i^t$, where $\mathbf{q}_j$ is the item embedding in the GRU network, and $\mathbf{e}_j$ is the item embedding in the KG. After transforming $\mathbf{u}_i^t$ and $\mathbf{v}_j$ to the same dimension, the user's preference for items is ranked with the score obtained from Equation 1, where $f(\cdot)$ is the inner product.

The other type of embedding-based method directly builds a user-item graph, where users, items, and their related attributes function as nodes. In the user-item graph, both attribute-level relation (brand, category, etc) and user-related relations (co-buy, co-view, etc.) serve as edges. After obtaining the embeddings of entities in the graph, the user's preference can be calculated with Equation 1, or by further

TABLE 3
Table of collected papers. In the table, ‘Emb.’ stands for embedding-based Method, ‘Uni.’ stands for unified method, ‘Att.’ stands for attention mechanism, ‘RL’ stands for reinforcement learning, ‘AE’ stands for autoencoder, and ‘MF’ stands for matrix factorization.

Method	Venue	Year	KG Usage Type			Framework							
			Emb.	**Path**	**Uni.**	CNN	RNN	Att.	GNN	GAN	RL	AE	MF
CKE [2]	KDD	2016	✓									✓	
entity2rec [66]	RecSys	2017	✓										
ECFKG [67]	Algorithms	2018	✓										
SHINE [68]	WSDM	2018	✓									✓	
DKN [48]	WWW	2018	✓			✓		✓					
KSR [44]	SIGIR	2018	✓				✓	✓					
CFKG [13]	SIGIR	2018	✓										
KTGAN [69]	ICDM	2018	✓							✓			
KTUP [70]	WWW	2019	✓										
MKR [45]	WWW	2019	✓					✓					
DKFM [71]	WWW	2019	✓										
SED [72]	WWW	2019	✓										
RCF [73]	SIGIR	2019	✓					✓					
BEM [74]	CIKM	2019	✓										
Hete-MF [75]	IJCAI	2013		✓									✓
HeteRec [76]	RecSys	2013		✓									✓
HeteRec_p [77]	WSDM	2014		✓									✓
Hete-CF [78]	ICDM	2014		✓									✓
SemRec [79]	CIKM	2015		✓									✓
ProPPR [80]	RecSys	2016		✓									✓
FMG [3]	KDD	2017		✓									✓
MCRec [1]	KDD	2018		✓		✓		✓					✓
RKGE [81]	RecSys	2018		✓			✓	✓					
HERec [82]	TKDE	2019		✓									✓
KPRN [83]	AAAI	2019		✓			✓	✓					
RuleRec [84]	WWW	2019		✓									✓
PGPR [85]	SIGIR	2019		✓							✓		
EIUM [86]	MM	2019		✓		✓		✓					
Ekar [87]	arXiv	2019		✓							✓		
RippleNet [14]	CIKM	2018			✓			✓					
RippleNet-agg [88]	TOIS	2019			✓			✓	✓				
KGCN [89]	WWW	2019			✓			✓	✓				
KGAT [90]	KDD	2019			✓			✓	✓				
KGCN-LS [91]	KDD	2019			✓			✓	✓				
AKUPM [92]	KDD	2019			✓			✓					
KNI [93]	KDD	2019			✓			✓	✓				
IntentGC [94]	KDD	2019			✓				✓				
RCoLM [95]	IEEE Access	2019			✓			✓					
AKGE [96]	arXiv	2019			✓			✓	✓				

considering the relation embedding in the graph via

$$\hat{y}_{i,j} = f(\mathbf{u}_i, \mathbf{v}_j, \mathbf{r}), \tag{6}$$

where $f(\cdot)$ maps the user representation $\mathbf{u}_i$, the item representation $\mathbf{v}_j$, as well as the relation embedding $\mathbf{r}$ into a scalar.

Zhang *et al.* [13] proposed CFKG, which constructs a user-item KG. In this user-item graph, user behaviors (purchase, mention) are regarded as one relation type between entities, and multiple types of item side information (review, brand, category, bought-together, etc.) are included. To learn the embedding of entities and relations in the graph, the model defines a metric function $d(\cdot)$ to measure the distance between two entities according to a given relation. In the recommendation phase, the system will rank candidate items j in an ascending order of the distance between u_i and v_j

$$d\left(\mathbf{u}_i + \mathbf{r}_{buy}, \mathbf{v}_j\right), \tag{7}$$

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 $\hat{y}_{i,j}$.

Wang *et al.* [68] proposed SHINE, which takes the celebrity recommendation task as the sentiment link prediction task between entities in the graph. In detail, SHINE builds a sentiment network G_s for users and targets (celebrities), and utilizes their social network G_r and profile information network G_p as side information. These three networks are embedded with the auto-encoder technique, and are then aggregated as the representation of the user and target. Finally, the recommendation can be generated by following Equation 1, where $f(\cdot)$ is a DNN layer. Dadoun *et al.* [71] proposed DKFM for POI recommendation. DKFM applies TransE over a city KG to enrich the representation of the destination, which shows improvement in the performance of POI recommendation.

Previous works generally directly utilize the raw latent vector of structural knowledge learned with the KGE technique for recommendation. Recently, some papers have tried to improve the recommendation performance by refining the learned entity/relation representation. For instance, Yang *et al.* [69] introduced a GAN-based model, KTGAN, for movie recommendation. In the first phase, KTGAN learns the knowledge embedding $\mathbf{v}_j^k$ for movie v_j by incorporating the Metapath2Vec model [105] on the movie's KG, and the tag embedding $\mathbf{v}_j^t$ with the Word2Vec model [106] 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. Then, a generator G and a discriminator D are proposed to refine initial representations of users and items. The generator G tries to generate relevant (favorite) 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,

$$\begin{aligned}\mathcal{L} = \min_\theta \max_\phi \sum_{i=1}^{M} \{ & \mathbb{E}_{v_j \sim p_{\text{true}}(v_j|u_i,r)} \left[\log P\left(v_j|u_i\right)\right] \\ & + \mathbb{E}_{v_j \sim p_\theta(v_j|u_i,r)} \left[\log\left(1 - P\left(v_j|u_i\right)\right)\right] \},\end{aligned} \tag{8}$$

where $P(v_j|u_i) = \frac{1}{1+\exp(-f_\phi(u_i,v_j))}$ 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 and movies can be ranked with G's score function $p_\theta(v_j|u_i, r)$. Later, Ye *et al.* [74] proposed BEM, which uses two types of graphs for items, the knowledge-related graph (containing item attributes information, like brand, category, etc.) and behavior graph (containing item interaction-related information, including co-buy, co-rate, co-add to cart) for recommendation. BEM first learns the initial embeddings from the knowledge-related graph and the behavior graph with the TransE model and a GNN-based model, respectively. Then, BEM applies a Bayesian framework to refine these two types of embeddings mutually. Recommendation can be generated by finding the closest items of the interacted items in the behavior graph, which are measured by the relation of 'co-buy' or 'co-click'.

Another trend is to adopt the strategy of multi-task learning, to jointly learn the recommendation task with the guidance of the KG-related task. Generally, in the recommendation task, a function $f(\mathbf{u}_i, \mathbf{v}_j)$ is learned from the user-item interaction matrix to contrast the observed interaction pair (u_i, v_j) and unobserved interaction pair $(u_i, v_j\prime)$; in the KG-related task, another function $g(\mathbf{e}_h, \mathbf{r}, \mathbf{e}_t)$ is learned to determine whether (e_h, r, e_t) is a valid triplet in the KG. These two parts are connected with the following objective function,

$$\mathcal{L} = \mathcal{L}_{rec} + \lambda \mathcal{L}_{KG}, \tag{9}$$

where $\mathcal{L}_{rec}$ is the loss function for recommendation, $\mathcal{L}_{KG}$ is the loss function for the KG related task, and λ is the hyper-parameter to balance the two tasks. A general motivation for the multi-task learning is that item embeddings in the recommendation module share features with the associated entity embeddings in the KG.

