Knowledge-Adaptive Contrastive Learning for Recommendation

Hao Wang
Beijing University of Posts and Telecommunications
Beijing, China

Yao Xu
Researcher
Beijing, China

Cheng Yang*
Chuan Shi
Beijing University of Posts and Telecommunications
Beijing, China

Xin Li
Researcher
Beijing, China

Ning Guo
Researcher
Beijing, China

Zhiyuan Liu
Tsinghua University
Beijing, China

ABSTRACT

By jointly modeling user-item interactions and knowledge graph (KG) information, KG-based recommender systems have shown their superiority in alleviating data sparsity and cold start problems. Recently, graph neural networks (GNNs) have been widely used in KG-based recommendation, owing to the strong ability of capturing high-order structural information. However, we argue that existing GNN-based methods have the following two limitations. *Interaction domination*: the supervision signal of user-item interaction will dominate the model training, and thus the information of KG is barely encoded in learned item representations; *Knowledge overload*: KG contains much recommendation-irrelevant information, and such noise would be enlarged during the message aggregation of GNNs. The above limitations prevent existing methods to fully utilize the valuable information lying in KG. In this paper, we propose a novel algorithm named **K**nowledge-**A**daptive **C**ontrastive **L**earning (KACL) to address these challenges. Specifically, we first generate data augmentations from user-item interaction view and KG view separately, and perform contrastive learning across the two views. Our design of contrastive loss will force the item representations to encode information shared by both views, thereby alleviating the *interaction domination* issue. Moreover, we introduce two learnable view generators to adaptively remove task-irrelevant edges during data augmentation, and help tolerate the noises brought by *knowledge overload*. Experimental results on three public benchmarks demonstrate that KACL can significantly improve the performance on top-K recommendation compared with state-of-the-art methods.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

*corresponding author, yangcheng@bupt.edu.cn.

KEYWORDS

Recommender System, Knowledge Graph, Graph Neural Networks, Contrastive Learning

ACM Reference Format:
Hao Wang, Yao Xu, Cheng Yang*, Chuan Shi, Xin Li, Ning Guo, and Zhiyuan Liu. 2023. Knowledge-Adaptive Contrastive Learning for Recommendation. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining (WSDM '23), February 27–March 3, 2023, Singapore, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3539597.3570483

1 INTRODUCTION

Recommendation systems are widely used in real-world services (*e.g.*, e-commerce, advertising) to deliver the most relevant items to users and alleviate the information overload [3, 24, 44]. To predict user preference from interaction data, the classical collaborative filtering (CF) [6, 26], which transforms interaction information into latent representations, have achieved great success in many recommendation scenarios [10, 32, 43].

Although CF-based methods have achieved promising performance, they usually suffer from data sparsity and cold start problems. To mitigate this problem, many works introduce knowledge graphs (KGs), which can provide rich side information of items, into recommendation. Early works [2, 43] generate embeddings from KG triplets based on shallow neural networks to provide context information and assist the recommendation. Recently, graph neural networks (GNNs) show strong ability in modeling relational data, and have been extensively used in knowledge-based recommendation [22, 23, 25], owing to the effective aggregation of high-order information.

The core challenge of knowledge-based recommendation is to learn informative user and item representations from such structural knowledge [23, 31]. Despite the effectiveness of existing GNN-based methods, we argue that they fall short in two aspects as shown in Figure 1. *Interaction domination:* The supervision signal of user-item interaction will dominate the model training, which indicates that the information of KG is barely encoded in learned item representations. Although some CKG-based methods [13, 25], which construct collaborative knowledge graph (CKG) as a combination of user-item graph and KG, employ extra TransR [11] loss on KG side, they still fail to fundamentally solve this problem. As an evidence, we find that the attention scores of entity nodes are much

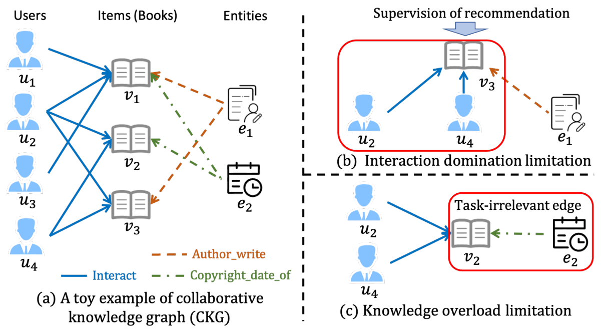

Users
Items (Books)
Entities
u_1
u_2
u_3
u_4
v_1
v_2
v_3
e_1
e_2
Author_write
Interact
Copyright_date_of
(a) A toy example of collaborative knowledge graph (CKG)
Supervision of recommendation
(b) Interaction domination limitation
Task-irrelevant edge
(c) Knowledge overload limitation

Figure 1: An illustration of two limitations in the state-of-the-art paradigm for KG-based recommendation (*i.e.,* GNN methods based on CKG modeling).

lower than those of users, when performing information propagation to item nodes in KGAT [25]. Thus, existing methods can not sufficiently capture and utilize the valuable information lying in KG. *Knowledge overload:* KG contains many recommendation-irrelevant triplets, *e.g.*, the triplet (e_2, *copyright_date_of*, v_1) in Figure 1 can hardly be used for recommending books to users. Besides, observed user-item interactions also contain noises. As previous research [33] mentioned, the neighborhood aggregation scheme in GNNs will inevitably enlarge the impact of such noises.

Inspired by the recent success of contrastive learning (CL) technique, in this work we propose **K**nowledge-**A**daptive **C**ontrastive **L**earning (KACL) method to address the above limitations. We first build two augmented views (*i.e.,* interaction view and knowledge view) via edge dropping, and perform message aggregation separately. Then our contrastive loss will force the representations of an item across two views to be closer to each other, while those of different items apart. Ideally, with the help of contrastive learning, the item representations will only encode the information shared between interaction and knowledge views, thereby alleviating the domination of interaction information and the noise of recommendation-irrelevant knowledge. To facilitate the removal of information unrelated to recommendation, we propose two learnable view generators to adaptively drop possibly unimportant edges in the data augmentation of contrastive learning. The graph encoders used to build user, item and entity representations are based on GNNs with relation-aware modifications to capture high-order connectivity. In this way, KACL can learn high-quality representations and help improve the recommendation performance. Finally, we leverage a multi-task training manner to jointly optimize the contrastive loss with the classic recommendation loss and KG loss to better encode the KG structure.

Experimental results on three public datasets show that KACL achieves consistent and significant improvement over state-of-the-art baselines on KG-based recommendation task. The relative improvements of ndcg@20 are 3.86%, 16.77% and 5.06% on Amazon-Book, LastFM and Movielens, respectively. Ablation studies and the results of cold start users/items further demonstrate our effectiveness. In addition, we analyze learned view generators and find that KACL is capable of identifying and removing task-irrelevant edges.

