Bi-Directional Transfer Graph Contrastive Learning for Social Recommendation

Lei Sang, Mingyuan Liu, Yi Zhang, Yuee Huang, and Yiwen Zhang

***Abstract*—Graph Neural Networks (GNNs) have emerged as an effective approach for social recommender systems. GNNs excel at capturing the graph-structured semantic information within the collaborative interaction graph and social networks. Recently, some methods have introduced self-supervised learning to GNNs, aiming to enhance recommendation performance by mitigating the data sparsity issue. However, these methods treat the interaction graph and social network as separate entities, which severely limits the number of samples available for self-supervised learning. Moreover, these separated methods also exacerbate the problem of information islands in collaborative and social domains, resulting in suboptimal performance. To tackle these challenges, we propose an innovative self-supervised social recommendation method called Bi-directional Transfer Graph Contrastive Learning (BTGCL). BTGCL jointly encodes node representations within both collaborative domain and social domain, then generates node views through feature augmentation. To bridge the information gap between domains, we devise a bi-directional migration mechanism that aligns features from the collaborative and social domains of the same positive pair. Through extensive experiments conducted on three publicly available datasets, we demonstrate the effectiveness of our proposed method in enhancing social recommendation performance.**

***Index Terms*—Social recommendation, graph neural networks, contrastive learning.**

I. Introduction

PERSONALIZED recommender systems have gained significant traction across various platforms, including e-commerce, social networks, and news or video suggestions. [1], [2]. Traditional recommendation methods, such as Collaborative Filtering (CF) [3], [4], leverage collaborative signals derived from historical user-item interaction data to generate refined representations of users and items. However, CF-based methods are impacted by the problem of data sparsity, where sparse user–item interactions lead to inaccurate recommendations. The proliferation of social networks has considerably facilitated the expression of users' preferences for items within these platforms, enabling them to share their interests with their social connections [5], [6], [7]. As a result, social recommendation techniques have garnered significant attention within these networks. The primary objective of such approaches is to capture the social influence and correlation existing among users, with the purpose of effectively mitigating the issue of sparsity [8], [9], [10], [11]. Traditional social recommendation mainly focused on enhancing the representation of users by employing two principal methodologies: the direct integration of user relationships and the regulation of the user's embedding learning process using information obtained from social neighbors [8], [10]. However, these methodologies are subject to specific constraints that affect their efficacy. These constraints include the underlying assumption of social homogeneity, heavy dependence on explicit social connections, and difficulties related to scalability. These limitations impede the ability of these approaches to provide precise recommendations for users within social networks.

In recent years, the progress in Graph Neural Network (GNN) techniques has led to the emergence of GNNs as a promising solution to tackle the challenges mentioned in recommender systems [12]. GNN–based recommendations aim to capture user-item interactions by representing them as graph structures. GNNs can iteratively leverage aggregate information from multi-hop neighbor nodes, thus learning expressive node representations for users and items [13], [14], [15]. The application of GNNs in social recommendation has also attracted significant research attention. In addition to modeling user–item interaction graphs from the collaborative domain, social relationships among users from the social domain are also abstracted into graph structures, commonly referred to as social networks [13]. For instance, GraphRec [13] is a pioneering study that introduces GNNs into social recommendation by treating user–item interactions and user relations as graph data. Another noteworthy approach, DiffNet++ [16], utilizes node-level attention mechanism to integrate user representations derived from GNN in both the social and collaborative domains. Although GNN-based recommender systems are proven to be effective, challenges persist due to sparse supervisory signals [17] and noisy interactions [18].

A recent trend in social recommender systems is the incorporation of Self-Supervised Learning (SSL) techniques [19], which provide additional supervised signals to mitigate data sparsity.

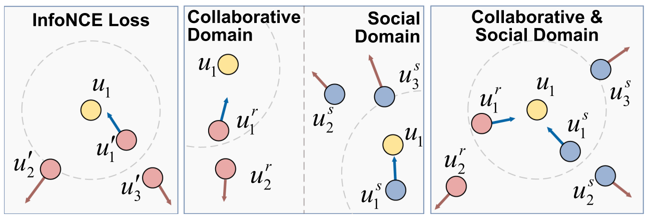
Manuscript received 27 June 2023; revised 17 October 2023; accepted 6 November 2023. Date of publication 10 April 2024; date of current version 15 May 2025. This work was supported in part by the National Science Foundation of China under Grant 62272001 and Grant 62206002, in part by the Anhui Provincial Natural Science Foundation under Grant 2208085QF195, in part by the University Collaborative Innovation Project of Anhui Province under Grant GXXT-2021-087, and in part by the Anhui Province Key Research and Development Program under Grant 202104a05020058. Recommended for acceptance by Special Issue On Foundation Model for Social Media. *(Corresponding author: Yiwen Zhang.)*

Lei Sang, Mingyuan Liu, Yi Zhang, and Yiwen Zhang are with the School of Computer Science and Technology, Anhui University, Hefei 230601, China (e-mail: sanglei@ahu.edu.cn; lmyundo@stu.ahu.edu.cn; zhangyi.ahu@gmail.com; zhangyiwen@ahu.edu.cn).

Yuee Huang is with the School of Public Health, Wannan Medical College, Wuhu 241002, China (e-mail: huangyewindow@163.com).

This article has supplementary downloadable material available at https://doi.org/10.1109/TBDATA.2024.3387340, provided by the authors.

Digital Object Identifier 10.1109/TBDATA.2024.3387340



(a) InfoNCE (b) Common Contrastive (c) Transfer Contrastive

Fig. 1. (a) Illustration of InfoNCE Loss. (b) Existing self-supervised learning methods for social recommendation usually compare user u_1 with the same node in the domain separately. (c) Transfer Contrastive fuses the same nodes in two domains to increase the number of samples and achieve knowledge migration.

SSL-based recommendation mainly include two steps: (1) Augmentation: augmenting the original user–item bipartite graph with structure perturbations (e.g., edge/node dropout at a certain ratio); (2) Contrastive Learning: maximizing the consistency of representations learned from different graph augmentations with contrastive learning under a joint learning framework. For example, MHCN [20] integrates SSL into the training of a hypergraph convolutional network, leveraging hierarchical mutual information maximization to capture high-order connectivity. SEPT [21], tackles the data sparsity issue by applying data augmentation, constructing two distinct views and employing a triple training strategy to generate self-supervised signals. DcRec [22] adopts a separate learning process for the collaborative and social domains with the goal of facilitating user knowledge transfer.

Despite the notable improvements in performance exhibited by self-supervised recommendation methods, we posit that these approaches remain encumbered by several unaddressed issues. (1) First, the graph augmentation techniques introduced by SSL [20], [21], frequently utilized for generating divergent views, impose a steep increase in time complexity due to their inherent dependence on laborious and time-consuming processes such as sampling, dropping, and singular value decomposition [23], [24]. This results in a significant rise in the time overheads associated with model training. (2) Furthermore, the performance improvement in SSL-based recommendation is primarily attributed to contrastive loss (e.g., InfoNCE [25]) rather than data augmentation [26]. This is because contrastive learning plays a role in establishing a uniform distribution of users and items in the same feature space [27], which mitigates popularity bias [28] and allows for more recommendations of long-tail items.

More importantly, when applied in social recommendation, existing SSL-based recommendation methods fail to effectively utilize positive and negative samples across multiple domains. This limitation ultimately results in suboptimal performance and impedes the transfer of information across different domains within social recommender systems. The mainstream SSL-based recommendation methods [23], [29], [30] focus on maximizing the similarity between nodes and their corresponding entities in another view (i.e., positive samples) while keeping them further away from other unrelated nodes (i.e., negative samples) in the feature space [25], as shown in Fig. 1(a). However, social recommendation includes two domains [31], namely the collaborative domain and the social domain. As illustrated in Fig. 1(b), most existing works [20], [21], [22] only consider the unlabelled information of the same node in a single domain, resulting in the absence of sufficient positive and negative samples for contrastive learning to provide supervisory signals [32]. Moreover, the consistencies and differences between multiple perspectives in different domains [12], [22] are ignored; as a result, the knowledge in each domain is independent from that in the other, making it difficult to achieve inter-domain transfer of information and leading to sub-optimal performance.