Cao *et al.* [70] proposed KTUP to jointly learn the task of recommendation and knowledge graph completion. In the recommendation module, the loss function is defined as

$$\mathcal{L}_{rec} = \sum_{(u,v,v')\in R} -\log \sigma \left[f\left(\mathbf{u}, \mathbf{v}', \mathbf{p}'\right) - f\left(\mathbf{u}, \mathbf{v}, \mathbf{p}\right) \right], \tag{10}$$

where (u, v) is the observed user-item pair in the user-item interaction matrix ($R_{uv} = 1$); $(u, v\prime)$ denotes the unobserved user-item pair ($R_{uv\prime} = 0$); $\mathbf{p}$ denotes the latent vector of user's preference for the given item; $f(\cdot)$ is the proposed translation-based model, TUP, to model the correctness of such a user-item pair; and σ is the sigmoid function. For the KG completion module, a hinge loss is adopted,

$$\mathcal{L}_{KG} = \sum_{(e_h,r,e_t)\in\mathcal{G}} \sum_{(e'_h,r',e'_t)\in\mathcal{G}^-} \left[g\left(\mathbf{e}_h, \mathbf{r}, \mathbf{e}_t\right) + \gamma - g\left(\mathbf{e}'_h, \mathbf{r}', \mathbf{e}'_t\right) \right]_+, \tag{11}$$

where $\mathcal{G}^-$ is constructed by replacing e_h or e_t in the valid triplet $(e_h, r, e_t) \in \mathcal{G}$; $g(\cdot)$ is the TransH model, and a lower $g(\mathbf{e}_h, \mathbf{r}, \mathbf{e}_t)$ value infers a higher correctness of such a triplet; $[\cdot]_+ \triangleq \max(0, \cdot)$; and γ is the margin between correct triplets and incorrect triplets. The recommendation module is to mine the preference relation between user u and item v, while the knowledge completion task is to mine the relation among items in the KG. The bridge between these two modules is that items can be aligned with corresponding entities in the KG, and the user's preference is related with relations among entities in the KG. Hence, embeddings of items and preferences can be enriched by transferring knowledge of entities, relations and preference in each module under the framework of KTUP. Meanwhile, Wang *et al.* [45] proposed MKR, which consists of a recommendation module and a KGE module. The former learns latent representation for users and items, while the latter learns representation for item associated entities with the semantic matching KGE model. These two parts are connected with a cross & compress unit to transfer knowledge and share regularization of items in the recommendation module and entities in the KG. Xin *et al.* [73] proposed RCF, which introduces a hierarchical description of items, including both the relation type embedding and relation value embedding. RCF utilizes the DistMult model for KGE to preserve the relational structure between items. Then, it models the user's type-level preference and value-level preference separately with the attention mechanism. With the jointly training of recommendation module and the KG relation modeling module, decent recommendations can be made.

Summary for Embedding-based Methods. Most embedding-based methods [2], [44], [45], [48], [69], [70], [72], [73], [74] build KGs with multiple types of item side information to enrich the representation of items, and such information can be used to model the user representation more precisely. Some models [13], [66], [67], [68], [71] build user-item graphs by introducing users into the graph, which can directly model the user preference. Entity embedding is the core of embedding-based methods, and some papers refine the embedding with GAN [69] or BEM [74] for better recommendation. Embedding-based methods leverage the information in the graph structure intrinsically. Papers [45], [70], [73] apply the strategy of multi-task learning to jointly train the recommendation module along with the graph-related task to improve the quality of recommendation.

4.2 Path-based Methods

Path-based methods build a user-item graph and leverage the connectivity patterns of the entity in the graph for recommendation. Path-based methods have been developed since 2013, and traditional papers call this type of method as recommendation in the HIN. In general, these models take advantage of the connectivity similarity of users and/or items to enhance the recommendation. To measure the connectivity similarity between entities in the graph, *PathSim* [107] is commonly used. It is defined as

$$s_{x,y} = \frac{2 \times |\{p_{x \rightsquigarrow y} : p_{x \rightsquigarrow y} \in \mathcal{P}\}|}{|\{p_{x \rightsquigarrow x} : p_{x \rightsquigarrow x} \in \mathcal{P}\}| + |\{p_{y \rightsquigarrow y} : p_{y \rightsquigarrow y} \in \mathcal{P}\}|}, \quad (12)$$

where $p_{m \rightsquigarrow n}$ is a path between the entity m and n.

One type of path-based method leverages semantic similarities of entities in different meta-paths as the graph regularization to refine the representation of users and items in the HIN. Then, u_i's preference for v_j can be predicted by following Equation 1, where $f(\cdot)$ refers to the inner product. Three types of entity similarities are commonly utilized,

- **User-User Similarity:** the objective function for this term is

$$\min_{\mathbf{U},\Theta} \sum_{l=1}^{L} \theta_l \sum_{i=1}^{m} \sum_{j=1}^{m} s_{i,j}^{l} \left\| \mathbf{u}_i - \mathbf{u}_j \right\|_F^2 . \quad (13)$$

where $\| \cdot \|_F$ denotes the matrix Frobenius norm, $\Theta = [\theta_1, \theta_2, \cdots, \theta_L]$ denotes the weight for each meta-path, $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_m]$ 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 for this term is

$$\min_{\mathbf{V},\Theta} \sum_{l=1}^{L} \theta_l \sum_{i=1}^{n} \sum_{j=1}^{n} s_{i,j}^{l} \left\| \mathbf{v}_i - \mathbf{v}_j \right\|_F^2 . \quad (14)$$

where $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n]$ 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 for this term is

$$\min_{\mathbf{U},\mathbf{V},\Theta} \sum_{l=1}^{L} \theta_l \sum_{i=1}^{m} \sum_{j=1}^{n} (\mathbf{u}_i^T \mathbf{v}_j - s_{i,j}^{l})^2 . \quad (15)$$

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.* [75] 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 weighted non-negative matrix factorization method [108] to refine low-rank representation of users and items for better recommendation. Later, Luo *et al.* [78] 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.

Yu *et al.* [76] proposed HeteRec, which leverages the meta-path similarities to enrich the user-item interaction matrix R, so that more comprehensive representations of users and items can be extracted. HeteRec first defines L different types of meta-paths that connect users and items in the HIN. The item-item similarity in each path is measured with *PathSim* [107], which further forms L item-item similar matrices $S^{(l)} \in \mathbb{R}^{n \times n}$, where $l = 1, 2, \cdots, L$. Next, L diffused user preference matrices $\tilde{R}^{(q)}$ are calculated via the equation $\tilde{R}^{(l)} = RS^{(l)}$. Then L refined latent vectors of users and items in different meta-paths can be obtained via applying the non-negative matrix factorization technique [109] on these diffused user preference matrices,

$$\left(\hat{\mathbf{U}}^{(l)}, \hat{\mathbf{V}}^{(l)} \right) = \mathrm{argmin}_{\mathbf{U},\mathbf{V}} \left\| \tilde{R}^{(l)} - \mathbf{U}^T \mathbf{V} \right\|_F^2 \text{ s.t. } \mathbf{U} \geq 0, \quad \mathbf{V} \geq 0. \quad (16)$$

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

$$\hat{y}_{i,j} = \sum_{l=1}^{L} \theta_l \cdot \hat{\mathbf{u}}_i^{(l)T} \hat{\mathbf{v}}_j^{(l)}, \quad (17)$$

where θ_l is the weight for the user-item latent vector pair in the l-th path.

