Graph Contrastive Learning with Knowledge Transfer for Recommendation

Baoxin Zhang, Dan Yang, Yang Liu, Yu Zhang

Abstract—**The graph collaborative filtering algorithm, known for its excellent data modeling capability, has found wide applications in recommendation systems. However, traditional graph collaborative filtering algorithms often face challenges related to sparse interaction data and inefficient utilization of side information between users and items. To address these issues, a graph Contrastive Learning with Knowledge Transfer (CLKT) has been proposed. The proposed method aims to integrate the side information of user and item attributes with interaction data using GNN. By leveraging contrastive learning, it achieves cross-view learning within the graph collaborative filtering algorithm. Recognizing that the impact of side information on user-item interaction data can vary, the proposed method introduces feature transformers and knowledge transfer networks into contrastive learning. This enables the adaptive embedding of node features for users and items in CLKT. To further enhance the embedding capability for different node relationships, the CLKT method incorporates cluster-based contrastive learning. This approach considers the diverse impacts that different types of node relationships can have on node feature learning. By simultaneously learning both the interaction relationships between nodes and the clustering relationships among nodes, the proposed method enables the exploration of multiple-node relationships for feature mining. Extensive experiments on three real datasets demonstrate the effectiveness and feasibility of the proposed CLKT, and the results demonstrated its promising performance.**

Index Terms—**Recommendation Algorithm, Self-Supervised Learning, Graph Neural Network, Contrastive Learning**

I. Introduction

THE graph collaborative filtering algorithm [1-3] has achieved significant success through its excellent ability to mine users' historical interaction information. However, most existing research primarily focuses on the interaction information between users and items. In the real world, there is still a wealth of side information available, such as social connections between users or associative information between semantically similar items. These side information can provide a more comprehensive feature learning capability for graph collaborative filtering algorithms. As a result, there is a growing trend in collaborative filtering research toward incorporating side information. Existing recommendation algorithms that focus on side information [4-6] can extract potentially interesting items for users based on the available side information between users and items. However, many of the current side-based recommendation algorithms are often affected by the sparsity of side information data. This sparsity issue limits the ability of recommendation algorithms to effectively learn the node features of users and items. Recently, some studies have utilized structural variations to mitigate the problem of sparse interaction data. Techniques such as sampling strategies [7] or subgraph modifications [8] have been employed. However, these approaches often introduce unnecessary noise to the node embedding process. On the other hand, self-supervised learning addresses the issue of sparse data in recommendation algorithms while avoiding modifications to the graph structure, thereby alleviating the impact of noise generation. Self-supervised learning is commonly integrated into recommendation algorithms through contrastive learning [9-11]. Contrastive learning is an effective algorithm for alleviating data sparsity. It can extract significant features of users and items even in the absence of labels. By enhancing the consistency between the current node and positive examples while reducing the association between negative examples.

Existing recommendation algorithms that incorporate contrastive learning are mostly focused on user-item bipartite graphs, with limited research exploring the side information between users and items. In the real world, users are not only connected to items through interactions but are also influenced by their side information when making choices. Considering the side information of users and items can result in more comprehensive feature learning compared to solely focusing on user-item interaction information. The main challenges faced by side information-based graph contrastive learning recommendation algorithms are as follows:

- **How to effectively integrate the side information of users and items with user-item interaction information.** For example, if two users have similar project interests, the traditional collaborative filtering approach would recommend the interests of one user to the other. However, considering the associated relationships between the two users in the side information can enhance the accuracy of the recommendation algorithm.
- **How to incorporate the impact of side information on the user-item interaction into node embeddings.** For

Manuscript received June 21, 2023; revised December 28, 2023. This work was supported by the General Scientific Research Project of Liaoning Provincial Department of Education (LJKMZ20220646).

Baoxin Zhang is a postgraduate student at School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, China (e-mail: ustlzhangbaoxin@163.com).

Dan Yang is a professor at School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, China (e-mail: asyangdan@163. com).

Yang Liu is an associate professor at School of Computer Science and Software Engineering, University of Science and Technology Liaoning, Anshan, China (corresponding author to provide e-mail: liuyang_lnas@163.com).

Yu Zhang is a senior engineer at China Telecom Digital Intelligence Technology Co, Ltd, Beijing, China (e-mail: zhangy193@chinatelecom.cn).

example, when a user purchases a product, the influence of being recommended by a friend is likely significant, while the influence of the product's similarity to other items is relatively minor.

- **How to leverage other types of relationships between users and items.** Based on existing user-item interaction information, to influence node embeddings. For example, there no direct interaction between two users, but their behaviors are similar. By applying clustering algorithms, it is possible to find other types of relationships between these two users.

To address the aforementioned challenges, we propose a graph Contrastive Learning with Knowledge Transfer (CLKT). This method utilizes Graph Neural Networks(GNN) as encoders and proposes two contrastive learning formulations to enhance node embedding capabilities based on the obtained user and item features during the encoding process. In terms of integrating side information and user-item interaction features, it combines feature transformers and knowledge transfer networks with side information-based contrastive learning. By addressing the issue of data sparsity, it performs feature embedding for users and items, facilitating the fusion of features across multiple views. In terms of node relationships where interaction connections are not available, CLKT combines clustering algorithms with clustering contrastive learning formulations to achieve feature embedding from clustering perspective. CLKT takes into account the side information of users and items, it also explores the relationship features among non-interacting nodes.

The main contributions of this paper are as follows:

- We propose a contrastive learning formulation that focuses on the side information of users and items, integrating the side information features of users and items into the recommendation process.
- We combine feature transformers and knowledge transfer networks with contrastive learning to enable CLKT to adaptively extract more important feature information.
- We propose a cluster-based contrastive learning formulation that integrates the effects of user and item clustering relationships on feature learning.
- We conducted experiments on three public datasets, and the results confirmed that CLKT outperforms existing state-of-the-art baseline algorithms in terms of performance.

II. Related Work

A. Graph Collaborative Filtering

Traditional collaborative filtering algorithms predict potential items of interest for users based on their past interaction data. Graph collaborative filtering algorithms integrate user-item interactions into an interaction graph and utilize GNN to learn node features, enabling predictions of users' future interaction behaviors. Early collaborative filtering algorithms typically employed random sampling to extract feature information from interaction data. Subsequently, the emergence of graph collaborative filtering algorithms introduced GNN into the field of collaborative filtering.

In related work [12], the NGCF model was proposed, which utilizes neighborhood aggregation algorithms in GNN to integrate neighbor features of user and item nodes. This model introduced a new research direction by incorporating collaborative filtering into the framework of graph-based learning. In related work [13], the LightGCN algorithm is proposed, a simplified graph convolutional algorithm. It removes unnecessary operations such as feature transformation and focuses on capturing high-order relations between the interaction graphs using graph convolutional operations. This approach reduces time complexity while improving the performance of the recommendation algorithm. In related work [14], an approach is proposed to construct multiple interaction graphs from the interaction data to capture richer interaction features. Although this approach improves performance, it does not consider the side information of users and items. Furthermore, this algorithm introduces a significant amount of noise and increases the time complexity, resulting in resource wastage. Related work [15] adopted the SGL model, which incorporates self-supervised learning into graph collaborative filtering algorithms. Combining random data augmentation operations in graph convolution layers with contrastive learning, SGL enhances the robustness and accuracy of graph collaborative filtering. Related work [16] adopted the NCL model, which combines contrastive learning of two types of node relationships with GNN. It alleviates the problem of data sparsity while enhancing the feature embedding capability of GNN. However, these algorithms mainly focus on user-item interaction data and overlook the role of user-side information and item-side information.