In this paper, we propose a novel method for social recommendation, named Bi-directional Transfer Graph Contrastive Learning (BTGCL), which thoroughly utilizes the homogeneity and disparity of information across collaborative and social domains. Specifically, we use graph encoders to model higher-order relationships between users and items in interaction graphs and social networks to obtain their embeddings. Next, to reduce the number of operations, we discard heavy graph augmentation (GA) in favor of light feature augmentation (FA), in which random noise is infused into the node representation to generate different views of the node. Moreover, to facilitate knowledge transfer between domains, we advocate a bi-directional transfer contrastive learning mechanism, which accommodates domain-specific features in disparate domains and transposes user features from another domain to this feature space so that InfoNCE loss [25] is calculated separately, as shown in Fig. 1(c). This straightforward yet powerful mechanism enables reciprocal knowledge transfer between domains while increasing sample amount and dismantling informational barriers. To summarize, this paper makes the following contributions:

- We propose a new graph self-supervised framework for social recommendation, named BTGCL, which generates representations through FA-based graph encoders to model the high-order relationships and improve the performance of the recommendation task.
- We design a new bi-directional transfer contrastive learning mechanism, which aligns features from the collaborative and social domains of the same positive pair to dismantle informational barriers.
- We conduct empirical experiments on three datasets to evaluate the performance of BTGCL against nine different baselines. Extensive results demonstrate that our method consistently outperforms the other competing baselines.

II. Preliminaries

In the following section, we provide an introduction to the fundamental concepts employed in this study. Table I lists the main notations and their definitions. Suppose we have two types of data in the recommendation scenario: the user–item interaction graph $\mathcal{G}_r$ and the user social network $\mathcal{G}_s$. Let $\mathcal{U} = \{u_1, u_2, \ldots, u_m\}(|\mathcal{U}| = m)$ and $\mathcal{I} = \{i_1, i_2, \ldots, i_n\}(|\mathcal{I}| = n)$ represent the set of users and items, respectively, where m denotes the number of users and n denotes the number of items. The user–item interaction matrix $\mathbf{R} \in \{0,1\}^{|\mathcal{U}|\times|\mathcal{I}|}$ is a binary matrix with entries 0 and 1 that represent user–item interactions in $\mathcal{G}_r$.

TABLE I
NOTATIONS AND EXPLANATIONS

Notations	Description
$\mathcal{G}_r$	collaborative information network
$\mathcal{G}_s$	social information network
$\mathcal{U}$	the set of users
$\mathcal{I}$	the set of items
$\mathbf{R}$	collaborative interaction matrix
$\mathbf{S}$	social relation matrix
$\mathbf{Z}^{(l)}$	embedding of node at the l-th layer
$\mathbf{z}_u^{(l)}$	embedding of user u at the l-th layer
$\mathbf{z}_u^r, \mathbf{z}_u^s$	final embedding of user u in collaborative and social domain
$\mathbf{p}_u^r, \mathbf{p}_u^s$	transferred embedding of user u in collaborative and social domain
$\tilde{\mathbf{z}}_u^r, \tilde{\mathbf{z}}_u^s$	1-th layer embedding of user u in collaborative and social domain
$\mathbf{z}_i^r$	final embedding of item i in collaborative domain
$\tilde{\mathbf{z}}_i^r$	1-th layer embedding of item i in collaborative domain
H	the encoder
τ	the noise vector
ϵ	the strength of noise
λ_1, λ_2	coefficient of transfer contrastive learning
λ_3	regularization coefficient

The social network matrix $\mathbf{S} \in \{0,1\}^{|\mathcal{U}|\times|\mathcal{U}|}$ denotes the social adjacency matrix, which is binary and symmetric, since we focus on undirected social networks with bidirectional relations.

Next, we outline the common paradigm of GNN-based collaborative filtering methods [33], [34]. The core concept is to apply a neighborhood aggregation scheme on $\mathcal{G}$ to update the representation of target nodes by aggregating the representations of sampled neighbors:

$$\mathbf{Z}^{(l)} = AGG(\mathbf{Z}^{(l-1)}, \mathcal{G}), \tag{1}$$

where $\mathbf{Z}^{(l)}$ represents the embedding of the node at the l-th layer, $\mathbf{Z}^{(l-1)}$ represents that of the previous layer, and $\mathbf{Z}^{(0)}$ represents the initial embeddings (trainable parameters). AGG is the propagation function, which can be formulated for the propagate and readout stages:

$$\mathbf{z}_u^{(l)} = f_{\text{propagate}}\left(\left\{\mathbf{z}_v^{(l-1)} \mid v \in \mathcal{N}_u \cup \{u\}\right\}\right), \tag{2}$$

where $\mathcal{N}_u$ denotes the neighbor sets of user u in the interaction graph $\mathcal{G}$. For user u, the propagation function propagate-aggregates the representations of its neighbors $\mathcal{N}_u$ and itself u at the $(l-1)$-th layer to update the l-th layer representation. By doing so, the l-th layer features encode the l-order neighbors of the node in the graph. After these L layer representations have been obtained, a readout function is still needed to generate the final representation of user u for recommendation:

$$\mathbf{z}_u^{(l)} = f_{\text{readout}}\left(\left\{\mathbf{z}^{(l)} \mid l = [0, \ldots, L]\right\}\right). \tag{3}$$

Here, we can use the concatenation [33] or weighted sum [34] operation as readout function, and item representations are obtained in a similar way.

III. The Proposed Method

In this section, we present the proposed Bi-directional Transfer Graph Contrastive Learning (BTGCL). BTGCL aims to incorporate knowledge transfer into contrastive learning methods to improve the representation learning of users and items. Fig. 2 displays the framework of BTGCL in three parts: (1) feature augmentation-based graph encoder, which outputs the final representation of the node in multiple views using FA; (2) bi-directional transfer contrastive learning, which collaboratively improves the embeddings through a bi-directional transfer mechanism; and (3) the joint learning strategy, which combines self-supervised learning as an auxiliary task with the GNN.

A. Feature Augmentation-Based Graph Encoder

In social recommendation, a user's final embedding relies on the social influence of that user's friends and their own preferences. Users' behavior may be influenced by what their friends do or think [35], which is also referred to as homogeneity in recommender systems [6]. To capture the self-supervised signals from the collaborative domain and the social domain independently, we utilize the user–item interaction matrix and the social matrix as the graph data inputs to the method, thus preventing noise caused by the mixing of information across these two domains.

By utilizing the adjacency matrices, denoted as $\mathbf{R}$ and $\mathbf{S}$, derived from the collaborative and social domains respectively, we can effectively employ data augmentation techniques to acquire diverse node representations from various perspectives. Inspired by prior research [36], [37], we introduce a graph encoder that leverages feature augmentation. This approach involves introducing controlled perturbations to the node embedding, which allows us to achieve graph data augmentation without the need for matrix operations. Moreover, our encoder utilizes the outputs of different layers to represent the target node under different viewpoints, thus mitigating the computational burden associated with employing additional encoders.

In more detail, for a given node i and its representation z_i in the d-dimensional embedding space, we can implement node-embedding-based data augmentation by:

$$\mathbf{z}' = \mathbf{z} + \tau, \tag{4}$$

where the added noise vector τ is subject to $\|\tau\|_2 = \epsilon$ and $\tau = \bar{\tau} \odot \text{sign}(\mathbf{z}), \bar{\tau} \in \mathbb{R}^d \sim U(0,1)$, while ϵ is the hyperparameter controlling the strength of the embedding noise. Among these two constraints, the first one is to control the value of τ on a hypersphere with a radius of ϵ, in order to prevent excessive feature augmentation from having a significant impact on the original data. The second condition ensures that the embedding and noise directions are correlated, preventing significant deviations.