We summarize the contributions of this work as follows:

- We highlight two key limitations in the paradigm of state-of-the-art methods for KG-based recommendation, *i.e., interaction domination* and *knowledge overload.*
- We propose a novel model named KACL, that can automatically drop task-irrelevant edges and encode the information shared between user-item interaction view and KG view for high-quality node representations.
- Experiments on three public benchmarks show that KACL can significantly improve the performance on top-K recommendation compared with state-of-the-art baselines.

2 RELATED WORK

This work is mainly relevant to two research lines: knowledge graph-based recommendation and graph contrastive learning.

2.1 Knowledge Graph-Based Recommendation

To alleviate the data sparsity problem in recommendation tasks, many researchers leverage knowledge graph (KG) as additional side information of items. Existing knowledge-based recommendation methods roughly fall into three categories: embedding-based methods [2, 21, 43], path-based methods [7, 20, 28] and GNN-based methods [13, 18, 25, 31]. Embedding-based methods learn entity representations by knowledge graph embedding (KGE) techniques (*e.g.* TransE [1] and TransR [11]), and then use them as extra context information to assist the recommender systems. Path-based methods capture long-range connectivity by designing connection patterns among items in KG (*e.g.*, meta-path, meta-graph). Recently, with the development of graph neural network (GNN), many methods adopt GNNs as the basic models to jointly learn user preference and perform KG completion. We can further group these methods into two categories: The first category [14, 18, 22, 23, 31], represented by KGCN [23], mainly considers the user preference to the relations in KG, and learns user-specific representation for each item based on information aggregation of GNNs. The second category is CKG-based modeling represented by KGAT [25], which constructs the hybrid structure of KG and user-item interaction graph as collaborative knowledge graph (CKG), and then performs information propagation over the CKG via attention mechanism. The CKG-based paradigm has shown its superiority and thus becomes the mainstream backbone of recent literatures [13, 25, 27, 30, 35]. However, as illustrated in Figure 1, the state-of-the-art CKG-based paradigm still has two key limitations to be addressed.

2.2 Contrastive Learning for Recommendation

Inspired by the success of contrastive learning on language modeling and visual representation learning, there have been some recent works [33, 34, 40–42] introducing CL into recommender system. For example, SGL [33] designs three operators to generate augmented views and then propose a multi-task strategy to jointly optimize contrastive loss and recommendation loss. SEPT [40] proposes a general socially-aware contrastive learning framework by mining extra social information of users. Most recently, few papers [16, 39, 45, 46] focus on KG-based recommendation with contrastive learning. CKER [16] introduces a CL module to learn sharing user preferences by deriving additional supervision signals. KGCL [39] designs knowledge graph augmentation schema

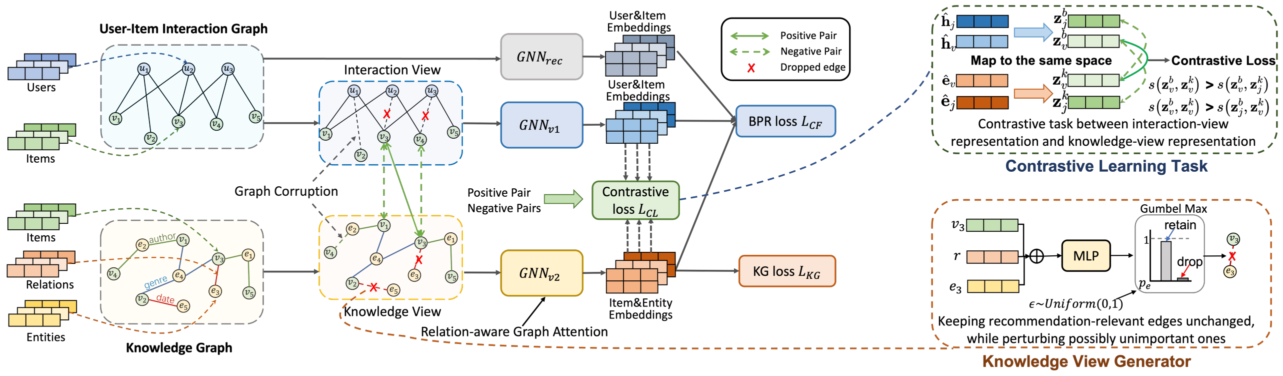

User-Item Interaction Graph
Users
Items
Interaction View
Graph Corruption
Items
Relations
Entities
Knowledge Graph
Knowledge View
Relation-aware Graph Attention
author
genre
date
GNN_{rec}
GNN_{v1}
GNN_{v2}
User&Item Embeddings
User&Item Embeddings
Item&Entity Embeddings
Positive Pair
Negative Pairs
Contrastive loss L_{CL}
BPR loss L_{CF}
KG loss L_{KG}
Positive Pair
Negative Pair
Dropped edge
Map to the same space
Contrastive Loss
$s(\mathbf{z}_v^b, \mathbf{z}_v^k) > s(\mathbf{z}_v^b, \mathbf{z}_j^k)$
$s(\mathbf{z}_v^b, \mathbf{z}_v^k) > s(\mathbf{z}_j^b, \mathbf{z}_v^k)$
Contrastive task between interaction-view representation and knowledge-view representation
Contrastive Learning Task
Gumbel Max
retain
drop
MLP
$\epsilon \sim Uniform(0,1)$
Keeping recommendation-relevant edges unchanged, while perturbing possibly unimportant ones
Knowledge View Generator

Figure 2: The model architecture of our proposed Knowledge-Adaptive Contrastive Learning (KACL).

to suppress noise in information aggregation and then proposes a knowledge-guided contrastive learning paradigm to derives more robust node representations. However, these CL modules cannont address the main limitations (*i.e. Interaction domination* and *Knowledge overload*) in recent works about KG-based recommendation.

3 PROBLEM FORMULATION

We begin by presenting the user-item interaction graph and knowledge graph, and then describe the KG-based recommendation task.

User-Item Interaction Graph: Following the settings in GNN-based recommendation [5, 25, 26], we construct a user-item bipartite graph $\mathcal{G}_b = \{(u, y_{uv}, v)\}$ by historical user-item interactions (e.g. consume, view, click), where $u \in \mathcal{U}$ and $v \in \mathcal{V}$ denote the user and item of an interaction, $\mathcal{U}$ and $\mathcal{V}$ are the set of users and items respectively, and y_{uv} is a binary value used to record whether there is an interaction between the user and the item. If user u has interacted with item v, $y_{uv} = 1$, otherwise $y_{uv} = 0$.

Knowledge Graph: In the KG-based recommendation problem, we also have a knowledge graph $\mathcal{G}_k = \{(h, r, t)\}$ available, where each triplet (h, r, t) describes that there is a relationship $r \in \mathcal{R}$ from head entity $h \in \mathcal{I}$ to tail entity $t \in \mathcal{I}$, where $\mathcal{I}$ and $\mathcal{R}$ denote the sets of entities and relations in the knowledge graph. Here, $\mathcal{I}$ is comprised of items $\mathcal{V}$ and non-item entities $\mathcal{I} \backslash \mathcal{V}$. For example, the triple (*Leonardo DiCaprio, ActorOf, Titanic*) states the fact that Leonardo DiCaprio is the leading actor in movie "Titanic". Note that $\mathcal{R}$ contains relations in both canonical direction (e.g., ActorOf) and inverse direction (e.g., ActedBy).