B. Contrastive Learning

Contrastive learning, as a form of self-supervised learning, is commonly used to handle data sparsity issues. As a result, there has been a recent surge in research on recommender systems that incorporate contrastive learning techniques. Related work [17] employs contrastive learning to maximize the consistency between item content and collaborative signals, thereby mitigating the issue of data sparsity. Related work [18] enhances feature learning of nodes by employing a two-tower DNN to augment the data. It leverages data augmentation and contrastive learning techniques to strengthen the node feature learning process. Related work [15] introduces the SGL model, which employs random node and edge removal as well as random walks in the convolutional layers for data augmentation. It then utilizes contrastive learning for node embedding. In addition to using deletion-based data augmentation, related work [19] proposes a reordering-based data augmentation method for sequential data, specifically for enhancing sequence recommendation tasks. In addition to addressing the issue of data sparsity, related work [20] contributes to resolving the exposure bias problem using contrastive learning. These contrastive learning algorithms have improved the performance of the algorithms, but they have not considered the side information of users and items, nor have they considered the impact of other types of node relationships between users and items, apart from their interaction.

III. Preliminaries

In this section, we utilize three types of graphs, namely the user-item interaction graph G_{ui}, the user social graph G_{uu}, and the item similarity graph G_{ii}, to represent the association information between users and items.

Definition 1. User-item interaction matrix A_{ui}. The representation of the user-item interaction graph is $G_{ui} = (V_u, V_i, E_{ui})$, The set representing the users is $V_u = \{u_1, u_2, \cdots, u_m\}$, The set representing the items is $V_i = \{i_1, i_2, \cdots, i_n\}$. E_{ui} represents the set of edges between users and items. In the interaction data, when user *u* interacts with item *i*, there exists an edge in E_{ui} that connects user *u* and item *i*. This paper defines the adjacency matrix $A_{ui} \in R^{m*n}$ for the user-item interaction graph G_{ui}, Where m represents the number of users and n represents the number of items.

Definition 2. User side information matrix A_{uu}. The paper defines the graph $G_{uu} = (V_u, E_{uu})$ to represent user side information, where E_{uu} represents the set of user side information. The adjacency matrix of the G_{uu} is denoted as $A_{uu} \in R^{m*m}$.

Definition 3. Item side information matrix A_{ii}. The paper defines the graph $G_{ii} = (V_i, E_{ii})$ to represent item side information, where E_{ii} represents the set of item side information. The adjacency matrix of the G_{ii} is denoted as $A_{ii} \in R^{n*n}$.

IV. The Proposed Framework

The overall framework of the CLKT recommendation method is shown in Fig. 1. The CLKT method enhances its adaptability by integrating user-item interaction information and side information through graph convolutional networks (GCN). It further incorporates feature transformers and knowledge transfer networks into the contrastive learning framework. This allows CLKT to effectively fuse features, learn from contrasting samples, and improve its ability to adapt to different scenarios. In terms of node relationships, CLKT incorporates contrastive learning based on clustering, allowing it to embed features from multiple types of node relationships. Finally, CLKT can predict users' preferences for items and generate a Top-*N* recommendation list.

A. Node Embedding Initialization

CLKT adopts Xavier initialization to initialize the embeddings of users and items based on their corresponding *id*, resulting in the initial embedding matrices e_u^0 and e_i^0 for the nodes. We introduce a self-gating module to derive user-based side information embeddings and item-based side information embeddings from a shared initial embedding space, which are shown as follows:

$$e_{uu}^0 = e_u^0 \times \sigma(e_u^0 W + b) \tag{1}$$

$$e_{ii}^0 = e_i^0 \times \sigma(e_i^0 W + b) \tag{2}$$

Where $e_{uu}^0 \in R^{m*d}$ and $e_{ii}^0 \in R^{n*d}$ represent the embeddings of the user side information graph G_{uu} and the item side information graph G_{ii}, respectively. × is a matrix multiplication operation, σ is an activation function, *W* represents the transformation parameters, and *b* represents the bias parameters. The self-gating mechanism enables e_{uu}^0 and

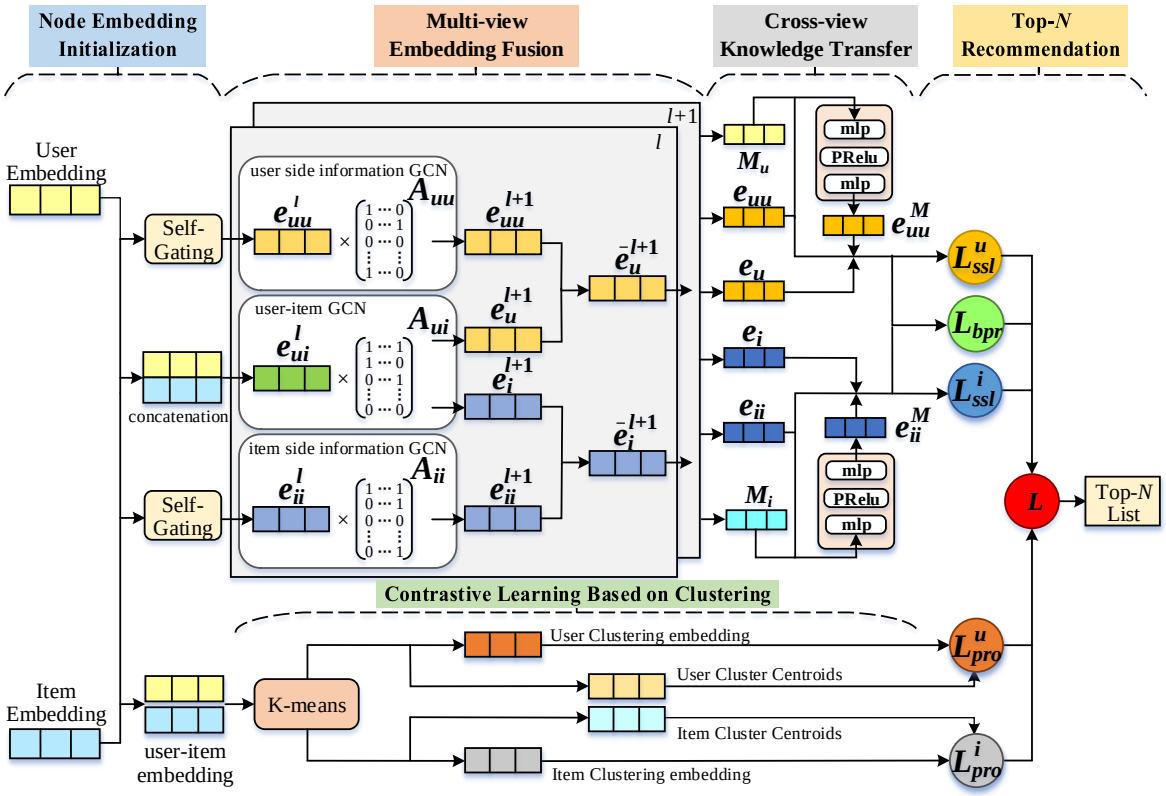

Fig. 1. Overall Framework of the CLKT

e_{ii}^0 to not only share semantic information with e_u^0 and e_i^0 but also capture the feature attributes among users and items.