To model user–item interactions and social relationships, we employ a GNN-based method to learn the representations of users and items in each domain. Specifically, following LightGCN [34], we eliminate nonlinear activation and feature transformation in the propagation function, and further impose

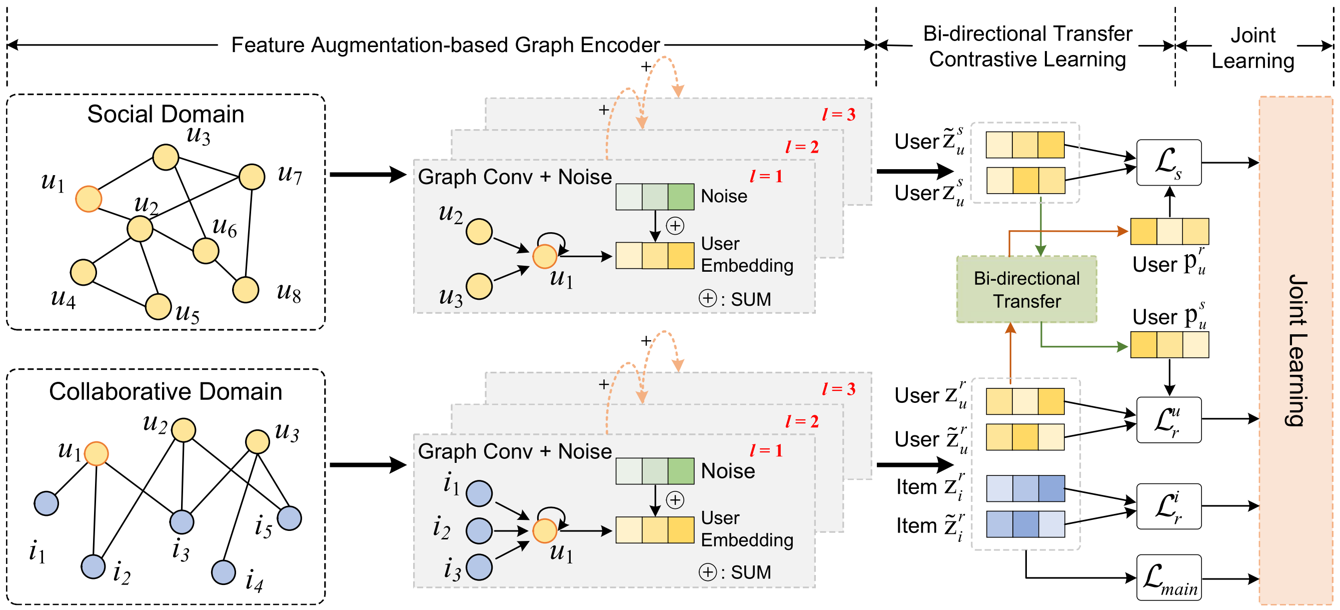


Fig. 2. Illustration of the BTGCL architecture. The embeddings are generated via feature augmentation. (1) Feature augmentation-based graph encoder, which outputs the final representation of the node in multiple views using FA; (2) Bi-directional transfer contrastive learning, which collaboratively improves the embeddings through a bi-directional transfer mechanism; and (3) The joint learning strategy, which combines self-supervised learning as an auxiliary task with the classical GNN.

random noise of different scales on the current node embeddings:

$$\mathbf{z}_u^{(l)} = \sum_{i \in N_u} \frac{1}{\sqrt{|\mathcal{N}_u|\,|\mathcal{N}_i|}} z_i^{(l-1)} + \tau_u. \tag{5}$$

After propagation with L layers, we use the weighted sum function as the readout function to combine the representations of all layers to obtain the final representation, as follows:

$$\mathbf{z}_u = \frac{1}{L}\sum_{l=1}^{L} \mathbf{z}_u^{(k)}, \tag{6}$$

which is more interpretable at the vector level:

$$\begin{aligned} \mathrm{Z}' = \frac{1}{L}\Big(\Big(\tilde{\mathrm{A}}\mathrm{Z}^{(0)} + \tau^{(1)}\Big) \\ + \Big(\tilde{\mathrm{A}}\left(\tilde{\mathrm{A}}\mathrm{Z}^{(0)} + \tau^{(1)}\right) + \tau^{(2)}\Big) + \ldots \\ + \Big(\tilde{\mathrm{A}}^L\mathbf{Z}^{(0)} + \tilde{\mathrm{A}}^{L-1}\tau^{(1)} + \ldots + \tilde{\mathrm{A}}\tau^{(L-1)} + \tau^{(L)}\Big)\Big). \end{aligned} \tag{7}$$

It is worth noting that we also omitted the initial embedding $\mathrm{Z}^{(0)}$ of both encoders when obtaining the final embedding, as we experimentally determined that a slight performance degradation is introduced when the output of the embedding includes the input embedding; however, this is not the case in the LightGCN method.

Let H_r be the collaborative domain encoder and H_s be the social domain encoder. Theoretically, these two encoders should be distinct due to the different semantics of the processed data. For simplicity, we assign the same structure to H_r and H_s and encode them in separate pairs of domain adjacency matrices $\mathbf{R}$ and $\mathbf{S}$:

$$\mathbf{Z}^r = H_r(\mathbf{E}, \mathbf{R}), \quad \mathbf{Z}^s = H_s(\mathbf{E}, \mathbf{S}), \tag{8}$$

where $\mathbf{Z}^r \in \mathbb{R}^{(m+n)\times d}, \mathbf{Z}^s \in \mathbb{R}^{m\times d}$ are the final representations of the interaction domain and social domain nodes, while E of the same size denotes the initial node embeddings, which are the common foundation shared by the two encoders.

B. Bi-Directional Transfer Contrastive Learning

Existing social graph collaborative filtering methods are primarily trained on the observed interactions in two domains independently, meaning that the potential relationships among users in different domains cannot be explicitly captured. To fully exploit domain-specific features in different domains, we design a bi-directional transfer contrastive learning based on predicted user representations with expanded samples. Specifically, this approach includes transfers in collaborative domain and transfers in social domain.

1) Transfer in Collaborative Domain: To fuse the supervised information from the collaborative domain, we design a transfer contrastive learning method based on the interaction domain. The final embedding $\mathbf{z}_u^s$ in the social domain is introduced in addition to the final embedding $\mathbf{z}_u^r$ and the first layer embedding $\tilde{\mathbf{z}}_u^r$ in the collaborative domain for user u. Considering the different user interactions in different domains, it is not reasonable to directly process the embedding in different domains. Thus, we design an MLP with one hidden layer for the embeddings to project them into the same semantic space:

$$\mathrm{p}_u^s = W^{(2)}\sigma\left(W^{(1)}\mathrm{z}_u^s + b^{(1)}\right) + b^{(2)}, \tag{9}$$

where $W^{(\cdot)} \in \mathbb{R}^{d\times d}$ and $b^{(\cdot)} \in \mathbb{R}^{d\times 1}$ are trainable parameters and $\sigma(\cdot)$ is an ELU non-linear function. With these representations, we can efficiently model a user's common features and

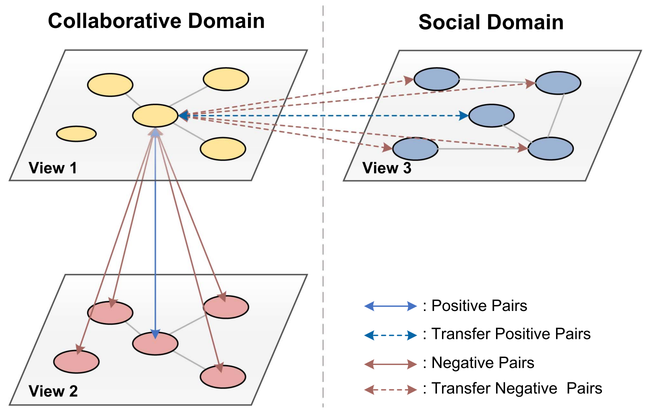


Fig. 3. Illustration of Bi-directional Transfer contrastive learning. User u_1 in view1 of the collaborative domain constitutes both positive and negative sample pairs with nodes in view2 in the collaborative domain and the migrated view3 in the social domain.

domain-specific features. Specifically, as Fig. 3 shows, we treat multiple views of users in different domains as positive pairs, and the views of any different nodes as negative pairs. Based on InfoNCE, we propose a collaborative transfer contrastive learning objective to minimize the distance of positive pairs and maximize that of negative pairs:

$$\mathcal{L}_r^u = \sum_{u\in\mathcal{U}} -\log \frac{\psi(\mathbf{z}_u^r, \tilde{\mathbf{z}}_u^r) + \psi(\mathbf{z}_u^r, \mathbf{p}_u^s)}{\sum_{v\in\mathcal{B}} \psi(\mathbf{z}_u^r, \tilde{\mathbf{z}}_v^r) + \sum_{v\in\mathcal{B}} \psi(\mathbf{z}_u^r, \mathbf{p}_v^s)}, \quad (10)$$

where $\psi(\mathbf{z}_u^r, \tilde{\mathbf{z}}_u^r) = \exp(s(\mathbf{z}_u^r, \tilde{\mathbf{z}}_u^l)/\tau)$, $s(\cdot)$ is the discriminator function that takes two vectors as the input and then scores the similarity between them (set as a cosine similarity function) and τ is the temperature to amplify the effect of discrimination in softmax.