Later, Yu *et al.* [77] proposed HeteRec-p, which further considers the importance of different meta-paths should vary for different users. HeteRec-p first clusters users based on their past behaviors into c groups and generates personalized recommendation with the clustering information, instead of applying a global preference model. The modified scoring function becomes

$$\hat{y}_{i,j} = \sum_{k=1}^{c} \mathrm{sim}\left(\mathbf{C}_k, \mathbf{u}_i\right) \sum_{l=1}^{L} \theta_l^k \cdot \hat{\mathbf{u}}_i^{(l)T} \hat{\mathbf{v}}_j^{(l)}, \quad (18)$$

where $\mathrm{sim}\left(\mathbf{C}_k, \mathbf{u}_i\right)$ 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.

To overcome the limitation of the meta-path's representation ability, Zhao *et al.* [3] designed FMG by replacing the meta-path with the meta-graph. As a meta-graph contains richer connectivity information than a meta-path, FMG can capture the similarity between entities more accurately. Then, the model utilizes the matrix factorization (MF) to generate the latent vectors for both users and items in each meta-graph. Next, the factorization machine (FM) is applied to fuse the features of users and items 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.

The above-mentioned path-based methods only utilize the data of user's favored interacted items. Shi *et al.* [79] proposed the SemRec which considers the interaction of user's favored and hated past items. This framework utilizes a weighted HIN and weighted meta-path to integrate attribute values in the link. By modeling both positive and negative preference patterns, more accurate item relations and user similarity can be depicted via these paths to propagate the real user preference.

Another disadvantage of previous methods is the tedious requirement of tuning hyper-parameters, for example, the number of selected meta-paths. To lighten the burden, Ma *et al.* [84] proposed RuleRec to learn relations between associated items (co-buy, co-view, etc.) by exploiting the item's connectivity in an external KG. 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. Next, it summarizes explainable rules, which is in the form of meta-paths in the KG. The corresponding weight for each rule is further learned. Then, the item recommendation module integrates the learned rules and rule weights with the user purchase history to generate recommendations with the MF technique. Since the rules and rule weights are explicit, this model makes the recommendation process explainable.

Recently, some frameworks have been proposed to learn the explicit embedding of paths that connect user-item pairs in order to directly model the user-item relations. 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

$$\mathbf{h} = g(\mathbf{h}_p), p = 1, 2, \cdots, K, \tag{19}$$

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

$$\hat{y}_{i,j} = f(\mathbf{u}_i, \mathbf{v}_j, \mathbf{h}) \tag{20}$$

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. A common selection for $f(\cdot)$ is a fully-connected layer.

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 first uses a look up layer to embed the user-item pair. Next, it defines L meta-path that connects u_i and v_j and samples K path instances for each meta-path. These path instances are embedded with CNN to obtain the representations of each path instance $\mathbf{h}_p$. Then, meta-path embeddings are calculated by applying the max-pooling operation on embeddings of path instances that belong to each type of meta-path. These meta-path embeddings are aggregated to obtain the final interaction embedding $\mathbf{h}$ via an attention mechanism. The representations of the user and item also get updated via the attention mechanism with the final interaction embedding $\mathbf{h}$. Finally, the preference score is calculated via Equation 20, where $f(\cdot)$ is an MLP layer. Sun *et al.* [81] proposed a recurrent knowledge graph embedding (RKGE) approach that mines the path relation between user u_i and item v_j automatically, without manually defining 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. Then, each path constructed by the entity embedding sequence is fed into a recurrent network to encode the entire path. Next, following Equation 19, 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 20 becomes $\hat{y}_{i,j} = f(\mathbf{h})$, where $f(\cdot)$ is a fully-connected layer. By leveraging the information of semantic paths between entity pairs, a better representation for u_i and v_j will be obtained and further be integrated with the recommendation generation. Similarly, Wang *et al.* [83] proposed a knowledge-aware path recurrent network (KPRN) solution. KPRN constructs the extracted path sequence with both the entity embedding and the relation embedding. These paths are encoded with an LSTM layer and the preference of u_i for v_j in each path is predicted through fully-connected layers. By aggregating the score in each path via a weighted pooling layer, the final estimation of preference can be used for recommendation.

Huang *et al.* [86] designed EIUM, which captures users' dynamic interests for sequential recommendation. The recommendation module follows the schedule in Equation 19 and 20. First, each path connecting the user-item pair is encoded and be aggregated to obtain the interaction embedding $\mathbf{h}$ of the user-item pair (u_i, v_j). The dynamic preference embedding $\mathbf{p}$ is further obtained by applying the attention mechanism on the interaction sequential. The preference score can be modeled via $\hat{y}_{i,j} = f(\mathbf{h}, \mathbf{p})$. Besides the path-based recommendation module, EIUM further integrates a multi-modal fusion constraint module. This module introduces the KG structural constraint into the framework,

$$\begin{array}{ll} c2c: \mathbf{e}_{h_{f_c}} + \mathbf{r} \approx \mathbf{e}_{t_{f_c}}, & \text{s2s}: \mathbf{e}_{h_{f_s}} + \mathbf{r} \approx \mathbf{e}_{t_{f_s}}, \\ c2s: \mathbf{e}_{h_{f_c}} + \mathbf{r} \approx \mathbf{e}_{t_{f_s}}, & \text{s2c}: \mathbf{e}_{h_{f_s}} + \mathbf{r} \approx \mathbf{e}_{t_{f_c}}, \end{array} \tag{21}$$

where $(e_h, r, e_t) \in \mathcal{G}$, f_c denotes the content feature (textual, visual), and f_s denotes the structural feature. The loss function of this module is

$$\begin{aligned} \mathcal{L}_{KG} &= \mathcal{L}_{c2c} + \mathcal{L}_{s2s} + \mathcal{L}_{c2s} + \mathcal{L}_{s2c} \\ &= \frac{1}{4} \sum_i \|h + r - t\|, i \in \{c2c, s2s, c2s, s2c\}. \end{aligned} \tag{22}$$

This term can refine features of entities under the structural constraint of the KG. In this way, more accurate recommendation can be generated.

Recently, Xian *et al.* [85] proposed Policy-Guided Path Reasoning (PGPR) to use reinforcement learning (RL) to search for reasonable paths between user-item pairs. 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 sample paths between users and items by carefully designing the path searching algorithm, the transition strategy, terminal conditions, and RL rewards. In the prediction phase, PGPR can generate recommended items for users with specific paths to interpret the reasoning process. Later, Song *et al.* [87] proposed a similar model, EKar*, which adopts the RL technique in generating recommendation as well.

Summary for Path-based Methods. Path-based methods generate recommendations based on user-item graphs, and such methods have also been called HIN-based recommendation in the past. Traditional path-based methods [3], [75], [76], [77], [78], [79], [82] generally integrate MF with extracted meta-paths in HINs. These methods utilize path connectivity to regularize or enrich the user and/or item representation. The disadvantage of these methods is that they commonly need domain knowledge to define the type and number of meta-paths. RuleRec [84] tries to overcome the limitation by exploiting rules in an external KG in an automatic fashion. With the development of deep learning techniques, different models [1], [81], [83], [85], [86], [87] have been proposed to encode the path embedding explicitly. Recommendation can be generated with the path embeddings, or by discovering the most salient paths that connect user-item pairs.