B. Multi-view Embedding Fusion

1) Message Propagation based on Graph Convolutional Networks

In the initial embedding matrix, e_u^0 and e_i^0 serve as inputs for the user-item interaction view, while e_{uu}^0 and e_{ii}^0 serve as inputs for the user side information view and the item side information view, respectively. CLKT utilizes GCN as encoders for the three views. Inspired by LightGCN [13], CLKT removes feature transformation and activation functions during the message propagation process. During the modeling process of the user-item interaction view, the embeddings of users and items are iteratively updated through message propagation using the user-item interaction view. The specific formulas for message propagation are as follows:

$$e_u^{l+1} = \sum_{i \in Nu} \frac{1}{\sqrt{|N_u|}\sqrt{|N_i|}} e_i^l \tag{3}$$

$$e_i^{l+1} = \sum_{u \in Ni} \frac{1}{\sqrt{|N_i|}\sqrt{|N_u|}} e_u^l \tag{4}$$

Where N_i and N_u represent the neighbors of user u and item i, respectively. e_u^{l+1} and e_i^{l+1} represent the l+1 layer iterative embedding vectors of user u and item i, respectively. Similarly, the user-user embedding generated from the user's side information matrix and the item-item embedding generated from the item's side information matrix is iteratively propagated using the same propagation method.

2) Information Aggregation based on Graph Convolutional Networks

In CLKT, the information for each iteration is aggregated from the user-item interaction information and side information. CLKT undergoes multiple layers of message propagation iterations to obtain high-order embeddings, and these high-order embeddings preserve the relational information between nodes through multi-hop connections. The specific process of integrating user and item embeddings are as follows:

$$\bar{e}_u^{l+1} = f\,(e_u^{l+1}, e_{uu}^{l+1}) \tag{5}$$

$$\bar{e}_i^{l+1} = f\,(e_i^{l+1}, e_{ii}^{l+1}) \tag{6}$$

Where $\bar{e}_u^{l+1} \in R^{m*d}$ and $\bar{e}_i^{l+1} \in R^{n*d}$ are combined to incorporate the user's side information and the item's side information, serving as the input data for the next layer. The fusion function $f()$ represents the merging of embedding information, and CLKT utilizes the average pooling function as the fusion function.

To integrate the high-order information between nodes, we designed embedding aggregation formulas:

$$e_u = e_u^0 + \sum_{l=1}^{L} \frac{e_u^l}{\left\| e_u^l \right\|} \tag{7}$$

$$e_i = e_i^0 + \sum_{l=1}^{L} \frac{e_i^l}{\left\| e_i^l \right\|} \tag{8}$$

Where L represents the total number of iterations, the embeddings from each layer are normalized and concatenated with the initial embeddings. The high-order information aggregation is performed in the same manner for the user's side information embedding e_{uu} and the item's side information embedding e_{ii}.

C. Cross-view Knowledge Transfer

CLKT integrates user side information and item side information to assist the graph collaborative filtering algorithm in the recommendation. In the real world, the weight of user side information and item side information may vary for different user-item interaction relationships. For example, some users prefer to discover related items based on their interests, while others prefer to select items recommended by their friends. Therefore, to enable CLKT to learn more accurate feature attributes, we have designed feature transformers and cross-view knowledge transfer networks specifically for user nodes and item nodes.

1) Feature Transformer

To associate the side information of users and items with the interaction information between users and items, we extract side information features of users and items along with the interaction information features between users and items, and then integrate the extracted features. The formulas for extracting features of users and items are as follows:

$$M_u = f_{mlp}(concat\,(e_u, e_{uu}, \sum_{i \in Nu} e_i)) \tag{9}$$

$$M_i = f_{mlp}(concat\,(e_i, e_{ii}, \sum_{u \in Ni} e_u)) \tag{10}$$

Where $M_u \in R^{m*3d}$ and $M_i \in R^{n*3d}$ represent the global features obtained by integrating the side information features and interaction features. Global features involve the side information embeddings of users e_{uu} and items e_{ii}, along with the embeddings of users e_u and items e_i, as well as the aggregated information from the node neighborhoods. In the features of user-item interaction information, we identify users' interests in relevant items. By incorporating the side information of users and items, we enhance the personalized embeddings of nodes, enabling the global feature information to effectively associate important and relevant contextual information from the side information and user-item interaction information.

2) Knowledge Transfer Based on Global Features

We feed the extracted global features into the feature transformer, which generates a parameter matrix based on these global features. The specific formulas are as follows:

$$W_u^{M1} = f_m^1(M_u) \tag{11}$$

$$W_u^{M2} = f_m^2(M_u) \tag{12}$$

Where f_m^1 and f_m^2 respectively represent the feature transformer, which consists of fully connected layers and PReLU activation functions. These feature transformers take the global feature matrix M_u as input and generate the globalized feature parameter matrices W_u^{M1} and W_u^{M2}. These parameter matrices consist of m matrices,

corresponding to *m* users, and are generated based on the features of the respective user and item, enabling personalized knowledge transfer. The rank of these two sets of matrices is constrained to $k < d$, which not only reduces the number of trainable parameters in the feature transformers but also enhances the stability of CLKT. In this paper, the generated global feature parameter matrices and a non-linear mapping function are utilized to construct the knowledge transfer network. The specific formula is as follows:

$$e_u^M = \sigma(W_u^{M1} W_u^{M2} e_{uu}) \tag{13}$$

Where $\sigma()$ denotes the PReLU activation function. $e_u^M \in R^{m*d}$ represents the embedding of global features that incorporate both user side information and user-item interaction information. It extracts important features between user side information and user-item interaction information, filters out noise information from user side information to user-item interaction information, and thereby enhances the recommendation effectiveness of CLKT. We utilize embedding e_u^M to enhance the user embeddings generated from the user-item interaction information. The specific formula for user embedding fusion is as follows:

$$e_u^f = \alpha_u * e_u + (1 - \alpha_u) * (e_{uu} + e_u^M) \tag{14}$$

Where α_u represents the weight parameters that control the balance between the user embeddings from the user-item interaction information and the user embeddings from the user's side information. This leads to the generation of the final embedding e_u^f, which is used for recommendation. The item embeddings are in the same manner as described above.

3) Contrastive Learning Based on Side Information

To further leverage the self-supervised learning property for enhancing the feature learning of users and items, and mitigating the impact of data sparsity, we devised a contrastive learning formulation based on side information. Because knowledge transfer based on global features can identify different user preferences, we combine it with contrastive learning. We generate embeddings(e_u^M , e_i^M) containing important features from side information by applying feature transformers and knowledge transfer networks to the embeddings(e_u , e_i) of user-item interaction information.