KG-based Recommendation: Given interaction graph $\mathcal{G}_b$ and knowledge graph $\mathcal{G}_k$, our goal is to learn a function $y(u, v)$ that can predict the probability that user u will interact with item v.

4 METHODOLOGY

In this section, we present our Knowledge-Aware Contrastive Learning framework (KACL), for the top-K recommendation task. Figure 2 presents the model architecture of our proposed KACL. We will first introduce a classical GNN-based recommender, which is used as backbones in this work and many previous KG-based recommendation systems [13, 25]. Then we will present our KACL algorithm for learning high-quality representations. Finally, we will present how to train the model via a multi-task learning manner.

4.1 Classical GNN-based Recommender

Recommender systems aim to refine collaborative information from the user-item interactions and measure the user preference on items. To exploit high-order connectivity in user-item interaction graph and perform high-quality recommendation, most existing models on KG-based recommendation [5, 25, 26] employ graph neural networks (GNNs) to learn a representation vector for each node after hierarchical message aggregations on graph $\mathcal{G}_b$. Inspired by the interpretability and effectiveness of GAT [19], many previous methods adopt GAT as the base architecture for recommendation.

4.1.1 ***Graph Attention Network.*** We start by describing a single layer of GAT [19], which consists of neighborhood aggregation and representation updating. Considering a node i, we use $\mathcal{N}_i$ to represent the neighbors of node i. To aggregate the first-hop structure of each node i, we first compute the linear combination of its neighbors' representations by $\mathbf{h}_{\mathcal{N}_i} = \sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{h}_j$, where $\mathbf{h}_j$ is the representation of node j, and α_{ij} denotes the attention score of node i to node j, calculated as follows:

$$\alpha_{ij} = \frac{\exp(LeakyReLU(a_b^T [W_b \mathbf{h}_i || W_b \mathbf{h}_j]))}{\sum_{k \in \mathcal{N}_i} \exp(LeakyReLU(a_b^T [W_b \mathbf{h}_i || W_b \mathbf{h}_k]))}, \tag{1}$$

where $||$ is the concatenation operation, a_b and W_b are trainable parameters.

Next, we employ GCN Aggregator [9] to update the representation of node i. More formally, the representation of node i in the l-th layer is updated as:

$$\mathbf{h}_i^{(l)} = \sigma(W_b^{(l)} (\mathbf{h}_i^{(l-1)} + \mathbf{h}_{\mathcal{N}_i}^{(l-1)})), \tag{2}$$

where $W_b^{(l)}$ is the learnable weight matrix in the l-th layer, σ is the activation function. Note that the representations $\mathbf{h}_i^{(0)}$ of every node i are free parameters to be trained. After stacking L GAT layers, the layer-aggregation mechanism [37] is adopted to generate the final representations $\mathbf{h}_i$ for prediction.

4.1.2 ***Recommendation Loss.*** Given the final representations of a user u and an item v, we can employ inner product $y(u, v) = \mathbf{h}_u{}^T \mathbf{h}_v$ to measure the user preference on item v.To optimize model parameters, the BPR loss [17] is a common choice to capture the pair-wise preference.

$$\mathcal{L}_{CF}(u, v^+, v^-) = -\log\sigma(y(u, v^+) - y(u, v^-)). \tag{3}$$

where (u, v^+) is the positive interaction, (u, v^-) is a random negative interaction, and σ is the Sigmoid function. We name the GNN encoder based on GAT used for $\mathcal{L}_{CF}$ as GNN_{rec}.

4.2 Knowledge-Adaptive Contrastive Learning

The contrastive learning part consists of three key components: (1) adaptive view generations from interaction and knowledge graphs separately; (2) relation-aware structural encoding for user/item/entity on augmented graphs; (3) contrastive learning task, which forces item representations to encode information shared by both views.

4.2.1 ***Adaptive Data Augmentation on Graph Structure.*** To conduct contrastive learning, we first derive two different augmented views from interaction graph and knowledge graph, respectively. For brevity, we name the view over the interaction graph as *interaction view*, and the view over the knowledge graph as *knowledge view*. Unlike the data augmentation in previous CL-based studies [36, 40] that only extract the relationship among a specific type of nodes, we propose to preserve the node/edge heterogeneity in augmented views. Such heterogeneity can retain more information of original data, and is essential for the knowledge view where the contributions of different relations to recommendation vary a lot. To fully explore the cross-view information helpful for recommendation, we propose to design augmentation strategies that tend to keep important and recommendation-relevant edges unchanged, while perturbing possibly unimportant ones.

Specifically, we introduce a novel augmentation method, which first corrupts the input graphs by randomly removing a certain ratio of edges and then employs two learnable view generators to further remove unimportant edges, respectively. For graph corruption, we directly perform perturbation to the entire graphs $\mathcal{G}_b$ and $\mathcal{G}_k$ by randomly dropping out edges with a certain ratio ρ in every epoch, which is similar to previous works [33, 40], and then we can obtain sampled subsets $\tilde{\mathcal{E}}_b$ and $\tilde{\mathcal{E}}_k$ of $\mathcal{G}_b$ and $\mathcal{G}_k$, respectively.

In order to further filter out noise and recommendation-irrelevant information, we introduce two view generators (*i.e.*, interaction view generator and knowledge view generator) to select a modified subset $\hat{\mathcal{E}}$ from $\tilde{\mathcal{E}}$. Formally, the generators will model a real-valued importance weight w_e and calculate sampling probability $p_e \in \{0, 1\}$ for each edge e. Edge e will be retained in $\hat{\mathcal{E}}$ if $p_e = 1$ and dropped otherwise.

For interaction view generator, the weight w_e^b is defined as:

$$w_e^b = MLP_b([\mathbf{h}_u^{(0)} || \mathbf{h}_v^{(0)}]), \tag{4}$$

where edge $e = (u, v)$, w_e^b denotes the edge importance, MLP is short for multi-layer perceptron, $\mathbf{h}_u^{(0)}, \mathbf{h}_v^{(0)}$ are user and item embeddings, respectively. A higher score of w_e^b suggests that the edge is more likely to be critical and should be preserved. To make the edge dropping procedure differentiable and enable an end-to-end optimization process, we relax the discrete p_e^b to a continuous variable in $[0, 1]$ and apply the Gumbel-Max reparameterization trick [12, 15]. Specifically, the probability p_e^b is calculated by:

$$p_e^b = \sigma((\log(\epsilon) - \log(1 - \epsilon) + w_e^b)/\tau_b) \tag{5}$$

where random variable $\epsilon \sim \text{Uniform}(0, 1)$, $\sigma(\cdot)$ is the Sigmoid function, and temperature hyper-parameter τ_b is used to control the approximation. When τ_b goes to 0, p_e^b will move towards binary. We denote the augmented interaction graph as $\hat{\mathcal{G}}_b$. In practice, we can multiply the sampling probability to the aggregation function as approximation and thus enable end-to-end training.