Note that both positive and negative sample pairs of the user embedding $\mathbf{z}_u^r$ have two sources, namely the intra-domain user embedding $\mathbf{z}_u^r$ and cross-domain user embedding $\mathbf{p}_u^s$, corresponding to the second term of the numerator and denominator respectively in (10). This mechanism not only enables the fusion of information between domains but also implicitly increases the number of samples in contrastive learning. This is beneficial for contrastive learning, as it provides more diverse and informative samples for training [32].

The user transfer contrastive loss explicitly gathers unsupervised information about users in the collaborative domain. However, the self-supervised information of items is equally important to mining. To fully capture this self-supervised information of items, we design an item-based contrastive learning loss in the collaborative domain. Similarly, the contrastive loss of the item side can be obtained as follows:

$$\mathcal{L}_r^i = \sum_{u\in\mathcal{U}} -\log \frac{\psi(\mathbf{z}_i^r, \tilde{\mathbf{z}}_i^r)}{\sum_{v\in\mathcal{B}} \psi(\mathbf{z}_i^r, \tilde{\mathbf{z}}_i^r)}, \quad (11)$$

where $\mathbf{z}_i^r, \tilde{\mathbf{z}}_i^r$ are the first and final embeddings of user i in the interaction domain.

2) Transfer in Social Domain: The collaborative domain transfer contrastive learning automatically mines common knowledge from the social domain. However, social domain-specific information is equally important to explore. To enhance the expressiveness of the learning representations of each instance in the social domain, we design a social domain transfer contrastive learning mechanism, which transfers knowledges from the collaborative domain to the social domain.

After obtaining the user representations under the collaborative domain, they are first transferred into the space in which the contrastive loss is calculated, a process identical to collaborative domain contrastive loss calculation:

$$\mathrm{p}_u^r = W^{(2)}\sigma\left(W^{(1)}\mathrm{z}_u^r + b^{(1)}\right) + b^{(2)}, \quad (12)$$

where $W^{(\cdot)} \in \mathbb{R}^{d\times d}$ and $b^{(\cdot)} \in \mathbb{R}^{d\times 1}$ are trainable parameters. Using the same positive and negative sampling strategy for transfer contrastive learning in the social domain, we have the following contrastive loss:

$$\mathcal{L}_s = \sum_{u\in\mathcal{U}} -\log \frac{\psi(\mathbf{z}_u^s, \tilde{\mathbf{z}}_u^s) + \psi(\mathbf{z}_u^s, \mathbf{p}_u^r)}{\sum_{v\in\mathcal{B}} \psi(\mathbf{z}_u^s, \tilde{\mathbf{z}}_v^s) + \sum_{v\in\mathcal{B}} \psi(\mathbf{z}_u^s, \mathbf{p}_v^r)}, \quad (13)$$

where $\tilde{\mathbf{z}}_u^s$ is the first layer embedding of user u in the social domain. In this way, we explicitly incorporate common knowledge and domain-specific knowledge into contrastive learning to alleviate data sparsity.

C. Optimization

To combine the recommendation task with the self-supervised task, we optimize the whole method using a multi-task training strategy.

1) Primary Recommendation Task: With the final representation, we use the inner product to predict the likelihood of user u interaction with item i:

$$\hat{y}_{ui} = \mathbf{z}_u^\top \mathbf{z}_i. \quad (14)$$

To capture information directly from interactions, we employ paired Bayesian personalized ranking (BPR) loss, which is an effective recommendation ranking objective function. Specifically, the BPR loss enforces that the prediction score of an observed interaction is higher than its unobserved counterpart. Formally, the objective function of the BPR loss is as follows:

$$\mathcal{L}_{main} = \sum_{(u,i,j)\in O} -\log\sigma\left(\hat{y}_{u,i} - \hat{y}_{u,j}\right), \quad (15)$$

where $\sigma(\cdot)$ is the sigmoid function, $\mathcal{O} = \{(u,i,j) \mid (u,i) \in \mathcal{O}^+, (u,j) \in \mathcal{O}^-\}$. Here, $\mathcal{O}^+$ is the observed pairwise training data, and $\mathcal{O}^-$ denotes the unobserved interactions. In this work, we select it as the main supervised task.

2) Joint Training: To improve the recommendation performance of our proposed method for knowledge transfer, we use a joint training strategy to optimize the recommendation loss and the contrastive learning loss; moreover, to enable us to simplify the parameters of the method, we assign equal weights to $\mathcal{L}_r^u$ and $\mathcal{L}_r^s$:

$$\mathcal{L} = \mathcal{L}_{main} + \lambda_1\mathcal{L}_r + \lambda_2\mathcal{L}_s + \lambda_3\|\Theta\|_2^2, \quad (16)$$

where $\mathcal{L}_r = \mathcal{L}_r^u + \mathcal{L}_r^i$, λ_1, λ_2, and λ_3 are the hyper-parameters used to balance the contributions of various contrastive learning

TABLE II
COMPARISON OF THEORETICAL TIME COMPLEXITY

	GNN	Method
NGCF	$O((\lvert E\rvert+\lvert V\rvert)Lds\lvert E\rvert/B)$	×
LightGCN	$O(\lvert E\rvert Lds\lvert E\rvert/B)$	×
DiffNet	$O((\lvert S\rvert L+\lvert E\rvert)ds\lvert E\rvert/B)$	×
S^2-MHCN	$O((\lvert H_s\rvert+\lvert H_j\rvert+\lvert H_p\rvert+\lvert E\rvert)Lds\lvert E\rvert/B)$	GA
SEPT	$O((\lvert A_f\rvert+\lvert A_s\rvert+\rho_1\lvert E\rvert)Lds\lvert E\rvert/B)$	GA
DcRec	$O(((1+2\rho_1)\lvert E\rvert+2\rho_2\lvert S\rvert)Lds\lvert E\rvert/B)$	GA
BTGCL	$O((\lvert E\rvert+\lvert S\rvert)Lds\lvert E\rvert/B)$	FA

losses and regularization, and Θ denotes the set of GNN model parameters.

D. Discussion

1) Novelty and Differences: For social recommendation [8], [10], [20], it is important to integrate social relations into the recommender system to address the data sparsity issue. Thus, contrastive learning is utilized to capture unsupervised information in order to provide more supervisory signals. Although several methods [20], [21] adopt rudimentary contrastive learning in social recommendation, our work is distinct from them in two key ways. (1) Existing SSL methods for social recommendation [21], [23], [24], primarily rely on GA, which involves complex matrix operations. In contrast, BTGCL generates representations through FA-based graph encoders, which can reduce the time complexity of the method. (2) Most existing works [21], [22] compute the contrastive learning loss with at most two views of embeddings. However, BTGCL can aligns features from collaborative and social domains of the same positive pair to dismantle informational barriers and produce satisfactory embeddings. (3) Unlike cross-domain recommendation [38], [39], [40], our model focuses more on integrating social information between users and implicit feedback information between users and items.

2) Complexity: In this section, we delve into the theoretical temporal complexity of BTGCL, juxtaposing it with the approach employed in social recommendations. Table II provides a comparison between BTGCL and a series of graph recommendation methods in terms of time complexity. The numbers of edges and vertices of the interaction graph are $|E|$ and $|V|$, while the number of edges in the social network is represented by $|S|$. In the encoder, L represents the number of GNN layers, d denotes the embedding size, s is the number of epochs required for training, B indicates the batch size, and ρ corresponds to the edge retention rate in graph augmentation. Furthermore, A_f, A_s represent the friend and social views engendered by the SEPT method, respectively, while H_s, H_j, H_p are the triad of convolutional channels for social, joint, and purchase within the MHCN method. For BTGCL, the time complexity for completing is $O((|E|+|S|))Ld)$ in each batch. Therefore, the time complexity for one iteration is $O((|E|+|S|))Ld*|E|/B)$, and after s iterations, the total time complexity becomes$O((|E|+|S|))Lds|E|/B)$.