Inspired by the application of contrastive learning in recommender systems, we propose a contrastive learning formulation based on InfoNCE to enhance the feature learning between nodes. The specific formula for the contrastive learning loss function of users is as follows:

$$L_{ssl}^u = \sum_{u \in Vu} -\log \frac{exp(s(e_u^M + e_{uu}, e_u)/\tau)}{\sum_{u' \in Vu} exp(s(e_u^M + e_{uu}, e_{u'})/\tau)} \tag{15}$$

Where *s*() represents the similarity function, CLKT uses cosine similarity for experimental purposes. $e_u^M, e_u \in R^d$ represent the embedded vectors that contain side information and interaction information, respectively. Where τ represents the temperature coefficient, which helps identify challenging negative samples, and u' represents negative samples from different indicators.

Similarly, we propose the contrastive learning loss function for items as follows:

$$L_{ssl}^i = \sum_{i \in Vi} -\log \frac{exp(s(e_i^M + e_{ii}, e_i)/\tau)}{\sum_{i' \in Vi} exp(s(e_i^M + e_{ii}, e_{i'})/\tau)} \tag{16}$$

The final contrastive learning loss function is the sum of the two aforementioned loss functions:

$$L_{ssl} = \alpha * L_{ssl}^u + L_{ssl}^i \tag{17}$$

Where α is a parameter used to control the weights of users and items.

D. Contrastive Learning Based on Clustering

To incorporate node clustering into contrastive learning, we introduce clustering algorithms. By considering node embedding from a clustering perspective, we can uncover node features that lack interaction information but exhibit similar attribute information. For user node contrastive learning based on clustering, we apply the *k*-means algorithm to the user embedding representations, obtaining *k* cluster centroids C_i and *k* distinct cluster sets *C*. During the process of node feature learning in CLKT, we consider the cluster centroid of the node's assigned cluster as the positive sample in the contrastive learning formula. We treat all other cluster centroids that do not belong to the same cluster as negative samples. We propose the user's contrastive learning based on InfoNCE, which is defined as follows:

$$L_{pro}^u = \sum_{u \in U} -\log \frac{exp(e_u * C_i/\tau)}{\sum_{C_t \in C} exp(e_u * C_t/\tau)} \tag{18}$$

Where e_u represents the current embedding representation of the user node, C_i represents the centroid of the cluster to which the node belongs, and C_t represents the centroids of the non-cluster sets for the current node, which are treated as negative examples in the contrastive learning formula, and τ is the temperature coefficient. We generate the contrastive learning formula for items in the same way, represented as L_{pro}^i.

The final contrastive learning loss function is the sum of the two aforementioned loss functions:

$$L_{pro} = \alpha * L_{pro}^u + L_{pro}^i \tag{19}$$

Where α is a parameter used to control the weights of users and items.

E. Top-N Recommendation

We predict the likelihood of interaction between user *u* and item *i* by taking the dot product between the fused embeddings e_u^F and e_i^F : $y_{u,i}^{'} = e_u^{F^T} * e_i^F$. Where $y_{u,i}^{'} \in R$ represents the score of the potential interaction between user *u* and item *i*. The $y_{u,i}^{'}$ is the probability of interaction between user *u* and item *i*. We utilize the Bayesian Personalized Ranking (BPR) pairwise loss function. For each user, there is a corresponding positive sample of an interacted item and a negative sample of a non-interacted item. The specific formula for the prediction loss function is as follows:

$$L_{bpr} = \sum_{(u,i+,i-)} -\ln(si(y'_{u,i+} - y'_{u,i-})) + \lambda\|\theta\|^2 \quad (20)$$

Where ln() denotes the natural logarithm, *si*() denotes the sigmoid function, λ represents the hyperparameter controlling the weight of the regularization term, $y'_{u,i+}$ represents the dot product of the positive sample, and $y'_{u,i-}$ represents the dot product of the negative sample.

We combine the BPR loss function, the contrastive learning loss function based on side information, and the contrastive learning loss function based on clustering to obtain the overall training loss function, which is defined as follows:

$$Loss = L_{bpr} + \beta * L_{ssl} + \alpha * L_{pro} \quad (21)$$

Where β and α represent the parameters that control the weights of the contrastive learning based on side information, and the contrastive learning based on clustering, respectively. The preference scores of users for items are calculated using dot product, and the scores are used to generate a recommendation list in descending order. The Top-*N* items from the list are selected as the final recommendations.

V. Experiments and Evaluation

A. Datasets

We evaluated the performance of our method using three publicly available datasets. Table I presents the statistical information of these three datasets. The Ciao and Epinions datasets are two recommendation datasets collected from online platforms. They include user ratings of different levels for items, as well as user trust relationships and category relationships among items. The Yelp dataset contains heterogeneous relationships (such as user social relationships, place rating relationships, business attributes, etc.) within the Yelp local business platform.

TABLE I
Statistics of the Datasets

Dataset	No.users	No.items	Interaction information
Ciao	6776	101415	265308
Epinions	15210	233929	630391
Yelp	161305	114852	957923

B. Evaluation Metrics

To evaluate the performance of the Top-*N* recommendation algorithm, we adopt two widely used metrics HR@*N* and NDCG@*N*.

1) Hit Ratio

The Hit Rate(HR) metric emphasizes the accuracy of the model by measuring whether the desired items of the users are included in the recommended items. *S* represents the number of samples or the number of desired items by users. The variable hit(*i*) represents whether the *i*-th desired item is present in the model's recommended item list. If the item is present in the list, the value of hit(*i*) is 1; otherwise, it is 0. The specific formula is as follows:

$$HR@N = \frac{1}{S}\sum_{i=1}^{S} hit(i) \quad (22)$$

2) Normalized Discounted Cumulative Gain

Normalized Discounted Cumulative Gain(NDCG) is an evaluation metric that measures the quality of a ranking result. It is an improvement over Discounted Cumulative Gain(DCG) by normalizing the values. NDCG can reflect the discrepancy between the recommended list of items and the user's actual interaction list, thus evaluating the accuracy of predictions. The formula for calculating NDCG@*N* is as follows:

$$NDCG@N = \frac{DCG}{IDCG} \quad (23)$$

The formula for calculating Discounted Cumulative Gain(DCG) and Ideal Discounted Cumulative Gain(IDCG) are as follows:

$$DCG = \sum_{i=1}^{N} \frac{rel_i}{\log_2(i+1)} \quad (24)$$

$$IDCG = \sum_{i=1}^{|REL|} \frac{rel_i}{\log_2(i+1)} \quad (25)$$

Where rel_i represents the true relevance score of the *i*-th result, and |REL| represents the set of the N highest-ranked results based on the true relevance scores in the best possible order. *N* is set to 10. The comprehensive ranking strategy is used to rank all candidate items that have not been interacted with.