Compared with interaction view, knowledge view has multiple types of relations. Previous work [30] has shown that the semantic meanings and importance of different relations are quite diverse from each other, and there are a massive amount of recommendation-irrelevant interactions in KGs. To tackle this issue, we model the edge weight w_e^k in knowledge view by jointly considering entities and relations. Given a triplet $e = (h, r, t)$ from knowledge view, the weight w_e^k can be calculated by:

$$w_e^k = MLP_k(W_r^k(\mathbf{e}_h^{(0)} || \mathbf{e}_t^{(0)})), \tag{6}$$

where W_r^k is the transformation matrix of relation r, $\mathbf{e}_h^{(0)}$ and $\mathbf{e}_t^{(0)}$ are entity embeddings. A higher score of w_e^k suggests that the triplet is more likely to be useful in recommendation. After getting weight w_e^k, we can generate augmented knowledge graph $\hat{\mathcal{G}}_k$ in a similar way as interaction view generator, and employ hyper-parameter τ_k to control the probability p_e^k.

4.2.2 ***Relation-aware Graph Attention for Node Encoding.*** Given the augmented graphs in interaction and knowledge views, we use two different graph encoders to capture the high-order structural context as node representations.

For interaction view, we denote the encoder as GNN_{v1}, which employs the same GNN architecture as classical recommender GNN_{rec}. Given the augmented graph $\hat{\mathcal{G}}_b$ and the initial representation $\hat{\mathbf{h}}_i^{(0)}$, GNN_{v1} can effectively encode the high-order context into $\hat{\mathbf{h}}_i$ for every user or item node i as illustrated in Section 4.1.

Though the architecture of GNN_{v1} is powerful in modeling interaction view, it may not be the best choice for knowledge view due to the neglect of relation types. To solve this problem, we extend the original attention mechanism by considering the influence of relations, and get the heterogeneous graph encoder GNN_{v2} specialized for knowledge view. In specific, we first allocate a learnable embedding for each relation and entity, and then calculate the attention score by incorporating the relational embedding into attention calculation. Formally, the attention score between head entity h and tail entity t in relation type $r(\langle h, t\rangle) \in \mathcal{R}$ is computed as:

$$\alpha_{ht} = \frac{\exp(LeakyReLU(a_k^T[W_k\mathbf{e}_h || W_r\mathbf{m}_{r(\langle h,t\rangle)} || W_k\mathbf{e}_t]))}{\sum_{j\in\mathcal{N}_h}\exp(LeakyReLU(a_k^T[W_k\mathbf{e}_h || W_r\mathbf{m}_{r(\langle h,j\rangle)} || W_k\mathbf{e}_j]))}, \tag{7}$$

where $\mathbf{e}_h$ and $\mathbf{e}_t$ are entity embeddings, $r(\langle h, t\rangle)$ denotes the relation type between head entity h and tail entity t, and $\mathbf{m}_{r(\langle h,t\rangle)}$ represents the embedding of relation $r(\langle h, t\rangle)$. Other modules of GNN_{v2} are the same as those of GNN_{v1}. Given the knowledge view encoder GNN_{v2} and augmented graph $\hat{\mathcal{G}}_k$, we can obtain the final node embeddings $\hat{\mathbf{e}}_i$ after multiple layers' message aggregation.

4.2.3 ***Contrastive Learning Task.*** Note that the representation space of interaction and knowledge views are different. Therefore, we will feed item embeddings $(\hat{\mathbf{h}}_v, \hat{\mathbf{e}}_v)$ from the two views into two corresponding MLPs, mapping them into the same space $(\mathbf{z}_v^b, \mathbf{z}_v^k)$ where contrastive loss is calculated.

Now we will introduce our strategy of selecting contrastive pairs. Our goal is to encourage graph encoders to preserve the information shared across interaction and knowledge views. Thus, for each item v, we treat the two views of the same item as a positive pair $\{\mathbf{z}_v^b, \mathbf{z}_v^k\}$. On the other side, we will pair item v with a random item j, and get $\{\mathbf{z}_v^b, \mathbf{z}_j^k\}$ and $\{\mathbf{z}_j^b, \mathbf{z}_v^k\}$ as negative pairs. Then for each item v, the contrastive loss will encourage the consistency between the representations of its different views, while enforce the divergence of negative pairs. Formally, we adopt an extension version based on InfoNCE loss [4]:

$$\mathcal{L}_{CL}(v) = -\log \frac{\exp(s(\mathbf{z}_v^b, \mathbf{z}_v^k)/\tau_{cl})}{\sum_{j \in \mathbb{N} \cup \{v\}} \exp(s(\mathbf{z}_v^b, \mathbf{z}_j^k)/\tau_{cl}) + \exp(s(\mathbf{z}_j^b, \mathbf{z}_v^k)/\tau_{cl})} \tag{8}$$

where $s(\cdot)$ measures the cosine similarity of two vectors, $\mathbb{N}$ is the set of negative samples, and τ_{cl} is the temperature hyper-parameter.

4.3 Model Prediction and Optimization

4.3.1 ***The Overall Loss Function.*** Note that the contrastive learning module only focuses on learning item representations in a self-supervised manner, and ignores the supervision of collaborative filtering. Therefore, we need an additional recommendation loss to learn user preference. In particular, we simply concatenate the user/item embeddings from the recommender and contrastive modules, and employ inner product as score function. Also, since knowledge view only contains item embeddings, we will allocate a trainable embedding $\hat{\mathbf{e}}_u$ for each user. Then we can compute the user preference function $y(u, v)$ as:

$$y(u, v) = (\mathbf{h}_u || \hat{\mathbf{h}}_u || \hat{\mathbf{e}}_u)^T (\mathbf{h}_v || \hat{\mathbf{h}}_v || \hat{\mathbf{e}}_v). \tag{9}$$

Then the recommendation loss is the same as Eq. (3).

Besides, to effectively encode KG in recommender systems, we add an auxiliary regularization term to the knowledge view. Following DistMult [38], we will optimize the scoring function below to strengthen the KG information in $\hat{\mathbf{e}}_i$:

$$f(h, r, t) = \hat{\mathbf{e}}_h^T R_r \hat{\mathbf{e}}_t,, \tag{10}$$

where R_r is a transformation matrix of relation r. A lower score of $f(h, r, t)$ suggests that it is more likely to be true. The training loss of the regularization term is also a pair-wise ranking one:

$$\mathcal{L}_{KG}(h, r, t^+, t^-) = -\log \sigma(f(h, r, t^-) - f(h, r, t^+)), \tag{11}$$

where (h, r, t^+) is a positive triplet in knowledge graph and (h, r, t^-) is a negative one by replacing tail entity randomly.