Path-based methods naturally bring interpretability into the recommendation process. For traditional path-based methods, the motivation is to match the similarity of the item or user on the meta-path level. The recommendation results can find a reference from the pre-defined meta-paths. RuleRec utilize an external KG to generate rules for recommendation. Since the rule and corresponding weight are explicit, the reason for recommendation is also available to users. More recent works take advantage of deep learning models to mine salient paths for a user-item pair automatically, which reflects the recommendation process in the graph.

4.3 Unified Methods

As discussed in Section 4.1 and Section 4.2, embedding-based methods leverage the semantic representation of users/items in the KG for recommendation, while path-based methods use the semantic connectivity information, and both approaches utilize only one aspect of information in the graph. To fully exploit the information in the KG for better recommendations, unified methods which integrate both the semantic representation of entities and relations, and the connectivity information have been proposed. The unified method is based on the idea of embedding propagation. These methods refine the entity representation with the guidance of the connective structure in the KG. After obtaining the enriched representations of user u_i and/or the potential item v_j, the user's preference can be predicted with Equation 1.

The first group of works refine the user's representation from their interaction history. These works first extract multi-hop ripple sets $\mathcal{S}_{u_i}^k(k = 1, 2, \cdots, H)$ (defined in Section 3), where $\mathcal{S}_{u_i}^1$ is the triple set (e_h, r, e_t) in the graph with the head entities being the user u_i's engaged items. The general idea of this method is to learn the user embedding by utilizing the embeddings of past interacted items as well as multi-hop neighbors of these interacted items. The process of learning user representation $\mathbf{u_i}$ can be written in a general form as

$$\mathbf{u_i} = g_u\left(\left\{\mathcal{S}_{u_i}^k\right\}_{k=1}^H\right), \tag{23}$$

where $g_u(\cdot)$ is a function to concatenate embeddings of multi-hop entities with bias. Since the propagation starts from the user's engaged items, this process can be regarded as propagating the user's preference in the graph.

Wang *et al.* [14] proposed RippleNet, which is the first work to introduce the concept of preference propagation. Specifically, RippleNet first assigns entities in the KG with initial embeddings. Then it samples ripple sets $\mathcal{S}_{u_i}^k(k = 1, 2, \cdots, H)$ from the KG. To refine the user representation, the aggregation process can be illustrated as follows. Starting from $S_{u_i}^1$, every head entity interacts with the embedding of the candidate item v_j in turn via

$$p_i = \frac{\exp\left(\mathbf{v}_j^T \mathbf{R}_i \mathbf{e}_{h_i}\right)}{\sum_{(e_{h_k}, r_k, e_{t_k}) \in \mathcal{S}_{u_i}^1} \exp\left(\mathbf{v}_j^T \mathbf{R}_k \mathbf{e}_{h_k}\right)}, \tag{24}$$

where $\mathbf{R}_i \in \mathbb{R}^{d \times d}$ represents the embedding of relation r_i, and $\mathbf{e}_{h_i} \in \mathbb{R}^d$ is the embedding of head entity in the ripple set. During this process, the similarities of the candidate item v_j and head entities are calculated in the relation space. Then, the user's 1-order response of historical interaction can be calculated via

$$\mathbf{o}_{u_i}^1 = \sum_{(e_{h_i}, r_i, e_{t_i}) \in \mathcal{S}_{u_i}^1} p_i \mathbf{e}_{t_i}, \tag{25}$$

where $\mathbf{e}_{t_i}$ represents the embedding of the tail entity in the ripple set. The user's h-order $(h = 2, 3, \cdots, H)$ response $\mathbf{o}_{u_i}^h$ can be obtained by replacing $\mathbf{v}_j$ with the $(h-1)$-order response $\mathbf{o}_u^{h-1}$ in Equation 24, then interacting with head entities in h-hop ripple set $\mathcal{S}_u^h$ iteratively. The final representation of u_i can be obtained 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 with

$$\hat{y}_{i,j} = \sigma\left(\mathbf{u_i}^T \mathbf{v_j}\right), \tag{26}$$

where $\sigma(x)$ is the sigmoid function. In this way, RippleNet propagates the user's preference from historical interests along the path in the KG.

Similar to RippleNet, Tang *et al.* [92] proposed AKUPM, which models users with their click history. AKUPM first applies TransR for the entity representation. During each propagation process, AKUPM learns the relations between entities with a self-attention layer and propagates the user's preference toward different entities with bias. Finally, embeddings from different-order neighbors of interacted items are aggregated with the self-attention mechanism to obtain the final user representation. Later, Li *et al.* [95] extended the AKUPM and designed RCoLM. RCoLM jointly trains the KG completion module and the recommendation module, where AKUPM serves as the backbone. With the assumption that an item should have the same latent representation in the two modules, RCoLm unifies two modules and facilitates their mutual enhancement. Thus, RCoLM outperforms the AKUPM model.

The second group of works focus on refining the item representation $\mathbf{v}_j$ by aggregating embeddings of an item's multi-hop neighbors $\mathcal{N}_v^k (k = 1, 2, \cdots, H)$. A general description for this process is

$$\mathbf{v}_j = g_v \left(\left\{ \mathcal{S}_{v_j}^k \right\}_{k=1}^{H} \right), \tag{27}$$

where $\mathcal{S}_{v_j}^k$ is the ripple set of candidate item v_j, and $g_v(\cdot)$ is the function to concatenate embeddings of multi-hop neighbors. There are two steps to concatenate the embeddings of multi-hop neighbors. The first step is to learn a representation of candidate item v_j's k-hop neighbors,

$$\mathbf{e}_{\mathcal{S}_{v_j}^k} = \sum_{(e_h, r, e_t) \in \mathcal{S}_{v_j}^k} \alpha_{(e_h, r, e_t)} \mathbf{e}_t, \tag{28}$$

where $\alpha_{(e_h, r, e_t)}$ denotes the importance of different neighbors. Then for $e_h \in \mathcal{S}_{v_j}^k$, the representation can be updated by

$$\mathbf{e}_h = \text{agg} \left(\mathbf{e}_h, \mathbf{e}_{\mathcal{S}_{v_j}^k} \right), \tag{29}$$

where agg is the aggregation operator. During this process, the information of k-hop neighbors is aggregated with that of $(k-1)$-hop neighbors. Four types of aggregators are commonly used:

- **Sum Aggregator.** The sum aggregator sums two representations, followed by a nonlinear transformation.

$$\text{agg}_{\text{sum}} = \Phi \left(\mathbf{W} \cdot \left(\mathbf{e}_h + \mathbf{e}_{\mathcal{S}_{v_j}^k} \right) + \mathbf{b} \right). \tag{30}$$

- **Concat Aggregator.** The concat aggregator concatenates two representations, then applies a nonlinear transformation.

$$\text{agg}_{\text{concat}} = \Phi \left(\mathbf{W} \cdot \left(\mathbf{e}_h \oplus \mathbf{e}_{\mathcal{S}_{v_j}^k} \right) + \mathbf{b} \right). \tag{31}$$

- **Neighbor Aggregator.** The neighbor aggregator directly replaces the representation of an entity with representations from neighbors.

$$\text{agg}_{\text{neighbor}} = \Phi \left(\mathbf{W} \cdot \mathbf{e}_{\mathcal{S}_{v_j}^k} + \mathbf{b} \right). \tag{32}$$

- **Bi-Interaction Aggregator.** The bi-interaction aggregator considers both the sum and the element-wise product relations between entities. The second term allows more information to be passed from similar entities.