C. Parameter Settings

CLKT is implemented using PyTorch and the Adam optimizer is used to optimize the model parameters. In the implementation of CLKT, the learning rate is set to 0.045 and the batch size is set to 8192. The embedding dimension is set to 32. The graph network has 2 layers. We evaluate CLKT by selecting 1 positive example and 99 negative examples for each user.

D. Baselines

The details of the baseline methods are described as follows:

SAMN[21]: The model incorporates an attention-based memory network to consider different social relationships and enhance the recommendation impact of the user-item model.

DGRec[22]: The approach utilizes recurrent neural networks to dynamically model user interests and employs graph-based attention networks to model the social influence in recommendations.

ETANN[23]: It introduces an adaptive algorithm that encodes the social domain into the user-item interaction patterns based on user-item interactions.

NGCF[12]: It incorporates social recommendations among users into collaborative filtering using GNN, utilizing graph convolutional operations for message passing.

KGAT[24]: The model incorporates item-based relationships into graph attention mechanisms to enhance the recommendation system.

GraphRec[25]: It jointly models the user-user social graph and the user-item interaction graph to capture the heterogeneous information in recommendations.

HERec[26]: It encodes the heterogeneous information in recommendation based on a meta-path random walk.

HAN[27]: The model applies meta-paths and incorporates a self-attention mechanism to generate representations for users and items.

TABLE II
EXPERIMENT RESULTS OF CLKT AND OTHER BASELINES

Dataset	Metrics	SAMN	DGRec	ETANN	NGCF	KGAT	GraphRec	HERec	HAN	HGT	HeCo	SMIN	MHCN	HGCL	CLKT
Ciao	HR@10	0.6576	0.6653	0.6738	0.6945	0.6601	0.6825	0.6800	0.6589	0.6939	0.6867	0.7108	0.7053	0.7338	**0.7419**
	NDCG@10	0.4561	0.4953	0.4665	0.4894	0.4512	0.4730	0.4712	0.4469	0.4869	0.4867	0.5012	0.4928	0.5215	**0.5372**
Epinions	HR@10	0.7592	0.7603	0.7650	0.7984	0.7510	0.7723	0.7642	0.7505	0.8150	0.7998	0.8179	0.8201	0.8226	**0.8329**
	NDCG@10	0.5614	0.5668	0.5663	0.5945	0.5578	0.5751	0.5495	0.5275	0.6126	0.5910	0.6137	0.6158	0.6166	**0.6272**
Yelp	HR@10	0.7910	0.7950	0.8031	0.8265	0.7881	0.8098	0.7928	0.7731	0.8364	0.8359	0.8478	0.8344	0.8712	**0.8802**
	NDCG@10	0.5516	0.5593	0.556	0.5854	0.5501	0.5679	0.5612	0.5604	0.5883	0.5847	0.5993	0.5799	0.6310	**0.6495**

TABLE III
RESULTS OF CLKT ABLATION EXPERIMENTS

Dataset	Metrics	CLKT-uu	CLKT-ii	CLKT-m	CLKT-ccl	CLKT-pcl	CLKT
Ciao	HR@10	0.7249	0.7216	0.7188	0.7226	0.7338	**0.7419**
	NDCG@10	0.5147	0.5155	0.5113	0.5124	0.5215	**0.5372**
Epinions	HR@10	0.8285	0.8204	0.8052	0.7926	0.8226	**0.8329**
	NDCG@10	0.6166	0.6127	0.5913	0.5855	0.6166	**0.6272**
Yelp	HR@10	0.8646	0.8623	0.8582	0.8426	0.8682	**0.8802**
	NDCG@10	0.6255	0.6208	0.6013	0.5955	0.6377	**0.6495**

HGT[28]: It introduces the Heterogeneous Mutual Attention (HMA) mechanism for message passing.

HeCo[29]: A model is a self-supervised approach that integrates contrastive learning with heterogeneous GNN to consider both local and higher-order graph structures. Different embeddings encoded with meta-path-based connections are used for contrastive learning.

SMIN[30]: It is a self-supervised social recommendation system that incorporates an auxiliary graph task into the main task to improve recommendation performance.

MHCN[10]: It designs a multi-channel hypergraph convolutional network to consider the global relationships among users.

HGCL[31]: This method combines user-item interaction information with side information to construct a contrastive learning formulation, thereby improving the performance of the recommendation algorithm.

E. Experimental Results and Analysis

The experimental results of the comparison between CLKT and the baseline methods are shown in Table II.

From Table II, it can be observed that most side information-based GNN algorithms (such as Heco) outperform other general recommendation algorithms (such as KGAT) in terms of recommendation performance. This indicates that the social relationships between users and the similarity relationships between items have a better impact on recommendations. The inclusion of self-supervised learning algorithms in recommendation models, such as SMIN, shows excellent performance. This indicates the rationality of applying self-supervised learning in embedding node features for recommendations.

Our proposed CLKT consistently outperforms other models in terms of recommendation performance compared to the current state-of-the-art recommendation models. Furthermore, it shows the most significant improvement in the Ciao dataset, which validates that the model is more suitable for handling sparse data problems. The performance improvement of CLKT can be attributed to several factors:

- CLKT achieves improved node embedding by effectively transferring knowledge between side information and user-item interaction information, enabling better learning of important features between users and items.
- CLKT leverages contrastive learning based on side information and contrastive learning based on clustering, allowing it to alleviate data sparsity and perform node feature embedding from different node relationships. This enhances CLKT's recommendation performance by capturing diverse aspects of node interactions.

1) Ablation Experiments

CLKT enhances recommendation embedding by employing contrastive learning based on side information. It also incorporates knowledge transfer to learn important features between side information and user-item interaction information. Additionally, CLKT utilizes contrastive learning based on clustering to enhance feature learning from a clustering perspective in the recommendation embedding. To validate the effectiveness of these contrastive learning approaches in improving recommendation performance, ablation experiments were conducted using HR@10 and NDCG@10 as evaluation metrics. The experimental results are presented in Table III.

- CLKT-uu: We removed the input of user side information from CLKT, thereby reducing the relationships between users in the learning process of user preferences.
- CLKT-ii: We removed the input of item side information from CLKT, thereby reducing the relationships between items in the learning process of user preferences.
- CLKT-m: We removed the feature transformer and knowledge transfer network from CLKT, thereby reducing the extraction of important features in contrastive learning based on side information.
- CLKT-ccl: We removed the contrastive learning based on

side information from CLKT, thereby limiting the association between user-item interaction information and side information.

- CLKT-pcl: We removed the contrastive learning based on clustering from CLKT, thereby reducing the influence of different types of node relationships on node feature embedding.

The ablation experiments results of CLKT and its variant methods on Ciao, Epinions, and Yelp datasets are shown in Table III. The recommendation performance of CLKT-uu and CLKT-ii significantly decreases compared to the CLKT, indicating that the feature relationships in side information can enhance the performance of recommendations. The performance of CLKT consistently outperforms CLKT-m in all scenarios, indicating that knowledge transfer enhances the process of node embedding and improves the effectiveness of node feature learning. The performance of CLKT-ccl shows a partial decrease compared to CLKT, indicating that side information-based contrastive learning can alleviate the issue of data sparsity and effectively uncover the feature associations between side information and user-item interaction information. The lower performance of CLKT-pcl compared to CLKT indicates that there are feature associations present between different types of node relationships as well.