Finally, we leverage a multi-task training strategy to jointly optimize the recommendation loss, the contrastive learning loss and the knowledge graph regularization term. The overall loss function is:

$$\mathcal{L} = \mathcal{L}_{CF} + \lambda_1 \mathcal{L}_{CL} + \lambda_2 \mathcal{L}_{KG}, \tag{12}$$

where λ_1, λ_2 are hyper-parameters to balance different terms. In practice, λ_1, λ_2 are fixed as 0.1 and 1, respectively.

Table 1: The statistics of datasets.

Dataset	Amazon-Book	LastFM	Movielens
# Users	70,679	23,566	37,385
# Items	24,915	48,123	6,182
# Interactions	846,434	3,034,763	539,300
# Entities	113,487	106,389	24,536
# Relations	39	9	20
# Triplets	2,557,746	464,567	237,155

4.3.2 ***Optimization Details and Complexity.*** To optimize the multi-task objective in Eq. (12), we decouple the training process into three iterative stages: knowledge graph regularization, contrastive learning and recommendation task. We iteratively update the corresponding parameters to minimize the losses until we reach the best performance on the validation set. In particular, we update the parameters of two view generators in the recommendation loss, and then freeze the parameters in the contrastive learning to adaptively filter out unimportant and recommendation-irrelevant edges. All the randomness in data augmentation will be re-sampled in every epoch. For negative samples, we will use all other items in the same batch. We employ Adam [8] optimizer for parameter training. The time and space complexity of KACL is linear with the scale of the dataset (*i.e.*, number of nodes and edges), which enables an efficient inference process. Code and more implementation details are provided in https://github.com/wsdmanonymous/KACL.

5 EXPERIMENTS

5.1 Experimental Settings

5.1.1 ***Dataset Description.*** To evaluate the effectiveness of KACL, we conduct experiments with three benchmark datasets: **Amazon-Book**[1], **LastFM**[2] and **Movielens**[3], which are publicly available and have been used in existing work [13, 22, 23, 25] on knowledge-based recommendation. All the above datasets adopt the 10-core setting (*i.e.* filtering out the low-frequency users and items which appear less than ten times) to ensure the quality of interaction data. The statistics of the three datasets are shown in Table 1. Following the same setting of KGAT [13, 25, 29], for each user, we randomly split 80% of interactions for training, and 20% interactions for testing. From the training set, we randomly select 10% of interactions as validation set to tune hyper-parameters.

5.1.2 ***Evaluation Metrics.*** We adopt two widely used metrics of top-K recommendation and preference ranking tasks: recall@K and ndcg@K, where K is set as 20. For each user in the test sets, recall@K indicates the percentage of one's rated items that emerge in the top K recommended items. ndcg@K is the normalized discounted cumulative gain at K, which takes the position of correctly recommended items into account. We report the average metrics of all users in the testing set. Larger values indicate better performance.

5.1.3 ***Baselines.*** To verify the effectiveness of our method, we compare KACL with the following recommendation baselines:

[1]http://jmcauley.ucsd.edu/data/amazon/
[2]https://grouplens.org/datasets/hetrec-2011/
[3]https://grouplens.org/datasets/movielens/

• **BPR-MF** [17] is a classic matrix factorization model that only considers the user-item interactions.

• **CKE** [43] combines collaborative filter (CF) module with knowledge embeddings of items derived from TransR.

• **KGCN** [23] presents a GNN-based model, which considers user preference on KG relations and generates user-specific item representations by weighted information aggregation.

• **KGNNLS** [22] provides a novel label smoothness regularization over the edge weights, which provides better inductive bias.

• **KGAT** [25] constructs a collaborative knowledge graph (CKG) from KG and user-item graph and applies an attentive aggregation mechanism to generate user and item representations.

• **CKAN** [31] further extends the KGCN by considering the user-item graph. Specifically, it utilizes different neighborhood aggregation strategies on the user-item graph and KG respectively.

• **KGPL** [18] proposes a augmenting labeled samples paradigm through pseudo-labelling to alleviate cold-start problem.

• **DSKReG** [30] employs a relational GNN with differentiable sampling strategy on CKG to learn user preference.

• **KGIN** [27] is a state-of-the-art KG-based method, which uses auxiliary item knowledge to explore the users' intention behind the user-item interactions.

• **CKER** [16] introduces self-supervised learning to maximize the mutual information between the interaction-side and knowledge-side user preferences by designing additional supervision signal.

• **KGCL** [39] integrates knowledge augmentation schema into a cross-view CL module to alleviate the influence of information noise for knowledge graph-enhanced recommender systems.

5.1.4 ***Parameter Settings.*** For a fair comparison, we set the embedding dimension as 64, the optimizer as Adam, and the batch size as 8192 for all models. For each baseline, all other hyper-parameters are set following the suggestions from the original settings in their papers. We randomly initialize the model parameters with Xavier initializer. As suggested in [13, 25, 27], pre-trained MF embeddings is used to stabilize and speed up the model training.

Other hyper-parameters of KACL are set as follows: the number of GNN layers are searched in $\{1, 2, 3, 4\}$ and set as 2 in all experiments; the temperature τ_{cl} is tuned in $\{0.5, 0.6, 0.7, 0.8, 0.9\}$ and set as $\tau = 0.7$; the edge dropout rate of graph corruption is set to $\rho = 0.3$ in all experiments; the temperature pair (τ_b, τ_k) of view generators is set to $(0.03, 1.5)$, $(0.7, 1.5)$ and $(0.5, 1.5)$ in Amazon-Book, LastFM and Movielens, respectively. To avoid over-fitting, we also apply L2 regularization (coefficients are set to 10^{-3}) and dropout (dropout rate is 0.1) in every layer of our proposed KACL.

5.2 Performance Comparison

The results of the top-K recommendation are presented in Table 2. Based on the experimental results, we can observe that:

• KACL consistently and significantly outperforms all state-of-the-art baselines across three datasets in terms of all measures. This observation demonstrates the robustness and effectiveness of KACL. We attribute these improvements to the relational modeling of KACL: (1) By encoding the interaction graph and KG separately, KACL can better capture user preference and item knowledge information; (2) By further performing knowledge-adaptive contrastive learning on two graphs, KACL can better encode the information shared across the two views into item representations. (3) Benefiting from the regularization to KG, KACL can effectively collect more informative signals from the entities and relations.

Table 2: Experimental results of recall@20 and ndcg@20 in top-K recommendation.