$$\begin{aligned} \text{agg}_{\text{Bi-Interaction}} =& \Phi \left(\mathbf{W} \cdot \left(\mathbf{e}_h + \mathbf{e}_{\mathcal{S}_{v_j}^k} \right) + \mathbf{b} \right) + \\ & \Phi \left(\mathbf{W} \cdot \left(\mathbf{e}_h \odot \mathbf{e}_{\mathcal{S}_{v_j}^k} \right) + \mathbf{b} \right). \end{aligned} \tag{33}$$

Wang *et al.* [89] proposed KGCN which models the final representation of a candidate item v_j by aggregating the embedding of entities in the KG from distant neighbors of v_j to v_j itself. KGCN first samples neighbors of the candidate item v_j in the KG, and it iteratively samples neighbors with a fixed number for each entity. Starting from the H-hop neighbors, it updates the representation of inner entities by replacing $k = H, H-1, \cdots, 1$ iteratively in Equation 29. During the aggregation process, the information of multi-hop neighbors can be propagated to the candidate item v_j inwardly. After this feature propagation process, the final representation of item v_j is a mixture of its initial representation and information from multi-hop neighbors. RippleNet and KGCN are two similar frameworks, the former models users by propagating the user's preference from historical interests outwardly, while the latter learns item representations from distant neighbors inwardly. Moreover, KGCN leverages the idea of GCN by sampling a fixed number of neighbors as the receptive field, which makes the learning process highly efficient and scalable. Recently, Wang *et al.* [91] proposed a follow-up approach, KGCN-LS, which further adds a label smoothness (LS) mechanism on the KGCN model. The LS mechanism takes the information of user interaction and propagates the user interaction labels on the KG, which is able to guide the learning process and obtain a comprehensive representation for the candidate item v_j.

RippleNet and its extension focus on using the embedding propagation mechanism on the item KG. Recently, some papers have explored the propagation mechanism in the user-item graph. Wang *et al.* [90] proposed KGAT, which directly models the high order relations between users and items with embedding propagation. KGAT first applies TransR to obtain the initial representation for entities. Then, it runs the entity propagation from the entity itself outwardly. During the outward propagation process, information from the entity e_i will be interacted with the multi-hop neighbors iteratively. The Equation 29 can be modified as

$$\mathbf{e}_i^{k+1} = \text{agg} \left(\mathbf{e}_i^k, \mathbf{e}_{\mathcal{S}_{e_i}^{k+1}} \right), k = 0, 1, \cdots, H-1, \tag{34}$$

where $\mathbf{e_i}^0$ represents the initial presentation of the entity, and $\mathbf{e_i}^k$ contains the connectivity information from k-hop neighbors. These H embeddings $\mathbf{e_i}^k$ are aggregated with bias to form the final representation $\mathbf{e_i}^*$. In this way, both the user representation and the item representation can be enriched with corresponding neighbors. The user preference is modeled via $\hat{y}_{u,v} = \mathbf{e}_u^{*T} \mathbf{e}_v^*$, where $\mathbf{e}_u^*$ and $\mathbf{e}_v^*$ stands for the final representation of the user u and item v, respectively.

Qu *et al.* [93] proposed KNI, which further considers the interaction between item-side neighbors and user-side neighbors, so that the refinement process of user embeddings and item embeddings are not separated. Zhao *et al.* [94] proposed IntentGC, which exploits rich user-related behaviors in the graph for better recommendation. They also designed a faster graph convolutional network to guarantee the scalability of IntentGC. Recently, Sha *et al.* [96] proposed AKGE, which learns the 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, then samples several 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 uses an attention-based GNN in this subgraph to propagate the information from neighbors for the final representation of this user-item pair. The construction of the subgraph filters out less related entities in the graph, facilitating mining high-order user-item relations for recommendation.

Summary for Unified Methods. Unified methods benefit from both the semantic embedding of the KG and semantic path patterns. These methods leverage the idea of embedding propagation to refine the representation of the item or user with multi-hop neighbors in the KG. These works generally adopt a GNN-based architecture that naturally fits the process of embedding propagation, and such methods have been a new research trend since the RippleNet [14] was proposed in 2018. Unified methods inherit interpretability from path-based methods. The propagation process can be treated as discovering user's preference patterns in the KG, which is similar to finding connectivity patterns in path-based methods.

4.4 Summary

Embedding-based methods preprocess the KG, either item graph or user-item graph, with KGE methods to obtain the embedding of entities and relations, which is further integrated into the recommendation framework. However, the informative connectivity patterns in the graph are ignored in this approach and few works can provide the recommendation results with reasons. Path-based methods utilize the user-item graph to discover path-level similarity for items, either by predefining meta-paths or mining connective patterns automatically. The path-based approach can also provide users with an explanation for the result. A recent research trend is to unify the embedding-based method and the path-based method to fully exploit information from both sides. Moreover, unified methods also have the ability to explain the recommendation process.

5 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 categorized 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 under 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 group KG based recommender systems according to the datasets which are summarized in Table 4. Generally, these works can be categorized into seven application scenarios and we will illustrate how different works construct the KG with each dataset.

- **Movie.** In this task, the recommender system needs to infer the user's preference based on movies watched in the past. Two datasets are most commonly used: MovieLens [110] and DoubanMovie. MovieLens maintains a set of datasets collected from the MovieLens website [111], among which three stable benchmark datasets with different rating numbers, MovieLens-100K, MovieLens-1M, and MovieLens-20M are most commonly used. Each dataset contains ratings, the movie's attributes and tags. DoubanMovie is crawled from Douban [112], a popular Chinese social media network. The dataset includes the social relation among users and the attributes of users and movies.

There are different ways to construct the movie-related KG for recommendation. Some papers [2], [14], [44], [45], [69], [70], [73], [88], [89], [91], [92], [93], [95] construct the movie-centric item graph to enrich the information of movies by extracting movies and related attributes from Satori, DBpedia, Freebase, CN-DBPedia, or IMDB [113]. In this way, movies are connected via attributes, including genres, countries, actors, directors, etc. This item graph serves as side information to facilitate the collaborative filtering module. Another approach is to directly take the user's rating as one type of relation and introduce the user to the graph. Some papers [1], [79], [82] build the user-item graph by directly leveraging the interaction data and attributes of movies inside the MovieLens dataset or the DoubanMovie dataset, while others [66], [75], [76], [77], [80], [81], [83], [86], [87], [96] still utilize external database to enrich the movie-side information.

- **Book.** Book recommendation is another popular task. There are five commonly used datasets: Book-Crossing [114], Amazon-Book [115], DoubanBook, DB-book2014, and IntentBooks [116]. Book-Crossing, DB-book2014, IntentBooks, and Amazon-Book contain binary feedback between users and books, and the KG for each dataset is built by mapping books to corresponding entities in Satori [2], [14], [45], [88], [89], [91], [92], [93], [95], DBpedia [70], [87], or Freebase [44], [90], [93]. The DoubanBook dataset is crawled from Douban [117], which contains both the user-item interaction data and books attributes, such as information about the author, publisher, and the year of publication. This work [82] builds the user-item graph by utilizing this knowledge in the DoubanBook dataset without the assistance of an external KG.
- **Music.** Last.FM [118] is the most popular dataset for music recommendation. The dataset contains information about users and their music listening records from the Last.fm online music system [119]. Some papers [44], [45], [89], [90], [91] construct the item graph by extracting music-related subgraphs from Freebase or Satori. Some papers [87], [96] build the user-item graph with knowledge from Freebase or Satori, while this paper [1] build the user-item graph from the Last.FM dataset directly. Another popular dataset is the KKBox dataset, which was released by the WSDM Cup 2018 Challenge [120]. This dataset contains both the user-item interaction data and the description of the music. Paper [73] builds the item graph and [83] builds the user-item graph from this dataset without leveraging any external databases.
- **Product.** The most popular dataset for the product recommendation task is the Amazon Product dataset [115]. This dataset includes multiple types of item and user information, such as interaction records, user reviews, product categories, product descriptions, and user behaviors. These works [3], [13], [67], [85], [94] build a user-item graph with this dataset alone, and [84] build the item graph by enriching the item information with the external Freebase database. There are also some papers [74], [94] use the data provided by Alibaba Taobao.
- **POI.** Point of Interest (POI) recommendation is the recommendation of new businesses and activities (restaurants,

TABLE 4
A collection of datasets for different application scenarios and corresponding papers.