2) Data Sparsity Experiments

To validate the effectiveness of CLKT across datasets with varying degrees of sparsity, we divided the user set into five groups based on their interaction levels. The groups were formed by grouping users with similar interaction counts, with the users sorted in descending order of their total interactions. This grouping represents different levels of user activity, allowing us to evaluate the performance of CLKT across various levels of user engagement. To compare the performance of CLKT with the baselines, we evaluated them using HR@10 and NDCG@10 as evaluation metrics. The experimental results are shown in Fig. 2. The results demonstrate that CLKT consistently outperforms the baseline algorithms in terms of recommendation performance across different levels of sparsity. This validates the superiority of CLKT in the recommendation task.

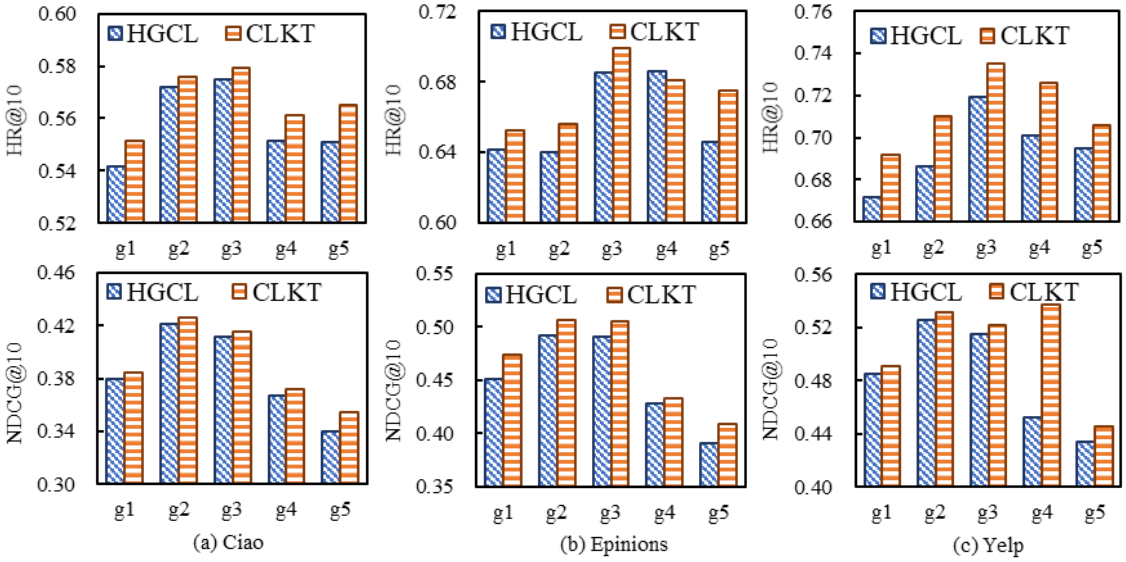

Fig. 2. Sparse Experimental Results of CLKT

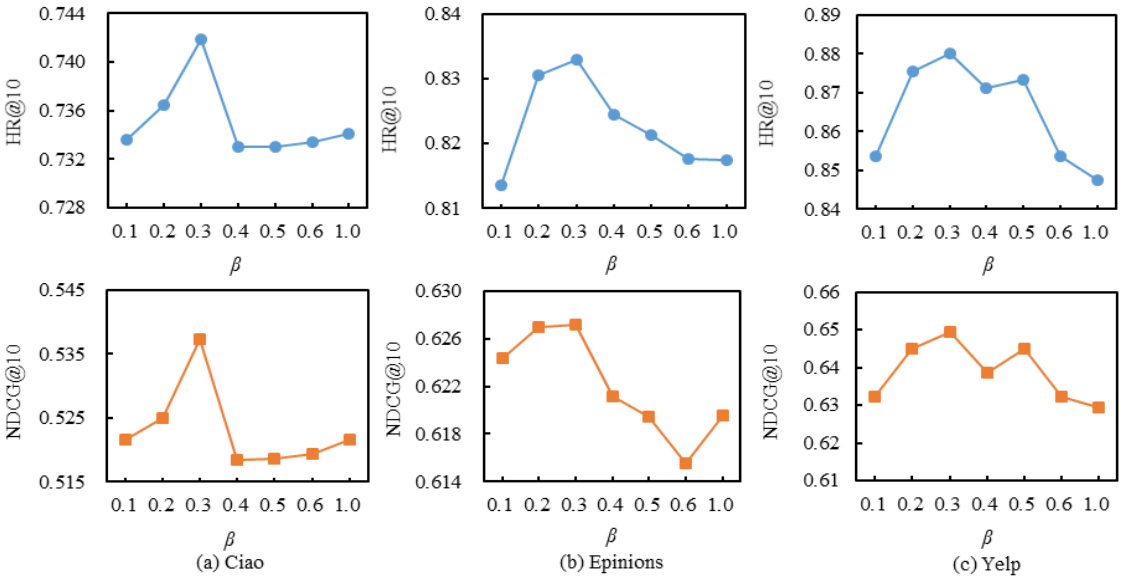

Fig. 3. Performance Comparison of different β

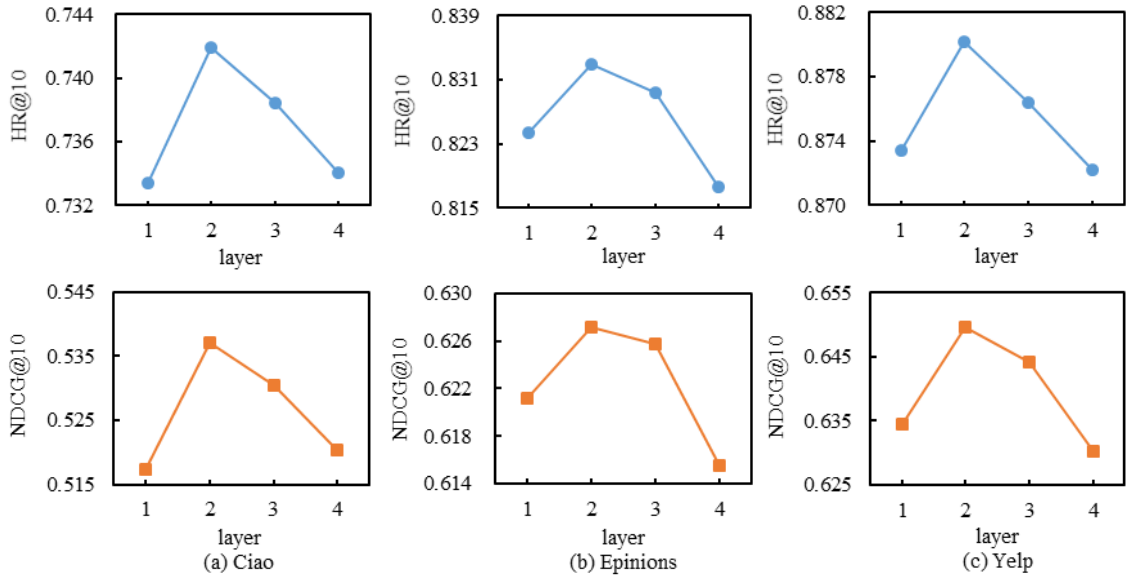

Fig. 4. Performance Comparison of different layers

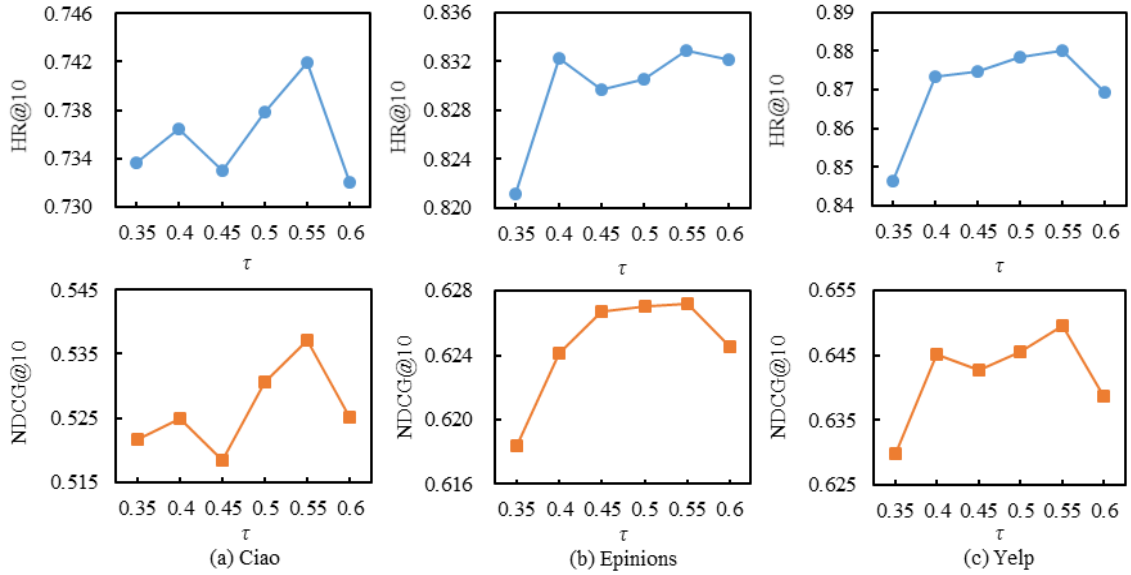

Fig. 5. Performance Comparison of different τ

3) Hyperparameter Analysis

In this section, we discusse the impact of parameters on CLKT. The experiments mainly focus on comparing the performance of CLKT on the Ciao, Yelp, and Epinions datasets with different values of β, layer, τ, and k.

● Impact of the Coefficient β

β is defined to balance the impact of cluster-based contrastive learning on the recommendation algorithm in (21). In this section, β is tested within the range of 0 to 1, and the experimental results are shown in Fig. 3. It can be observed that β needs to be adjusted to a specific value in order to achieve significant performance improvement in CLKT. When β is set to 0.3, CLKT shows the best recommendation performance. This indicates that conducting feature learning through multiple types of relationships can effectively enhance node embedding capability in CLKT.

● Impact of the Graph Propagation Layers