Algorithm 1: Framework of BTGCL.

```
Input: User feedback $\mathcal{G}_r$, Bidirectional social
       relations $\mathcal{G}_s$, randomly initialized input
       feature $Z$ of nodes.
Output: Final embedding.
1  Initialize all parameters;
2  while not converged do
3      Noise-based augmentation and graph
        convolution;
4      for each user $u$ do
5          Calculate $\mathcal{L}_r^u$ according to Eq. 10;
6          Calculate $\mathcal{L}_s$A according to Eq. 13;
7      end
8      Calculate $\mathcal{L}_r^i$ according to Eq. 11;
9      Jointly optimize the overall objective $\mathcal{L}$ in Eq.
        16;
10     Backpropagate and update parameters;
11 end
```

TABLE III
STATISTICS OF THE DATASETS

Dataset	#Users	#Items	#Interactions	#Relation	Density
Last.fm	1,892	17,632	92,834	25,434	0.278%
Douban-Book	13,024	22,347	792,062	169,150	0.272%
Yelp	19,539	21,266	450,884	363,672	0.109%
Douban-Movie	2848	39586	894,887	35,770	0.794%

IV. EXPERIMENTS

A. Datasets

To evaluate the performance of BTGCL, we conduct experiments on four benchmark datasets: Last.fm,[1] Douban-Book,[2] Yelp[3] and Douban-Movie[4], all of which have different sizes and sparsity and are publicly available. The statistics of the datasets are summarized in Table III. Following previous conventions, we set all ratings in the datasets to 1 to explore hermit feedback among users and items; moreover, for each dataset, we randomly select 80% of each user's interaction history as the training set and use the rest as the test set.

B. Baselines

In this paper, nine recent graph neural network methods are compared with the proposed method to test its effectiveness.

- **NGCF** [33]: A GNN-based collaborative filtering method that leverages both user-item interactions and auxiliary information to improve recommendation accuracy.
- **LightGCN** [34]: This model designs a simplified GCN by linearly propagating user-item embeddings to make them cleaner and generate more suitable recommendations.

[1][Online].Available: htp://files.grouplens.org/datasets/hetrec2011/
[2][Online].Available: https://github.com/librahu/HIN-Datasets-for-Recommendation-and-Network-Embedding
[3][Online].Available: htps://github.com/Coder-Yu/QRec
[4][Online].Available: htps://pan.baidu.com/s/1hrJP6rq

TABLE IV
PERFORMANCE COMPARISON FOR RECOMMENDATION METHODS AT DIFFERENT LAYERS

Method		Last.fm		Douban-Book		Yelp	
		Recall	NDCG	Recall	NDCG	Recall	NDCG
1-Layer	LightGCN	0.2624	0.2697	0.1276	0.1209	0.1003	0.0608
	SEPT	0.2669(+1.71%)	0.2745(+1.78%)	0.1365(+6.97%)	0.1287(+6.45%)	0.1056(+5.28%)	0.0645(+6.09%)
	BTGCL	**0.2842(+8.27%)**	**0.2936(+8.86%)**	**0.1594(+26.96%)**	**0.1459(+27.05%)**	**0.1270(+25.82%)**	**0.0775(+29.28%)**
2-Layer	LightGCN	0.2688	0.2763	0.1350	0.1277	0.1028	0.0629
	SEPT	0.2844(+5.80%)	0.2967(+7.38%)	0.1510(+11.85%)	0.1444(+13.08%)	0.1125(+9.44%)	0.0701(+11.45%)
	BTGCL	**0.2915(+8.59%)**	**0.3023(+9.88%)**	**0.1779(+27.93%)**	**0.1703(+37.38%)**	**0.1346(+26.85%)**	**0.0821(+31.32%)**
3-Layer	LightGCN	0.2693	0.2763	0.1376	0.1302	0.1045	0.0642
	SEPT	0.2571(-4.53%)	0.2670(-3.37%)	0.1397(+1.53%)	0.1317(+1.15%)	0.0993(-4.98%)	0.0610(-4.98%)
	BTGCL	**0.2943(+8.80%)**	**0.3043(+11.15%)**	**0.1801(+27.33%)**	**0.1723(+34.64%)**	**0.13685(+27.66%)**	**0.0832(+30.53%)**
4-Layer	LightGCN	0.2696	0.2785	0.1373	0.1307	0.1048	0.0641
	SEPT	0.2304(-14.54%)	0.2271(-18.46%)	0.1319(-3.93%)	0.1238(-5.28%)	0.0894(-14.69%)	0.0540(-15.76%)
	BTGCL	**0.2880(+6.82%)**	**0.2985(+7.18%)**	**0.1786(+28.48%)**	**0.1738(+34.43%)**	**0.1363(+24.71%)**	**0.0837(+28.24%)**

• **DiffNet** [31]: A GCN-based social recommendation method that models the recursive social diffusion process for each use, allowing simulation of how users are influenced by the recursive social diffusion process.

• **MHCN** [20]: A social recommendation method based on hypergraph convolutional networks, which uses multi-channel hypergraph convolutional networks to capture higher-order user relationships.

• **SGL** [23]: This method introduces self-supervised learning to recommendation by using graph augmentation to directly change the structure of the user-item interaction graph, drawing on the idea of contrastive learning to build SSL tasks.

• **SEPT** [21]: SEPT is designed for social recommendation, where social homogeneity is exploited to generate multiple views and tri-training is utilized to construct multiple encoders for generating self-supervised signals

• **DcRec** [22]: A state-of-the-art GA-based CL model of social recommendation, which performs disentangled contrastive learning by separately modeling user information in the collaborative and social domains.

• **SimGCL** [29]: A CL-based general recommendation model that incorporates uniform noise in the embedding space to generate multiple views without modifying the graph structure, which can improve training speed and accuracy.

• **XSimGCL** [36]: A state-of-the-art FA-based CL method of general recommendation that adds uniform noise to a single encoder to generate representations of nodes under multiple views, thus further improving training efficiency.

We discard potential baselines such as MF [41], NeuMF [4], SocialMF [8], SERec [42], etc., as previous work has shown that they are outperformed by the baselines selected here for comparison. In this study, the encoder used is LightGCN, which is a strong candidate for encoder selection due to its simplicity, its speed, and the effectiveness of its training mechanism.

C. Evaluation Metrics

To accurately measure the top-N recommendation performance of all methods, we adopt two widely used metrics [4], [43], Recall@N and NDCG@N, where N is set to 5, 10, 15, and 20 for consistency. We report the average metrics for all users in the test set and present the performance improvements for each metric as a percentage.

D. Parameter Settings

We implement our BTGCL method in PyTorch. To ensure a fair comparison, the embedding size is set to 50 and the batchsize is set to 2000 for all methods. We optimize all methods with the Adam optimizer and initialize the model parameters with the Xavier initializer [44]. We conduct a grid search for the hyperparameters: the learning rate is tuned in the range of [0.0001, 0.005], the L_2 normalization coefficient is tuned in [10^{-5}, 10^2], and the drop ratio is searched in [0.0, 0.08]. All layers L of the model are tuned in [1, 3]. For S^2-MHCN, we search for self-supervised weights β in [0.001, 0.5]. For SGL, we apply SGL-DG as the final version because it is recognized as the best model in the original paper, with self-supervised weights searched between [0.0001, 0.5] and droprate searched between [0.1, 0.9]. For SEPT, we follow the optimal hyperparameter settings of the original paper and also fine-tune them, since the results of the original paper are based on python2 but the open source code is based on python3. For SimGCL, the self-supervised weights are searched in [0.01, 0.8]. For XSimGCL, the total number of layers and the layers for comparison learning are searched in [1, 3], the data enhancement weight in [0.01, 0.8], and the temperature coefficients in [0.05, 0.5]. Moreover, an early stopping strategy is implemented: in short, if the recall@20 on the validation data does not increase in the next 30 consecutive epochs, the training process is halted. We tune the hyper-parameters of BTGCL λ_1 and λ_2 in [0.01, 1.0], and ϵ in [0.0, 0.5].