Model	Amazon-Book		LastFM		Movielens	
	recall	ndcg	recall	ndcg	recall	ndcg
BPR-MF	0.1321	0.0682	0.0715	0.0637	0.4052	0.2609
CKE	0.1352	0.0699	0.0746	0.0652	0.4106	0.2669
KGCN	0.1464	0.0769	0.0819	0.0705	0.4237	0.2753
KGNNLS	0.1448	0.0759	0.0806	0.0695	0.4218	0.2741
KGAT	0.1507	0.0802	0.0877	0.0749	0.4532	0.3007
CKAN	0.1467	0.0702	0.0812	0.0690	0.4314	0.2891
KGPL	0.1503	0.0785	0.0896	0.0751	0.4417	0.2864
DSKReG	0.1551	0.0863	0.0924	0.0816	0.4589	0.3017
KGIN	0.1631	0.0881	0.0967	0.0847	0.4661	0.3120
CKER	0.1619	0.0863	0.0951	0.0832	-	-
KGCL	0.1569	0.0833	0.0899	0.0793	0.4516	0.2967
KACL	**0.1657**	**0.0915**	**0.1133**	**0.0989**	**0.4752**	**0.3278**
%Improv	1.35%	3.86%	17.18%	16.77%	1.95%	5.06%

• Jointly analyzing KACL across the three datasets, we find that the improvements on LastFM are more significant than those on Amazon-Book and Movielens. The relative improvements of Recall@20 on both Amazon-Book and Movielens are under 2%. This might be caused by the characteristics of datasets: (1) both interaction and KG data on LastFM offer denser and richer information than that on Amazon-Book and Movielens; and (2) in Amazon-Book and Movielens, the item context information contained in the knowledge graph is similar to that in the interaction graph (*i.e.*, there is little supplementary information in KG). Specifically, for each item, we compare the similarity of the second-order item neighbors from knowledge graph and interaction graph, and then we find that the similarity score of Movielens is the highest and very close to Amazon-Book, and LastFM is the lowest. This indirectly indicates that KACL is good at capturing KG information.

• Compared with BPR-MF, the results of KG-based methods (*e.g.*, CKE, KGCN, KGAT, KGIN) verify the importance of knowledge graph. By simply incorporating KG embeddings into MF, CKE performs better than BPR-MF. However, since TransR in CKE only captures the first-order neighbors in KG and ignores long-range connectivity, GNN-based methods substantially outperform CKE in most cases. Also, as the state-of-the-art paradigm for KG-based recommendation, CKG-based methods (*i.e.*, KGAT, KGIN, DSKReG) perform better than KGCN, KGNNLS, and CKAN. The most possible reason is that KGCN-based methods ignore the high-order context in user-item interaction graph.

• Compared with CKER and KGCL which design CL to incorporate the KG information to enrich node representations, the improvement of KACL mostly comes from our design of the knowledge-adaptive contrastive task, which employs adaptive data augmentation to filter out recommendation-irrelevant noises, and relation-aware graph encoder to fully utilize the structure information.

5.3 Performance under Different Sparsity

Since a key motivation of KG-based model is to alleviate data sparsity and cold start problem, we further investigate the performance of these methods for users and items with different sparsity level.

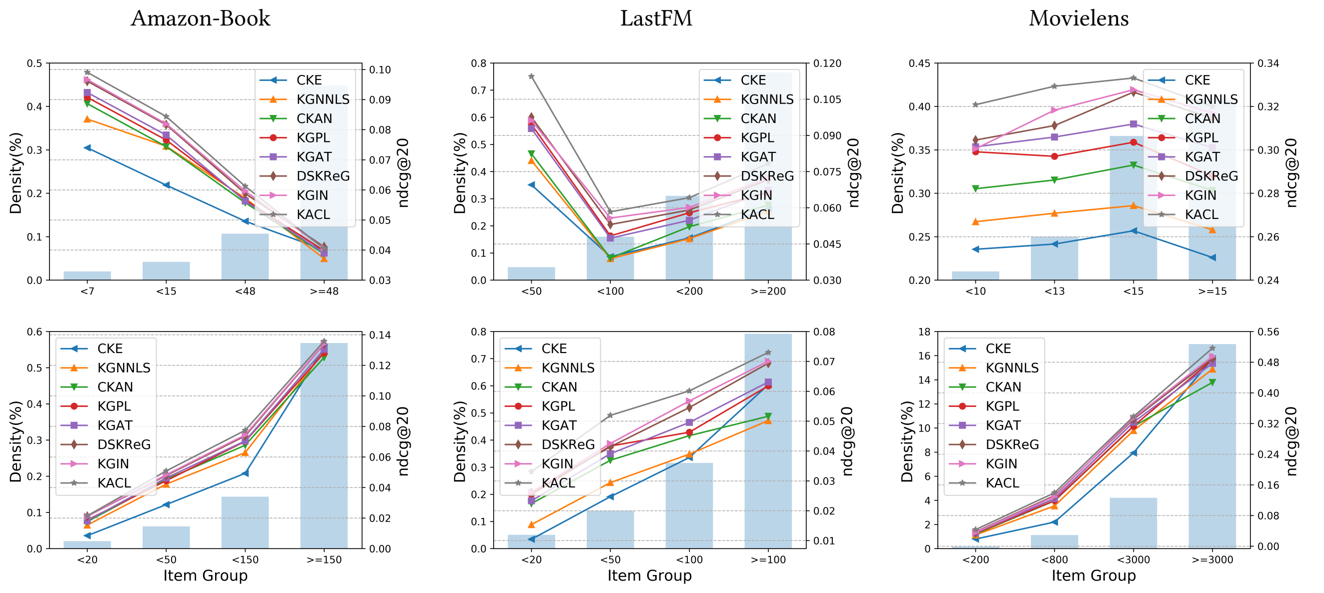


Figure 3: Performance comparison over user/item groups with different sparsity. The background histograms indicate the density of each group. Each line presents the ndcg@20 performance of the corresponding method. Due to space limitation, we omit the results of KGCN which have a similar trend to that of KGNNLS. Best viewed in color.

Table 3: Comparison of KACL and its ablated variants. The improvements of KACL over all variants are significant (0.05 level paired t-test).

Model	Amazon-Book		LastFM		Movielens	
	recall	ndcg	recall	ndcg	recall	ndcg
w/o CL	**0.1563**	**0.0802**	**0.0912**	**0.0808**	**0.4614**	**0.3094**
w/o Ada	**0.1614**	**0.0883**	**0.1091**	**0.0946**	**0.4719**	**0.3227**
w/o KG	**0.1584**	**0.0827**	**0.1026**	**0.0912**	**0.4695**	**0.3196**
KACL	**0.1657***	**0.0915***	**0.1133***	**0.0989***	**0.4752***	**0.3278***

5.3.1 ***User-side Performance Analysis.*** Following the setting of KGAT [25], we construct four groups of users with different sparsity levels according to the number of observed interactions for each user in the training set, and let different groups have similar total interactions. Then we divide the testing data into four subsets by user groups. Figure 3 presents the results of ndcg@20 metric on different user groups in Amazon-Book, Last-FM, and Movielens. We can observe that our KACL achieves consistently higher ndcg@20 than other methods for each user group, especially on the sparsest user groups in all datasets. It demonstrates that our proposed model performs well on cold start users by introducing KG information into collaborative filtering. Compared with sparse user groups, KACL slightly outperforms baselines in the most right user groups, especially in the Amazon-Book dataset. One possible reason is that users can obtain enough information from rich interactions, and knowledge graph can not provide additional useful information.