In the GNN, we conducted experiments on CLKT with propagation layers ranging from 1 to 4. The experimental results are shown in Fig. 4. From Fig. 4, it can be observed that the recommendation algorithm achieves the best performance when the number of layers is set to 2. This indicates that multiple layers facilitate effective information propagation, allowing the model to capture rich neighborhood and semantic information. However, when the number of layers reaches 4, the recommendation performance starts to decline. This suggests that an excessive number of layers can result in the over-smoothing of embeddings in CLKT, leading to a negative impact on the recommendation performance.

● Impact of the Temperature τ

The parameter τ plays a critical role in the effectiveness of contrastive learning, as demonstrated in (15) and (16). We conducted a series of experiments, varying τ from 0.35 to 0.6, and the experimental results are presented in Fig. 5. It can be observed that the performance of CLKT varies with changes in τ, and optimal performance is achieved when τ is adjusted to an appropriate value. Excessive tuning of τ may diminish the auxiliary capability of contrastive learning in enhancing recommendation performance.

● Impact of Clustering Quantity k

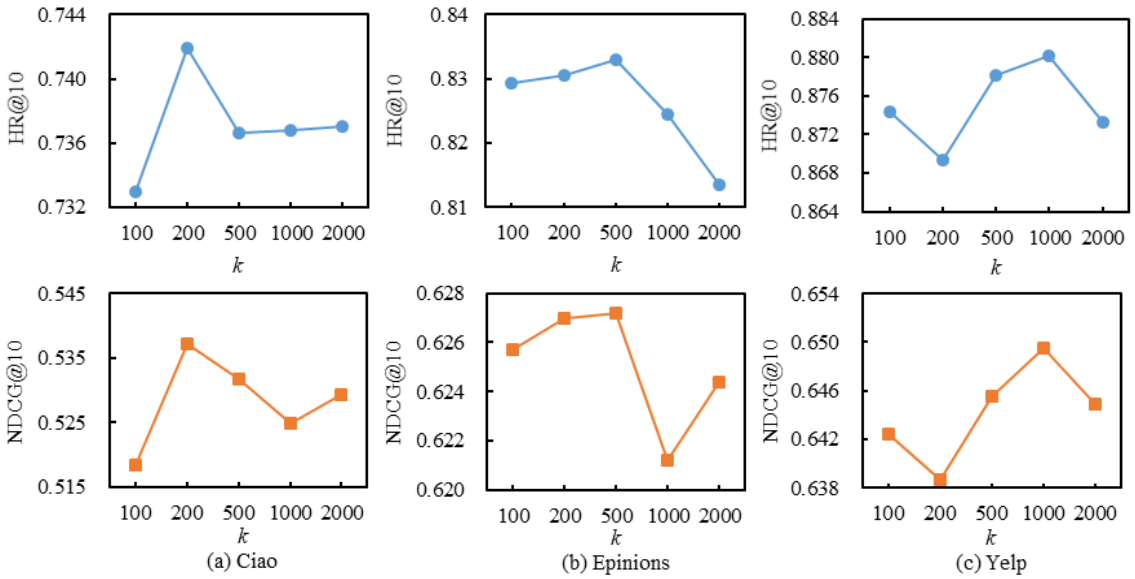

Fig. 6. Performance Comparison of different k

The parameter k represents the number of clusters in the clustering algorithm. To study the impact of the number of clusters on the performance of contrastive learning based on clustering, we conducted experiments varying k from 100 to 2000, and the results are presented in Fig. 6. The experimental results indicate that CLKT achieves significant performance improvement only when k is adjusted to an appropriate value in different datasets. This suggests that incorporating multiple node relationships in node embedding can enhance recommendation performance. However, when k becomes excessively large, the recommendation performance notably declines. This indicates that the number of clusters needs to be adjusted to an appropriate value to achieve better integration of contrastive learning based on clustering and graph collaborative filtering algorithms.