E. Overall Performance

1) Comparison With LightGCN: We first present the performances of LightGCN, SEPT, and BTGCL with different layers, as detailed in Table IV. The improvements are based on the

TABLE V
PERFORMANCE COMPARISON OF DIFFERENT RECOMMENDATION METHODS WITH VARYING TOP-N ON FOUR DATASETS

Datasets	Metrics	NGCF	LightGCN	DiffNet	MHCN	SGL	SEPT	DcRec	SimGCL	XSimGCL	BTGCL	Imp
Last.fm	R@5	0.1182	0.1248	0.1155	0.1332	0.1342	0.1333	0.1356	0.1348	0.1358	**0.1390***	+2.36%
	N@5	0.2672	0.2853	0.2600	0.3045	0.3066	0.3053	0.3058	0.3075	0.3073	**0.3143***	+2.21%
	R@10	0.1802	0.1878	0.1770	0.1970	0.1995	0.1992	0.2035	0.2005	0.2027	**0.2084***	+2.41%
	N@10	0.2200	0.2322	0.2148	0.2478	0.2480	0.2475	0.2483	0.2486	0.2480	**0.2551***	+2.61%
	R@15	0.2243	0.2327	0.2199	0.2413	0.2460	0.2461	0.2477	0.2450	0.2453	**0.2560***	+3.35%
	N@15	0.2465	0.2590	0.2406	0.2743	0.2756	0.2755	0.2746	0.2753	0.2731	**0.2832***	+2.87%
	R@20	0.2583	0.2696	0.2544	0.2773	0.2820	0.2834	0.2844	0.2808	0.2810	**0.2943***	+3.48%
	N@20	0.2646	0.2785	0.2590	0.2929	0.2946	0.2933	0.2937	0.2942	0.2919	**0.3043***	+3.43%
Douban-Book	R@5	0.0503	0.0569	0.0538	0.0685	0.0772	0.0653	0.0776	0.0784	0.0789	**0.0845***	+7.10%
	N@5	0.0985	0.1262	0.1063	0.1482	0.1657	0.1419	0.1677	0.1717	0.1665	**0.1758***	+2.39%
	R@10	0.0813	0.0900	0.0845	0.1029	0.1153	0.1017	0.1193	0.1162	0.1191	**0.1250***	+4.78%
	N@10	0.1015	0.1238	0.1068	0.1427	0.1596	0.1385	0.1628	0.1639	0.1612	**0.1690***	+3.11%
	R@15	0.1055	0.1159	0.1077	0.1293	0.1439	0.1287	0.1469	0.1445	0.1479	**0.1547***	+4.60%
	N@15	0.1061	0.1264	0.1102	0.1441	0.1609	0.1405	0.1632	0.1644	0.1627	**0.1696***	+3.16%
	R@20	0.1261	0.1376	0.1272	0.1504	0.1667	0.1510	0.1693	0.1703	0.1707	**0.1794***	+5.10%
	N@20	0.1112	0.1302	0.1144	0.1473	0.1642	0.1444	0.1665	0.1669	0.1660	**0.1729***	+3.60%
Yelp	R@5	0.0329	0.0392	0.0313	0.0463	0.0506	0.0440	0.0523	0.0519	0.0534	**0.0557***	+4.31%
	N@5	0.0343	0.0435	0.0331	0.0514	0.0551	0.0486	0.0573	0.0565	0.0580	**0.0584***	+0.69%
	R@10	0.0554	0.0643	0.0535	0.0749	0.0803	0.0713	0.0834	0.0821	0.0831	**0.0876***	+5.04%
	N@10	0.0419	0.0512	0.0406	0.0601	0.0642	0.0570	0.0667	0.0659	0.0672	**0.0679***	+1.04%
	R@15	0.0740	0.0858	0.0715	0.0973	0.1046	0.0934	0.1072	0.1064	0.1076	**0.1142***	+6.13%
	N@15	0.0481	0.0583	0.0466	0.0673	0.0720	0.0641	0.0745	0.0737	0.0752	**0.0764***	+1.60%
	R@20	0.0903	0.1048	0.0873	0.1168	0.1248	0.1125	0.1287	0.1271	0.1283	**0.1369***	+6.37%
	N@20	0.0533	0.0641	0.0516	0.0734	0.0783	0.0701	0.0810	0.0801	0.0816	**0.0832***	+1.96%
Douban-Movie	R@5	0.0266	0.0296	0.0275	0.0298	0.0340	0.0322	0.0336	0.0388	0.0382	**0.0429***	+10.57%
	N@5	0.2434	0.2649	0.2169	0.2560	0.2911	0.2839	0.2861	0.3061	0.3108	**0.3137***	+0.93%
	R@10	0.0507	0.0533	0.0473	0.0541	0.0593	0.0574	0.0582	0.0654	0.0658	**0.0723***	+9.87%
	N@10	0.2315	0.2491	0.2049	0.2409	0.2703	0.2672	0.2651	0.2823	0.2883	**0.2916***	+1.14%
	R@15	0.0685	0.0707	0.0649	0.0728	0.0794	0.0764	0.0788	0.0886	0.0863	**0.0947***	+6.88%
	N@15	0.2215	0.2384	0.1982	0.2307	0.2580	0.2549	0.2542	0.2686	0.2742	**0.2775***	+1.20%
	R@20	0.0830	0.0883	0.0816	0.0909	0.1000	0.0932	0.0985	0.1059	0.1074	**0.1142***	+6.33%
	N@20	0.2161	0.2331	0.1944	0.2251	0.2508	0.2466	0.2471	0.2603	0.2664	**0.2695***	+1.16%

The symbol * denotes that the improvement is significant with a p-value < 0.001 based on a two-tailed paired t-test. R@K represents Recall@K, and N@K represents NDCG@K.

performance of LightGCN; the figures representing the best performance are indicated in bold.

• In the vast majority of cases, SEPT and BTGCL can largely outperform LightGCN. The suboptimal performance of LightGCN might be due to the sparsity of supervisory signals in these datasets (which contain a large number of inactive users) indicating the superiority of supplementing the recommendation task with self-supervised learning.

• SEPT outperforms LightGCN with 1-layer and 3-layer settings, and gets suboptimal performance with a depth of 3 to 4. One possible reason is that SEPT utilizes only the user's self-supervised information, and increasing the number of layers tends to result in oversmoothing problems, the optimized methods tend to treat all nodes in the graph equally.

• BTGCL achieves the best performance in almost all cases, which demonstrates the effectiveness of the bi-directional migration-based contrastive learning. In particular, it performs significantly better than the self-supervised SEPT-based method on the huge and sparse datasets Douban-Book and Yelp.

2) Comparison With State-of-The-Art Methods: To further demonstrate the outstanding performance of BTGCL, we compare it with nine recent GNN-based or CL-based recommendation methods. Examining the results in Table V, we can make several observations:

• Out of the GNN-based methods, such as NGCF, LightGCN, and Diffnet, LightGCN performs best on all datasets, which reflects the effectiveness of the simplified construct [34]. Diffnet performs worse than LightGCN, probably due to the neglect of the latent collaborative signals of users.

• Compared to the above methods, CL-based methods are consistently superior to supervised methods on all datasets, demonstrating the competitiveness of contrastive learning in

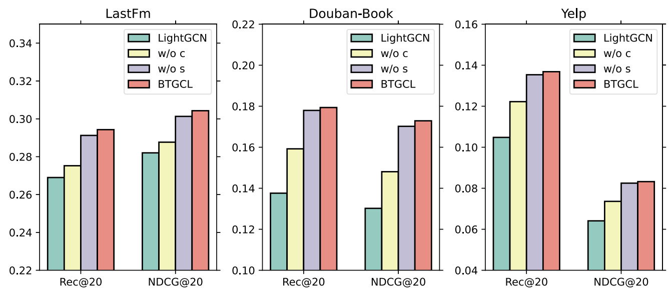


Fig. 4. Performance comparison between transfer in collaborative domain and transfer in social domain.