Scenario	Dataset	Paper
Movie	MovieLens-100K	[1], [73], [75], [76], [77], [80]
	MovieLens-1M	[2], [14], [44], [45], [66], [70], [81], [83], [87], [92], [93], [95], [96]
	MovieLens-20M	[44], [86], [88], [89], [91], [93]
	DoubanMovie	[69], [79], [82]
Book	DBbook2014	[70], [87]
	Book-Crossing	[14], [45], [88], [89], [91], [92], [93], [95]
	Amazon-Book	[44], [90], [93]
	IntentBooks	[2]
	DoubanBook	[82]
News	Bing-News	[14], [45], [48], [88]
Product	Amazon Product data	[3], [13], [67], [84], [85], [94]
	Alibaba Taobao	[74], [94]
POI	Yelp challenge	[1], [3], [76], [77], [79], [80], [81], [82], [90], [96]
	Dianping-Food	[91]
	CEM	[71]
Music	Last.FM	[1], [44], [45], [87], [89], [90], [91], [96]
	KKBox	[73], [83]
Social Platform	Weibo	[68]
	DBLP	[78]
	MeetUp	[78]

museums, parks, cities, etc.) to users based on their historical check-in data. The most popular dataset is the Yelp Challenge [121], which contains the information of businesses, users, check-ins, and reviews. These papers [1], [3], [76], [77], [79], [80], [81], [82], [96] build a user-item graph with the data of check-ins, reviews and the attributes in the dataset, while [90] construct the item graph. Paper [71] utilizes the CEM dataset[1] to recommend next trip. Another work [91] uses the Dianping-Food dataset, which is provided by Dianping.com [122] for restaurant recommendation.

• **News.** News recommendation is challenging [48] because the news itself is time-sensitive, and the content is highly condensed, which requires commonsense to understand. Moreover, 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 [14], [45], [48], [88] to find the logical relations between different news and improve the precision of recommendation. The most popular dataset is Bing-News, collected from server logs of Bing News [123], which contains the user click information, news title, etc. To build a KG for news recommendation, the first step is to extract entities in the title. Then, subgraphs are constructed by extracting neighbors of these entities in Satori.

• **Social Platform.** This task is to recommend potentially interested people or meetings to users in the community. One application is to recommend unfollowed celebrities to users on the social platform Weibo [124] with the collected Weibo tweets data [68]. Despite the user-item graph to represent sentiment links between users and celebrities, an item graph with knowledge extracted from the Satori is built to enrich the information of celebrities. Another application is to recommend offline meetings for users on a social website, MeetUp [125], with data on that platform. The last application lies in the academic domain, to recommend conferences to researchers with the DBLP data [126].

6 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.* [127] 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, considering the nature of the KG has the potential to improve the performance of graph-based recommendation. For example, there may exist missing facts in the KG,

[1] an Amadeus database containing bookings over a dozen of airlines

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. [70], [95] 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 [45] and item relation regulation task [73]. 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 not equal across domains. For example, on the Amazon platform, book ratings are denser 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.* [128] proposed a matrix-based method for cross-domain recommendation. Later, Zhao *et al.* [129] 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 user-item interaction graph for better cross-domain recommendation performance.

• **Knowledge Enhanced Language Representation.** To improve the performance of various natural language processing tasks, there is a trend to integrate external knowledge into the language representation model. The knowledge representation and the text representation can be refined mutually. For example, Chen *et al.* [130] proposed the STCKA for short text classification, which utilizes the prior knowledge from KGs, such as YAGO, to enrich the semantic representation of short texts. Zhang *et al.* [131] 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 [48] 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 to achieve more accurate recommendation results.

• **Knowledge Graph Embedding Method.** There are two types of KGE methods, translation distance models and semantic matching models, based on the different constraints. In this survey, these two types of KGE methods are used in all three kinds of KG-based recommender systems and recommendation tasks. However, there is no comprehensive work to suggest under which circumstances, including data sources, recommendation scenarios, and model architectures, should a specific KGE method be adopted. Therefore, another research direction lies in comparing the advantages of different KGE methods under various conditions.

• **User Side Information.** Currently, most KG-based recommender systems build the graph by incorporating item side information, while few models consider user side information. However, user side information, such as the user network, and user's demographic information, can also be naturally integrated into the framework of current KG-based recommender systems. Recently, Fan *et al.* [132] used the GNN to represent a user-user social network and a user-item interaction graph separately, which outperforms traditional CF-based recommender systems with user social information. A recent paper in our survey [96] integrated user relations into the graph and showed the effectiveness of this strategy. Therefore, considering user side information in the KG could be another research direction.

7 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 work in this area.

Acknowledgments

The research work 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, 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 *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 1531–1540.

[2] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 353–362.

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

[4] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE transactions on knowledge and data engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[5] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, 2009.

[6] Z. Sun, Q. Guo, J. Yang, H. Fang, G. Guo, J. Zhang, and R. Burke, “Research commentary on recommendations with side information: A survey and research directions,” *Electronic Commerce Research and Applications*, vol. 37, p. 100879, 2019.

[7] S. Sen, J. Vig, and J. Riedl, “Tagommenders: connecting users to items through tags,” in *Proceedings of the 18th international conference on World wide web*. ACM, 2009, pp. 671–680.

[8] Y. Zhen, W.-J. Li, and D.-Y. Yeung, “Tagicofi: tag informed collaborative filtering,” in *Proceedings of the third ACM conference on Recommender systems*. ACM, 2009, pp. 69–76.

[9] L. Zheng, V. Noroozi, and P. S. Yu, “Joint deep modeling of users and items using reviews for recommendation,” in *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 2017, pp. 425–434.

[10] 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 *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2018, pp. 1356–1361.

[11] P. Massa and P. Avesani, “Trust-aware recommender systems,” in *Proceedings of the 2007 ACM conference on Recommender systems*. ACM, 2007, pp. 17–24.

[12] M. Jamali and M. Ester, “Trustwalker: a random walk model for combining trust-based and item-based recommendation,” in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 397–406.

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

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

[15] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, “Freebase: a collaboratively created graph database for structuring human knowledge,” in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, 2008, pp. 1247–1250.

[16] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. Van Kleef, S. Auer *et al.*, “Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia,” *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.

[17] F. M. Suchanek, G. Kasneci, and G. Weikum, “Yago: a core of semantic knowledge,” in *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007, pp. 697–706.

[18] A. Singhal, “Introducing the knowledge graph: things, not strings,” 2012, https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html.

[19] L. Ehrlinger and W. Wöß, “Towards a definition of knowledge graphs.” *SEMANTiCS (Posters, Demos, SuCCESS)*, vol. 48, 2016.

[20] 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*. Springer, 2017, pp. 1–14.

[21] S. Nurdiati and C. Hoede, “25 years development of knowledge graph theory: the results and the challenge,” *Memorandum*, vol. 1876, 2008.

[22] X. Huang, J. Zhang, D. Li, and P. Li, “Knowledge graph embedding based question answering,” in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 2019, pp. 105–113.