5.3.2 ***Item-side Performance Analysis.*** To further validate the improvement of KACL on cold-start items, we use a similar strategy to the user side and then get four item groups for each dataset. As shown in Figure 3, we can observe that the performance of our model and baselines has a significant decline in the sparse item groups due to the challenge of learning representations for inactive items. Compared with baselines, KACL shows significant relative improvements in all groups, especially for the sparest group. Hence, our knowledge-adaptive contrastive learning is beneficial to guide the representation learning of items with sparse behaviors.

5.4 Ablation Analysis

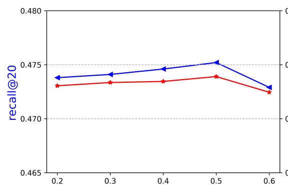
To study the impact of our key components, we consider different model variants of KACL from three perspectives and analyze their effects. The KACL w/o CL removes knowledge-adaptive contrastive learning module to evaluate the benefits from contrastive learning. To examine the effect of the adaptive view generators, we only retain graph corruption in data augmentation procedure, termed KACL w/o Ada. Moreover, the KACL w/o KG disables the auxiliary KG regularization to verify the necessity of KG embedding.

From the results of the ablation study in Table 3, we make the following observations: (1) The full version of our KACL consistently achieves the best performance, which demonstrates that each component of KACL will contribute to the performance. (2) Performing multi-task learning with extra contrastive task achieves better performance than recommendation task alone. (3) The adaptive data augmentation is necessary in recommendation scenario, which can effectively drop out recommendation-irrelevant information and alleviate knowledge overload problem. (4) The regularization of KG embedding is beneficial in capturing the KG structure.

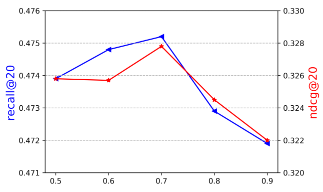
5.5 Influence of Hyper-parameters

We present the hyper-parameter study results of KACL in Figure 4. Due to the space limitation, we omit the results on Amazon-Book and LastFM which have a similar trend to that on Movielens.

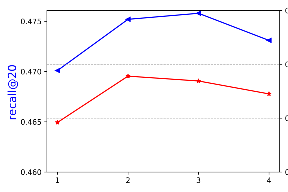
• **Effect of temperatures τ_k and τ_b** To evaluate the influence of τ_k, we vary its value from 0.2 to 0.6 and draw both recall@20



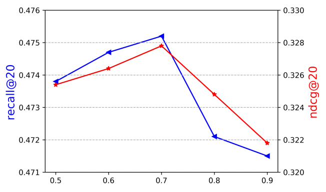
(a) Temperature τ_k



(b) Temperature τ_{cl}



(c) Layer size



(d) Dropout rate ρ

Figure 4: Hyper-parameter sensitivity analysis on Movielens.

Table 4: Top-5 relations with the lowest/highest learned dropping ratios on Amazon-Book dataset.

Id	Relation	Dropping ratio	Triplets
1	object_type	$< 10^{-4}$	666,473
8	book_subject	$< 10^{-4}$	39,180
9	literary_genre	$< 10^{-4}$	50,388
14	author	$< 10^{-4}$	37,255
17	book_character	$< 10^{-4}$	10,182
2	copyright_date	0.853	11,657
7	notable_types	0.852	106,637
6	date_of_first_publication	0.810	17,298
11	original_language	0.797	13,916
12	is_reviewed	0.601	4,752

and ndcg@20 scores to show the influence. In Figure 4(a), we can find the model has the best performance in 0.5. For τ_b, the change under different values is even less than that of τ_k, and thus we omit the lines of τ_b to avoid overlapping.

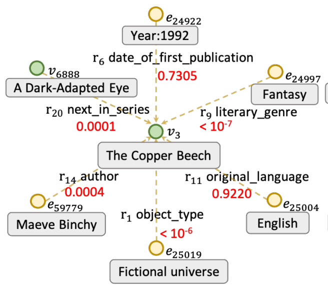
• **Effect of temperature** τ_{cl} In Figure 4(b), we draw ndcg@20 score to show the effect of the value of τ_{cl}. KACL performs better when the value increases from 0.5 to 0.7, and then decreases quickly as the value increases.

• **Effect of layer number** L **of both** GNN_{v_1} **and** GNN_{v_2} Figure 4(c) shows the ndcg@20 results by stacking GNN layer from 1 to 4. We can observe that the best performances are achieved when L is 2 and the results start to decline when we further increase the number of layers because of over-smoothing.

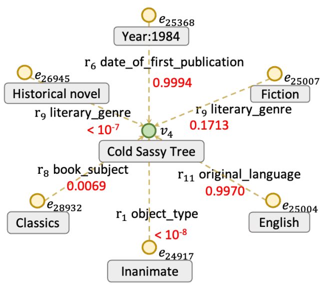
• **Effect of edge dropout rate** ρ To explore the impact of edge dropout rate, we experiment KACL under different rates while keeping other hyper-parameters unchanged. The comparison results show that KACL with $\rho = 0.3$ performs better.

5.6 Study on Effectiveness and Explainability

To validate the effectiveness of KACL on alleviating knowledge overload, we analyze the learned edge dropping probability in the adaptive knowledge view generator on Amazon-Book, *i.e.,* the



(a) Book "The Copper Beech"



(b) Book "Cold Sassy Tree"

Figure 5: Case study of learned edge dropping probabilities on Amazon-Book dataset.

dataset with the largest number of relations. First, we compute the average edge dropping ratio of each relation, and show the top-5 relations with the lowest or highest ratios in Table 4. Intuitively, the upper five relations are more useful for recommending books to users, while the lower five provide little information relevant to the recommendation. This observation shows the ability of our adaptive data augmentation in alleviating noises. To further prove that the relations with the highest learned dropping ratios are truly task-irrelevant, we construct two subsets of KG with similar numbers of total triplets: relation sets $(8, 9, 14, 19)$ and $(2, 6, 7, 11)$, and respectively train two KACL models by replacing the entire KG with the two subsets. The recall@20 are respectively 0.1612 and 0.1559, which demonstrates the denoising ability of KACL in relation-level. Besides, we present the case study of book ids 3 and 4 on Amazon dataset in Figure 5, and the results can also match our intuition.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose a novel and compact model KACL for KG-based recommendation, which goes beyond the limitations of the typical CKG-based paradigm and employs a new paradigm based on contrastive learning. We employ a multi-task learning framework which can supplement the classical recommendation loss with extra contrastive learning between user-item interaction graph view and knowledge graph view. The contrastive learning module can extract the information shared by both views, and thus alleviates the interaction domination and task-irrelevant noises. Adaptive view generators are also proposed to help remove edges unrelated to recommendation in data augmentation. Experimental results validate the advantages of our KACL over all state-of-the-art models, and demonstrate the effectiveness of each proposed component.

For future work, we will explore the migration of KACL to other recommendation scenarios suffering from heavy noises. We can also investigate other graph augmentation techniques for recommendation task as well, such as pseudo-labeling or self-training.

ACKNOWLEDGMENTS

This work is supported in part by the National Natural Science Foundation of China (No. U20B2045, 62192784, 62172052, 62002029, 62006129).