TABLE VI
PERFORMANCE COMPARISON BETWEEN DIFFERENT VARIANTS

	Last.fm		Douban-Book		Yelp	
	Recall	NDCG	Recall	NDCG	Recall	NDCG
LightGCN	0.2696	0.2785	0.1376	0.1302	0.1048	0.0641
w/o t-cs	0.2881	0.2998	0.1720	0.1748	0.1325	0.0837
w/o t-c	0.2899	0.3009	0.1733	0.1743	0.1321	0.0829
w/o t-s	0.2931	0.3039	0.1750	0.1742	0.1326	0.0835
BTGCL	0.2943	0.3043	0.1794	0.1729	0.1369	0.0832

improving recommendation performance. Moreover, SGL performs better than MHCN and SEPT on most datasets, proving the effectiveness of graph-based augmentation methods [30]. DcRec outperforms over SGL, showing the great potential of latent relationships between users in social domains for recommendation. In addition, SimGCL and XSimGCL have similar performance, and both perform better than SGL, indicating the strong competitiveness of feature enhancement.

• Finally, we can see that the proposed BTGCL has a significant advantage on all datasets. This improvement is brought about by the bi-directional transfer of knowledge contrastive learning. In particular, BTGCL has a prominent advantage over baselines on the three larger datasets, Douban-Book, Yelp and Douban-Movie. Furthermore, we perform a significance test and obtain a p-value < 0.05, indicating that the performance improvement of BTGCL over the strongest baseline is statistically significant on the three datasets.

F. Ablation Study

In BTGCL, we build the transfer contrastive learning methodology in two domains. It is not clear which part has a stronger impact on the recommendation performance. Accordingly, in this section, we conduct an ablation study to compare the contributions of the transfer mechanism in the collaborative domain and in the social domain. The results are shown in Fig. 4; here, ”w/o c” and ”w/o s” represent the variants obtained by removing collaborative domain transfer and social domain transfer, respectively. From the figure, we can observe that removing each part leads to a decrease in performance, while both variants achieve better performance than baseline LightGCN. This indicates that self-supervised signals in both domains have a beneficial effect on improving the performance of graph collaborative filtering. In addition, the contribution of the transfer mechanism in the collaborative domain is more prominent in comparison, which is likely due to the higher quality of interaction signals in the collaborative domain than social signals in the social domain.

To further verify the effectiveness of our transfer mechanism, we ablate the mechanism to investigate the contribution of each part. In Table VI, “w/o t-c” and “w/o t-s” respectively indicate that the direction transfer mechanism is detached from the collaborative and social domains. “w/o t-cs” means that only intra-domain contrastive learning is used. From Table VI, we can observe that all elements of the transfer mechanism make contributions on all three datasets. Moreover, when the knowledge transfer mechanism is removed, BTGCL shows lower performance but still outperforms LightGCN. Furthermore, the importance of the transfer from the collaborative domain is generally higher than the transfer from the social domain. We speculate that the domain-specific information of users in collaborative domains is more important, because the collaborative domain encoder models the higher-order connectivity of users and items.

G. Training Efficiency Comparison

As analyzed above, BTGCL is lighter than most CL-based social recommendation methods. In this section, we compare the actual training time of different methods on three datasets, which is as informative as the theoretical analysis. The reported figures are collected on a workstation with a GeForce RTX 3090 GPU and an Intel i9-12900 k CPU (64 G memory). These methods are implemented with Tensorflow 1.15, and a 3-layer setting is applied to all. Based on the results in Fig. 6, we can make the following observations:

• LightGCN has the shortest training time per epoch because of its simple structure. In contrast, S^2-MHCN has the longest training time, as hypergraph computation takes almost ten times longer than LightGCN. DcRec ranks second as a result of excessive contrastive loss computation, taking eight times as long as LightGCN. The training time of SEPT is four times longer than LightGCN since it requires constructing additional perspectives. BTGCL has only a slightly higher training cost than LightGCN, as it only performs message drop without a graph structure perturbation operation.

• LightGCN requires the largest number of epochs, nearly an order of magnitude more than the rest of the other methods, resulting in a higher total time overhead. SEPT needs slightly less training time than LightGCN because it only captures user information and not the item’s self-supervised information. Although DcRec’s single epoch training time is less than that of MHCN, their total training time is similar due to the higher number of epochs required, which is the highest among the baselines. BTGCL requires a similar number of epochs as SEPT and MHCN; notably, however, its total training time is the lowest among all methods because of its lightweight structure.

We can draw several key conclusions from these observations. Compared to LightGCN, CL can reduce the number of epochs

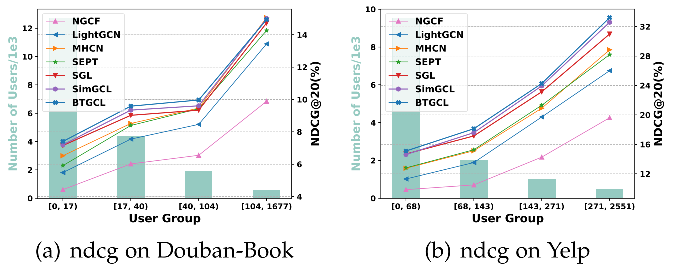


Fig. 5. Performance comparison over the sparsity distribution of user groups on different datasets.

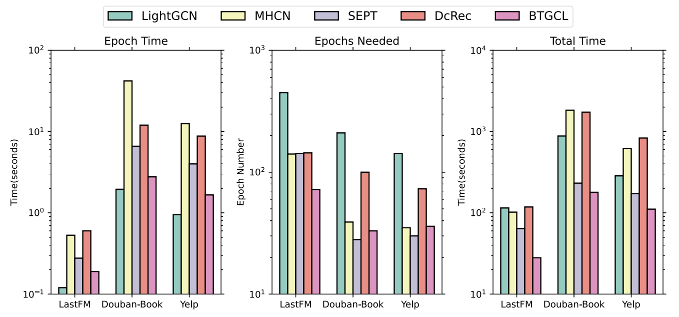


Fig. 6. Comparison of BTGCL and other methods for Epoch Time, Epochs Needed and Total Time metrics on three datasets.

needed for training. Moreover, inappropriate graph augmentation methods and agent tasks may result in more total training time overhead. Finally, the bi-directional transfer mechanism can not only significantly reduce training time but also improve performance; this is because the mechanism increases the number of contrastive learning samples, provides more supervised signals, and increases the convergence speed.

H. Impact of Data Sparsity Levels

In order to verify whether BTGCL can alleviate the sparsity of interaction data mentioned in the introduction section, we test its performance for users with different degrees of sparsity. Specifically, we split all users into four groups based on the number of user interactions while keeping the total number of interactions for each group the same. Note that since the user interactions in the Last.fm dataset are mostly the same (with 50 interactions), we choose to conduct experiments on the Douban-Book and Yelp datasets. Taking the Douban-Book dataset as an example, the number of user interactions in each group is less than 17, 40, 104, and 1677, respectively. The results are presented in Fig. 5.

From the figure, we find that the performance of BTGCL is better than all baselines in all user groups, and moreover that the improvement brought by BTGCL in the group with a low number of interactions is higher. These results show that BTGCL can learn higher quality representations from sparse data, which is due to the bi-directional transfer mechanism providing more supervisory signals.

I. Parameter Sensitivity Analysis

In this section, we focus on the effect of the three hyperparameters on the performance of the BTGCL method: λ_1 and λ_2 are the adjustment coefficients of the transfer contrastive learning mechanism in the collaborative domain and social domain, respectively. ϵ controls the strength of the feature augmentation. Specifically, this section describes the use of grid search to select the optimal values of the three hyperparameters. The experimental results are presented in Figs. 7 and 8.