[23] D. Hakkani-Tür, A. Celikyilmaz, L. Heck, G. Tur, and G. Zweig, “Probabilistic enrichment of knowledge graph entities for relation detection in conversational understanding,” in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[24] A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka Jr, and T. M. Mitchell, “Coupled semi-supervised learning for information extraction,” in *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010, pp. 101–110.

[25] F. Belleau, M.-A. Nolin, N. Tourigny, P. Rigault, and J. Morissette, “Bio2rdf: towards a mashup to build bioinformatics knowledge systems,” *Journal of biomedical informatics*, vol. 41, no. 5, pp. 706–716, 2008.

[26] T. Pellissier Tanon, D. Vrandečić, S. Schaffert, T. Steiner, and L. Pintscher, “From freebase to wikidata: The great migration,” in *Proceedings of the 25th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 2016, pp. 1419–1428.

[27] M. Färber and A. Rettinger, “Which knowledge graph is best for me?” *arXiv preprint arXiv:1809.11099*, 2018.

[28] Google, “Freebase data dumps,” 2013, https://developers.google.com/freebase/data.

[29] G. A. Miller, *WordNet: An electronic lexical database*. MIT press, 1998.

[30] “The geonames geographical database,” 2006, http://www.geonames.org/.

[31] R. Qian, “Understand your world with bing,” *Bing search blog, Mar*, 2013.

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

[33] B. Xu, Y. Xu, J. Liang, C. Xie, B. Liang, W. Cui, and Y. Xiao, “Cn-dbpedia: A never-ending chinese knowledge extraction system,” in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*. Springer, 2017, pp. 428–438.

[34] “Wikipedia,” 2001, http://www.wikipedia.org/.

[35] “Nndb,” 2007, https://www.nndb.com/.

[36] “The fashion model directory,” 2000, http://www.fashionmodeldirectory.com/.

[37] “Musicbrainz,” 2000, https://musicbrainz.org/.

[38] “Baidu baike,” 2006, https://baike.baidu.com/.

[39] “Hudong baike,” 2005, http://www.baike.com/.

[40] “Wikidata,” 2012, http://www.wikidata.org/.

[41] “Under the hood: The entities graph,” 2013, https://www.facebook.com/notes/facebookengineering/under-the-hood-the-entitiesgraph/10151490531588920/.

[42] “Facebook,” 2004, https://www.facebook.com/.

[43] P. Ernst, C. Meng, A. Siu, and G. Weikum, “Knowlife: a knowledge graph for health and life sciences,” in *2014 IEEE 30th International Conference on Data Engineering*. IEEE, 2014, pp. 1254–1257.

[44] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, “Improving sequential recommendation with knowledge-enhanced memory networks,” in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 2018, pp. 505–514.

[45] H. Wang, F. Zhang, M. Zhao, W. Li, X. Xie, and M. Guo, “Multi-task feature learning for knowledge graph enhanced recommendation,” in *The World Wide Web Conference*, ser. WWW ’19. New York, NY, USA: ACM, 2019, pp. 2000–2010.

[46] 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 *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5458–5465.

[47] P. Zhao, H. Zhu, Y. Liu, J. Xu, Z. Li, F. Zhuang, V. S. Sheng, and X. Zhou, “Where to go next: A spatio-temporal gated network for next poi recommendation,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5877–5884.

[48] H. Wang, F. Zhang, X. Xie, and M. Guo, “Dkn: Deep knowledge-aware network for news recommendation,” in *Proceedings of the 2018 World Wide Web Conference*, ser. WWW ’18. Republic and Canton of Geneva, Switzerland: International World Wide Web Conferences Steering Committee, 2018, pp. 1835–1844.

[49] Z. Huang, Q. Liu, C. Zhai, Y. Yin, E. Chen, W. Gao, and G. Hu, “Exploring multi-objective exercise recommendations in online education systems,” in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 1261–1270.

[50] C. Qin, H. Zhu, C. Zhu, T. Xu, F. Zhuang, C. Ma, J. Zhang, and H. Xiong, “Duerquiz: A personalized question recommender system for intelligent job interview,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 2165–2173.

[51] X. Amatriain, J. M. Pujol, and N. Oliver, “I like it... i like it not: Evaluating user ratings noise in recommender systems,” in *International Conference on User Modeling, Adaptation, and Personalization*. Springer, 2009, pp. 247–258.

[52] G. Jawaheer, M. Szomszor, and P. Kostkova, “Comparison of implicit and explicit feedback from an online music recommendation service,” in *proceedings of the 1st international workshop on*

information heterogeneity and fusion in recommender systems, 2010, pp. 47–51.

[53] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *2008 Eighth IEEE International Conference on Data Mining*. Ieee, 2008, pp. 263–272.

[54] C. Wang, H. Zhu, C. Zhu, C. Qin, and H. Xiong, "Setrank: A setwise bayesian approach for collaborative ranking from implicit feedback," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[55] R. Salakhutdinov and A. Mnih, "Bayesian probabilistic matrix factorization using markov chain monte carlo," in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 880–887.

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

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

[58] C.-N. Ziegler, G. Lausen, and L. Schmidt-Thieme, "Taxonomy-driven computation of product recommendations," in *Proceedings of the thirteenth ACM international conference on Information and knowledge management*. ACM, 2004, pp. 406–415.

[59] J. Han, L. Zheng, Y. Xu, B. Zhang, F. Zhuang, S. Y. Philip, and W. Zuo, "Adaptive deep modeling of users and items using side information for recommendation," *IEEE transactions on neural networks and learning systems*, 2019.

[60] W.-T. Chu and Y.-L. Tsai, "A hybrid recommendation system considering visual information for predicting favorite restaurants," *World Wide Web*, vol. 20, no. 6, pp. 1313–1331, 2017.

[61] D. Liang, M. Zhan, and D. P. Ellis, "Content-aware collaborative music recommendation using pre-trained neural networks." in *ISMIR*, 2015, pp. 295–301.

[62] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention," in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*. ACM, 2017, pp. 335–344.

[63] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme, "Learning attribute-to-feature mappings for cold-start recommendations." in *ICDM*, vol. 10. Citeseer, 2010, pp. 176–185.

[64] 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 *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 2016, pp. 277–288.

[65] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[66] E. Palumbo, G. Rizzo, and R. Troncy, "Entity2rec: Learning user-item relatedness from knowledge graphs for top-n item recommendation," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*. ACM, 2017, pp. 32–36.

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

[68] H. Wang, F. Zhang, M. Hou, X. Xie, M. Guo, and Q. Liu, "Shine: Signed heterogeneous information network embedding for sentiment link prediction," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, 2018, pp. 592–600.

[69] D. Yang, Z. Guo, Z. Wang, J. Jiang, Y. Xiao, and W. Wang, "A knowledge-enhanced deep recommendation framework incorporating gan-based models," *IEEE International Conference on Data Mining (ICDM)*, pp. 1368–1373, 2018.

[70] 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 *The World Wide Web Conference*, ser. WWW '19. New York, NY, USA: ACM, 2019, pp. 151–161.

[71] A. Dadoun, R. Troncy, O. Ratier, and R. Petitti, "Location embeddings for next trip recommendation," in *Companion Proceedings of The 2019 World Wide Web Conference*. ACM, 2019, pp. 896–903.

[72] K. Joseph and H. Jiang, "Content based news recommendation via shortest entity distance over knowledge graphs," in *Companion Proceedings of The 2019 World Wide Web Conference*. ACM, 2019, pp. 690–699.

[73] X. Xin, X. He, Y. Zhang, Y. Zhang, and J. Jose, "Relational collaborative filtering: Modeling multiple item relations for recommendation," in *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR'19. New York, NY, USA: ACM, 2019, pp. 125–134.