VI. Conclusion

In this paper, we proposed graph Contrastive Learning with Knowledge Transfer(CLKT). This approach incorporates side information into the recommendation method and combines knowledge transfer with contrastive learning based on side information. This enables CLKT to adaptively perform node feature embedding for users and items. We introduce contrastive learning based on clustering to explore multiple node relationship features, thereby improving the performance of graph collaborative filtering algorithms. Extensive experiments were conducted on three real-world datasets, and the results demonstrate that CLKT outperforms existing state-of-the-art recommendation algorithms significantly. This confirms the effectiveness of the different structures employed in CLKT for optimizing recommendation performance. In future work, we plan to further explore contrastive learning and investigate the impact of different types of node relations on recommendations. Another direction of research is to explore the mining of other types of relevant information from side information.

References

[1] K. Mao, J. Zhu, J. Wang, et al. “SimpleX: A simple and strong baseline for collaborative filtering,” in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, New York, 2021, pp. 1243-1252.

[2] J. Xia, D. Li, H. Gu, et al. “Incremental graph convolutional network for collaborative filtering,” in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, New York, 2021, pp. 2170-2179.

[3] X. Xie, Z. Liu, S. Wu, et al. “Causcf: Causal collaborative filtering for recommendation effect estimation,” in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, New York, 2021, pp. 4253-4263.

[4] W. Chen, M. He, Y. Ni, et al. “Global and Personalized Graphs for Heterogeneous Sequential Recommendation by Learning Behavior Transitions and User Intentions,” in Proceedings of the 16th ACM Conference on Recommender Systems, New York, 2022, pp. 268-277.

[5] J. Zheng, Q. Ma, H. Gu, et al. “Multi-view denoising graph auto-encoders on heterogeneous information networks for cold-start recommendation,” in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, New York, 2021, pp. 2338-2348.

[6] X. Wang, Q. Li, D. Yu, et al. “Off-policy Learning over Heterogeneous Information for Recommendation,” in Proceedings of the ACM Web Conference 2022, New York, 2022, pp. 2348-2359.

[7] X. Cai, C. Huang, L. Xia, et al. “LightGCL: Simple Yet Effective Graph Contrastive Learning for Recommendation,” arXiv:2302.08191, 2023. Available: https://arxiv.org/abs/2302.08191.

[8] W. Wang, L. Wei, Y. Li, et al. “Attentional Meta-path Contrastive Graph Convolutional Networks for Knowledge Concept Recommendation,” in 2022 Tenth International Conference on Advanced Cloud and Big Data (CBD), New York, 2022, pp. 200-205.

[9] X. Xia, H. Yin, J. Yu, et al. “Self-supervised hypergraph convolutional networks for session-based recommendation,” in Proceedings of the AAAI Conference on Artificial Intelligence, Menlo Park, CA, 2021, 35(5), pp. 4503-4511.

[10] J. Yu, H. Yin, J. Li, et al. “Self-supervised multi-channel hypergraph convolutional network for social recommendation,” in Proceedings of the Web Conference 2021, New York, 2021, pp. 413-424.

[11] Y. Chen, Z. Liu, J. Li, et al. “Intent contrastive learning for sequential recommendation,” in Proceedings of the ACM Web Conference 2022, New York, 2022, pp. 2172-2182.

[12] X. Wang, X. He, M. Wang, et al. “Neural graph collaborative filtering,” in Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, 2019, pp. 165-174.

[13] X. He, K. Deng, X. Wang, et al. “Lightgcn: Simplifying and powering graph convolution network for recommendation,” in Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, 2020, pp. 639-648.

[14] J. Sun, Y. Zhang, C. Ma, et al. “Multi-graph convolution collaborative filtering,” in 2019 IEEE International Conference on Data Mining (ICDM), New York, 2019, pp. 1306-1311.

[15] J. Wu, X. Wang, F. Feng, et al. “Self-supervised graph learning for recommendation,” in Proceedings of the 44th International ACM

SIGIR Conference on Research and Development in Information Retrieval, New York, 2021, pp. 726-735.
[16] Z. Lin, C. Tian, Y. Hou, et al. “Improving graph collaborative filtering with neighborhood-enriched contrastive learning,” in Proceedings of the ACM Web Conference 2022, New York, 2022, pp. 2320-2329.
[17] Y. Wei, X. X. Wang, Q. Li, et al. “Contrastive learning for cold-start recommendation,” in Proceedings of the 29th ACM International Conference on Multimedia, New York, 2021, pp 5382-5390.
[18] T. Yao, X. Yi, D. Z. Cheng, et al. “Self-supervised learning for large-scale item recommendations,” in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, New York, 2021, pp 4321-4330.
[19] X. Xie, F. Sun, Z. Liu, et al. “Contrastive learning for sequential recommendation,” in 2022 IEEE 38th International Conference on Data Engineering (ICDE), New York, 2022, pp. 1259-1273.
[20] C. Zhou, J. Ma, J. Zhang, et al. “Contrastive learning for debiased candidate generation in large-scale recommender systems,” in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, New York, 2021, pp 3985-3995.
[21] C. Chen, M. Zhang, Y. Liu, et al. “Social attentional memory network: Modeling aspect-and friend-level differences in recommendation,” in Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, New York, 2019, pp 77-185.
[22] W. Song, Z. Xiao, Y. Wang, et al. “Session-based social recommendation via dynamic graph attention networks,” in Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, New York, 2019, pp 555-563.
[23] C. Chen, M. Zhang, C. Wang, et al. “An efficient adaptive transfer neural network for social-aware recommendation,” in Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, 2019, pp 225-234.
[24] X. Wang, X. He, .Y Cao, et al. “Kgat: Knowledge graph attention network for recommendation,” in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, New York, 2019, pp 950-958.
[25] W. Fan, Y. Ma, Q. Li, et al. “GNN for social recommendation,” in The World Wide Web Conference, New York, 2019, pp 417-426.
[26] C. Shi, B. Hu, W. X. Zhao, et al. “Heterogeneous information network embedding for recommendation,” IEEE Transactions on Knowledge and Data Engineering, vol. 31(2), pp. 357-370, 2018.
[27] X. Wang, H. Ji, C. Shi, et al. “Heterogeneous graph attention network,” in The World Wide Web Conference, New York, 2019, pp 2022-2032.
[28] Z. Hu, Y. Dong, K. Wang, et al. “Heterogeneous graph transformer,” in Proceedings of the Web Conference 2020, New York, 2020, pp 2704-2710.
[29] X. Wang, N. Liu, H. Han, et al. “Self-supervised heterogeneous graph neural network with co-contrastive learning,” in Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, New York, 2021, pp 1726-1736.
[30] X. Long, C. Huang, Y. Xu, et al. “Social recommendation with self-supervised metagraph informax network,” in Proceedings of the 30th ACM International Conference on Information & Knowledge Management, New York, 2021, pp 1160-1169.
[31] M. Chen, C. Huang, L. Xia, et al. “Heterogeneous Graph Contrastive Learning for Recommendation,” in Proceedings of the 16th ACM International Conference on Web Search and Data Mining, New York, 2023, pp 544-552.