REFERENCES

[1] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems* 26 (2013).

[2] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *WWW*. 151–161.

[3] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*. 191–198.

[4] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 297–304.

[5] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*. 639–648.

[6] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *WWW*. 173–182.

[7] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S Yu. 2018. 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*. 1531–1540.

[8] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR abs/1412.6980* (2014).

[9] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *ICLR* (2016).

[10] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[11] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*.

[12] Dongsheng Luo, Wei Cheng, Dongkuan Xu, Wenchao Yu, Bo Zong, Haifeng Chen, and Xiang Zhang. 2020. Parameterized Explainer for Graph Neural Network. *Advances in Neural Information Processing Systems* 33 (2020).

[13] Qingsong Lv, Ming Ding, Qiang Liu, Yuxiang Chen, Wenzheng Feng, Siming He, Chang Zhou, Jianguo Jiang, Yuxiao Dong, and Jie Tang. 2021. Are we really making much progress? Revisiting, benchmarking, and refining heterogeneous graph neural networks. (2021).

[14] Chen Ma, Liheng Ma, Yingxue Zhang, Haolun Wu, Xue Liu, and Mark Coates. 2021. Knowledge-enhanced top-k recommendation in poincaré ball. *arXiv preprint arXiv:2101.04852* (2021).

[15] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. arXiv:1611.00712 [cs.LG]

[16] Zhiqiang Pan and Honghui Chen. 2021. Collaborative Knowledge-Enhanced Recommendation with Self-Supervisions. *Mathematics* 9, 17 (2021), 2129.

[17] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618* (2012).

[18] Riku Togashi, Mayu Otani, and Shin'ichi Satoh. 2021. Alleviating Cold-Start Problems in Recommendation through Pseudo-Labelling over Knowledge Graph. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*. 931–939.

[19] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *ICLR* 2 (2018).

[20] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 417–426.

[21] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. DKN: Deep knowledge-aware network for news recommendation. In *WWW*. 1835–1844.

[22] Hongwei Wang, Fuzheng Zhang, Mengdi Zhang, Jure Leskovec, Miao Zhao, Wenjie Li, and Zhongyuan Wang. 2019. 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*. 968–977.

[23] Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge Graph Convolutional Networks for Recommender Systems. In *WWW*. 3307–3313.

[24] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale commodity embedding for e-commerce recommendation in alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 839–848.

[25] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 950–958.

[26] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *SIGIR*. 165–174.

[27] Xiang Wang, Tinglin Huang, Dingxian Wang, Yancheng Yuan, Zhenguang Liu, Xiangnan He, and Tat-Seng Chua. 2021. Learning Intents behind Interactions with Knowledge Graph for Recommendation. In *Proceedings of the Web Conference 2021*. 878–887.

[28] Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. 2019. Explainable reasoning over knowledge graphs for recommendation. In *AAAI*, Vol. 33. 5329–5336.

[29] Xiang Wang, Yaokun Xu, Xiangnan He, Yixin Cao, Meng Wang, and Tat-Seng Chua. 2020. Reinforced negative sampling over knowledge graph for recommendation. In *Proceedings of The Web Conference 2020*. 99–109.

[30] Yu Wang, Zhiwei Liu, Ziwei Fan, Lichao Sun, and Philip S Yu. 2021. Dskreg: Differentiable sampling on knowledge graph for recommendation with relational gnn. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 3513–3517.

[31] Ze Wang, Guangyan Lin, Huobin Tan, Qinghong Chen, and Xiyang Liu. 2020. CKAN: Collaborative Knowledge-aware Attentive Network for Recommender Systems. In *SIGIR*. 219–228.

[32] Markus Weimer, Alexandros Karatzoglou, and Alex Smola. 2008. Adaptive collaborative filtering. In *Proceedings of the 2008 ACM conference on Recommender systems*. 275–282.

[33] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *SIGIR*. 726–735.

[34] Yiqing Wu, Ruobing Xie, Yongchun Zhu, Xiang Ao, Xin Chen, Xu Zhang, Fuzhen Zhuang, Leyu Lin, and Qing He. 2022. Multi-view Multi-behavior Contrastive Learning in Recommendation. *arXiv preprint arXiv:2203.10576* (2022).

[35] Lianghao Xia, Chao Huang, Yong Xu, Peng Dai, Xiyue Zhang, Hongsheng Yang, Jian Pei, and Liefeng Bo. 2021. Knowledge-Enhanced Hierarchical Graph Transformer Network for Multi-Behavior Recommendation. In *AAAI*, Vol. 35. 4486–4493.

[36] Xin Xia, Hongzhi Yin, Junliang Yu, Yingxia Shao, and Lizhen Cui. 2021. Self-Supervised Graph Co-Training for Session-based Recommendation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 2180–2190.

[37] Keyulu Xu, Chengtao Li, Yonglong Tian, Tomohiro Sonobe, Ken-ichi Kawarabayashi, and Stefanie Jegelka. 2018. Representation learning on graphs with jumping knowledge networks. In *International Conference on Machine Learning*. PMLR, 5453–5462.

[38] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575* (2014).

[39] Yuhao Yang, Chao Huang, Lianghao Xia, and Chenliang Li. 2022. Knowledge Graph Contrastive Learning for Recommendation. *arXiv preprint arXiv:2205.00976* (2022).

[40] Junliang Yu, Hongzhi Yin, Min Gao, Xin Xia, Xiangliang Zhang, and Nguyen Quoc Viet Hung. 2021. Socially-Aware Self-Supervised Tri-Training for Recommendation. *arXiv preprint arXiv:2106.03569* (2021).

[41] Junliang Yu, Hongzhi Yin, Jundong Li, Qinyong Wang, Nguyen Quoc Viet Hung, and Xiangliang Zhang. 2021. Self-supervised multi-channel hypergraph convolutional network for social recommendation. In *Proceedings of the Web Conference 2021*. 413–424.

[42] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Quoc Viet Hung Nguyen. 2022. Are Graph Augmentations Necessary? Simple Graph Contrastive Learning for Recommendation. In *SIGIR*. 1294–1303.

[43] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. 2016. Collaborative knowledge base embedding for recommender systems. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 353–362.

[44] Guanjie Zheng, Fuzheng Zhang, Zihan Zheng, Yang Xiang, Nicholas Jing Yuan, Xing Xie, and Zhenhui Li. 2018. DRN: A deep reinforcement learning framework for news recommendation. In *WWW*. 167–176.

[45] Ding Zou, Wei Wei, Xian-Ling Mao, Ziyang Wang, Minghui Qiu, Feida Zhu, and Xin Cao. 2022. Multi-level Cross-view Contrastive Learning for Knowledge-aware Recommender System. *arXiv preprint arXiv:2204.08807* (2022).

[46] Ding Zou, Wei Wei, Ziyang Wang, Xian-Ling Mao, Feida Zhu, Rui Fang, and Dangyang Chen. 2022. Improving Knowledge-aware Recommendation with Multi-level Interactive Contrastive Learning. In *CIKM*.