[74] Y. Ye, X. Wang, J. Yao, K. Jia, J. Zhou, Y. Xiao, and H. Yang, "Bayes embedding (bem): Refining representation by integrating knowledge graphs and behavior-specific networks," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM, 2019, pp. 679–688.

[75] X. Yu, X. Ren, Q. Gu, Y. Sun, and J. Han, "Collaborative filtering with entity similarity regularization in heterogeneous information networks," *IJCAI HINA*, vol. 27, 2013.

[76] X. Yu, X. Ren, Y. Sun, B. Sturt, U. Khandelwal, Q. Gu, B. Norick, and J. Han, "Recommendation in heterogeneous information networks with implicit user feedback," in *Proceedings of the 7th ACM conference on Recommender systems*. ACM, 2013, pp. 347–350.

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

[78] C. Luo, W. Pang, Z. Wang, and C. Lin, "Hete-cf: Social-based collaborative filtering recommendation using heterogeneous relations," in *2014 IEEE International Conference on Data Mining*. IEEE, 2014, pp. 917–922.

[79] 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 *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, 2015, pp. 453–462.

[80] R. Catherine and W. Cohen, "Personalized recommendations using knowledge graphs: A probabilistic logic programming approach," in *Proceedings of the 10th ACM Conference on Recommender Systems*. ACM, 2016, pp. 325–332.

[81] Z. Sun, J. Yang, J. Zhang, A. Bozzon, L.-K. Huang, and C. Xu, "Recurrent knowledge graph embedding for effective recommendation," in *Proceedings of the 12th ACM Conference on Recommender Systems*, ser. RecSys '18. New York, NY, USA: ACM, 2018, pp. 297–305.

[82] C. Shi, B. Hu, W. X. Zhao, and S. Y. Philip, "Heterogeneous information network embedding for recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 2, pp. 357–370, 2018.

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

[84] W. Ma, M. Zhang, Y. Cao, W. Jin, C. Wang, Y. Liu, S. Ma, and X. Ren, "Jointly learning explainable rules for recommendation with knowledge graph," in *The World Wide Web Conference*. ACM, 2019, pp. 1210–1221.

[85] Y. Xian, Z. Fu, S. Muthukrishnan, G. de Melo, and Y. Zhang, "Reinforcement knowledge graph reasoning for explainable recommendation," *arXiv preprint arXiv:1906.05237*, 2019.

[86] 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 *Proceedings of the 27th ACM International Conference on Multimedia*. ACM, 2019, pp. 548–556.

[87] W. Song, Z. Duan, Z. Yang, H. Zhu, M. Zhang, and J. Tang, "Explainable knowledge graph-based recommendation via deep reinforcement learning," *arXiv preprint arXiv:1906.09506*, 2019.

[88] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, "Exploring high-order user preference on the knowledge graph for recommender systems," *ACM Transactions on Information Systems (TOIS)*, vol. 37, no. 3, p. 32, 2019.

[89] H. Wang, M. Zhao, X. Xie, W. Li, and M. Guo, "Knowledge graph convolutional networks for recommender systems," in *The World Wide Web Conference*, ser. WWW '19. New York, NY, USA: ACM, 2019, pp. 3307–3313.

[90] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "Kgat: Knowledge graph attention network for recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge*

Discovery & Data Mining, ser. KDD '19. New York, NY, USA: ACM, 2019, pp. 950–958.

[91] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, and Z. Wang, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ser. KDD '19. New York, NY, USA: ACM, 2019, pp. 968–977.

[92] X. Tang, T. Wang, H. Yang, and H. Song, "Akupm: Attention-enhanced knowledge-aware user preference model for recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 1891–1899.

[93] Y. Qu, T. Bai, W. Zhang, J. Nie, and J. Tang, "An end-to-end neighborhood-based interaction model forknowledge-enhanced recommendation," *arXiv preprint arXiv:1908.04032*, 2019.

[94] J. Zhao, Z. Zhou, Z. Guan, W. Zhao, W. Ning, G. Qiu, and X. He, "Intentgc: a scalable graph convolution framework fusing heterogeneous information for recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2019, pp. 2347–2357.

[95] 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.

[96] X. Sha, Z. Sun, and J. Zhang, "Attentive knowledge graph embedding for personalized recommendation," *arXiv preprint arXiv:1910.08288*, 2019.

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

[98] Q. Wang, Z. Mao, B. Wang, and L. Guo, "Knowledge graph embedding: A survey of approaches and applications," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 12, pp. 2724–2743, 2017.

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

[100] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Twenty-Eighth AAAI conference on artificial intelligence*, 2014.

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

[102] G. Ji, S. He, L. Xu, K. Liu, and J. Zhao, "Knowledge graph embedding via dynamic mapping matrix," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015, pp. 687–696.

[103] B. Yang, W.-t. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," *arXiv preprint arXiv:1412.6575*, 2014.

[104] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[105] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 135–144.

[106] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[107] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, vol. 4, no. 11, pp. 992–1003, 2011.

[108] S. Zhang, W. Wang, J. Ford, and F. Makedon, "Learning from incomplete ratings using non-negative matrix factorization," in *Proceedings of the 2006 SIAM international conference on data mining*. SIAM, 2006, pp. 549–553.

[109] C. H. Ding, T. Li, and M. I. Jordan, "Convex and semi-nonnegative matrix factorizations," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 1, pp. 45–55, 2008.

[110] "Movielens dataset," 1997, https://grouplens.org/datasets/movielens/.

[111] "Movielens website," 1997, https://movielens.org/.

[112] "Douban movie," 2005, http://movie.douban.com/.

[113] "Imdb," 1990, https://www.imdb.com/.

[114] "Book-crossing dataset," 2004, http://www2.informatik.uni-freiburg.de/∼cziegler/BX/.

[115] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2015, pp. 43–52.

[116] A. Uyar and F. M. Aliyu, "Evaluating search features of google knowledge graph and bing satori: entity types, list searches and query interfaces," *Online Information Review*, vol. 39, no. 2, pp. 197–213, 2015.

[117] "Douban book," 2005, http://book.douban.com/.

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

[119] "Last.fm online music system," 2002, http://www.last.fm/.

[120] "Kkbox dataset," 2018, https://wsdm-cup-2018.kkbox.events/.

[121] "Yelp challenge dataset," 2013, https://www.yelp.com/dataset/challenge/.

[122] "Dianping.com," 2009, https://www.dianping.com/.

[123] "Bing news," 2009, https://www.bing.com/news.

[124] "Sina weibo," 2009, http://weibo.com.

[125] "Meetup," 2002, http://www.meetup.com/.

[126] "Dblp dataset," 2013, https://dblp.uni-trier.de/xml/.

[127] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 2019, pp. 555–563.

[128] Q. Zhang, J. Lu, D. Wu, and G. Zhang, "A cross-domain recommender system with kernel-induced knowledge transfer for overlapping entities," *IEEE transactions on neural networks and learning systems*, 2018.

[129] C. Zhao, C. Li, and C. Fu, "Cross-domain recommendation via preference propagation graphnet," in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM, 2019, pp. 2165–2168.

[130] J. Chen, Y. Hu, J. Liu, Y. Xiao, and H. Jiang, "Deep short text classification with knowledge powered attention," vol. 33, no. 01, pp. 6252–6259, 2019.

[131] Z. Zhang, X. Han, Z. Liu, X. Jiang, M. Sun, and Q. Liu, "Ernie: Enhanced language representation with informative entities," pp. 1441–1451, 2019.

[132] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *The World Wide Web Conference*. ACM, 2019, pp. 417–426.