We fix $\epsilon = 0.2$ and conduct experiments with a 3-layer setting. Fig. 7 shows the sensitivity of λ_1 and λ_2 for BTGCL. We explore the sensitivity on the three datasets and alter λ_1 and λ_2 among {0.01, 0.03, 0.05, 0.1, 0.2, 0.5, 1.0} and {0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 1.0}, respectively. As we can see, the

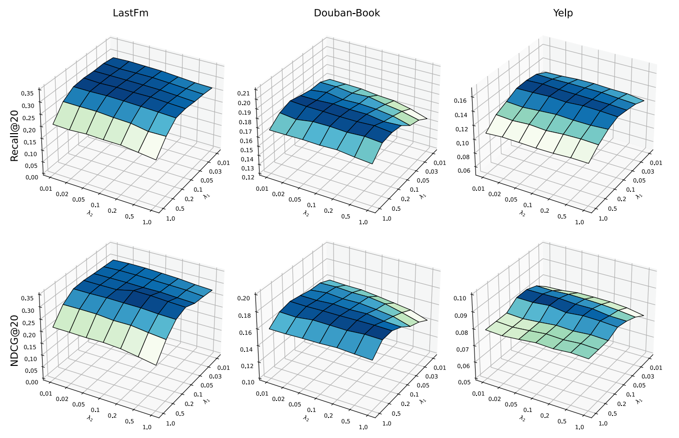


Fig. 7. Performance analysis of BTGCL with parameters λ_1 and λ_2 on three datasets. The top shows the Recall@20 results and the bottom shows the NDCG@20 results.

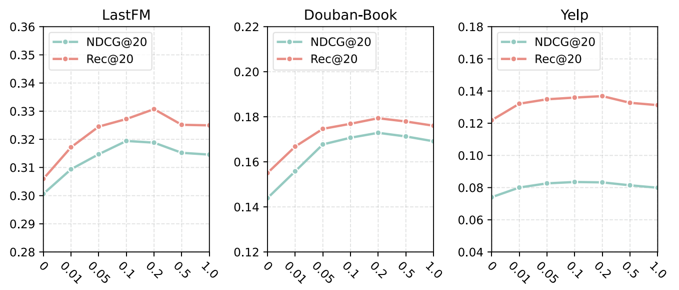


Fig. 8. Performance analysis of BTGCL for parameter ϵ on three datasets.

accuracy of BTGCL is relatively smooth when the parameters are within certain ranges. However, extremely large values of λ_1 and λ_2 result in low performances on all datasets and should be avoided in practice. Moreover, increasing λ_1 and λ_2 from 0.01 to 0.2 improves the accuracy on all datasets, demonstrating the effectiveness of the proposed aggregation mechanism. In addition, BTGCL is more sensitive to λ_1; this phenomenon can be explained by Fig. 4.

To study the effect of feature augmentation strength, we try different combinations of ϵ with the set {0, 0.01, 0.05, 0.1, 0.2, 0.5, 1.0}. As shown in Fig. 8, on all the datasets, BTGCL reaches its best performance when ϵ is set to 0.2, and is more sensitive

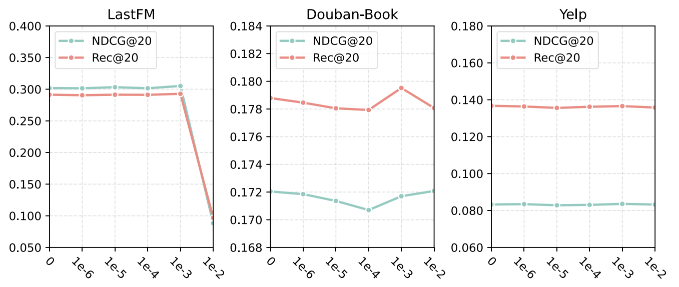


Fig. 9. Performance of 3-layer BTGCL w.r.t. different regularization coefficient λ_3 on Last.fm, Douban-Book and Yelp.

on the LastFM and Douban-Book datasets. Moreover, ϵ should be neither too large nor too small to optimize the performance of BTGCL.

Furthermore, we demonstrate the impact of the hyperparameter L_2 regularization coefficient λ_3 on the performance of BTGCL. As shown in Fig. 9, BTGCL is relatively insensitive to λ_3. Even when λ_3 is set to 0, BTGCL still outperforms the baselines. This indicates that BTGCL is not prone to overfitting because the only trainable hyperparameters in BTGCL are the embeddings of users and items in the 0-th layer, and the social graph encoder and interaction graph encoder share the initial user embeddings. This makes the entire model easy to train and regularize. The optimal parameters on the three datasets can be set to $1e^{-3}$. In the case of the Last.fm dataset, when λ_3 is larger than $1e^{-3}$, the performance rapidly declines. This may be due to the small size of the Last.fm dataset, indicating that overly strong regularization can have a negative impact on the normal training of the model.

V. Related Work

A. GNN-Based Recommendation

Graph neural networks [12], [45] have shown great potential in the field of recommender systems due to their ability to model complex relationships in graph-structured data. Several GNN-based methods have been proposed to address various challenges in the field of recommendation, such as NGCF [33] and LightGCN [34]. The basic idea of these GCN-based methods is to use the neighbor information in the user-item graph to refine the embeddings of the target nodes by aggregating the embeddings of the neighbors [46], [47]. Due to its powerful graph data modeling capabilities, GNNs have also achieved extraordinary success on some specific graphs, such as social recommendation. GraphRec [13] is the first approach to introduce GNN into social recommendation, modeling user-item and user-user interactions as graph data. DGRec [48] proposes an online recommendation method based on dynamic graph attention neural networks. ESRF [49] designs a deep GCN-based adversarial framework to address problems such as data sparsity and noise in social recommendation. In summary, these works share the common idea of using multiple graph neural layers to capture more user-user and user-item information.

B. Self-Supervised Recommendation

Following the success of self-supervised learning [19], [50] in computer vision contexts [25], self-supervised learning has been widely applied in fields including natural language processing, speech processing, recommender systems, and graph representation learning. As CL is capable of handling large volumes of unlabeled data, it is a competitive solution to the data sparsity problem in the field of recommender systems [55]. Due to the promising performance of SSL in other fields, several recent works [21], [22], [23], [30], [51] have applied CL to recommendation. S^3-Rec [52] devises auxiliary self-supervised objectives to pre-train sequence recommendation models by randomly masking features of items, skipping items, and subsequences of a given sequence. SEPT [21] proposes a socially aware SSL framework that integrates triple training to capture supervisory signals not only from the ego, but also from other nodes. GRACE [53] proposes a framework for node-level graph comparison learning and performs disruption by de-edging discarding and masking node features. NCL [30] designs a prototypical comparison objective to capture the correlation between a user/item and its context. CLRec [54] applies CL to mitigate exposure bias in recommendations and improve the fairness and efficiency of deep matching.

VI. Conclusion and Future Work

In this article, we demonstrate the shortcomings of current contrastive learning models for social recommendations and propose a new contrastive learning paradigm, called Bidirectional Transfer Graph Contrastive Learning for Social Recommendation (BTGCL), which captures potential self-supervised signals from both social and interactive perspectives into contrastive learning for graph collaborative filtering. It achieves better user/item representation learning in two dimensions: (1) BTGCL integrates the social and interactive perspectives for comparative learning, and captures the self-supervised information within each perspective while mining the rarely utilized cross-view user features and structural information. (2) BTGCL leverages the semantic differences between cross-domain users to incorporate a larger set of positive and negative samples into the contrastive learning target. This expanded sample contrastive learning paradigm enhances user representations. Extensive experiments conducted on three public datasets demonstrate the effectiveness of the proposed BTGCL.

As future work, we will extend our framework to other recommendation tasks, such as cross-domain recommendations and sequence recommendations. In addition, we consider developing a more refined approach that can exploit and use the self-supervised information of items through their multimodality and heterogeneous graph techniques.

References

[1] F. Ricci, L. Rokach, and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*. Berlin, Germany: Springer, 2010, pp. 1–35.

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

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

[4] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 173–182.

[5] N. E. Friedkin, *A Structural Theory of Social Influence*. Cambridge, U.K.: Cambridge Univ. Press, 1998.

[6] R. B. Cialdini and N. J. Goldstein, "Social influence: Compliance and conformity," *Annu. Rev. Psychol.*, vol. 55, pp. 591–621, 2004.

[7] A. Anagnostopoulos, R. Kumar, and M. Mahdian, "Influence and correlation in social networks," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 7–15.

[8] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proc. 4th ACM Conf. Recommender Syst.*, 2010, pp. 135–142.

[9] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proc. fourth ACM Int. Conf. Web Search Data Mining*, 2011, pp. 287–296.

[10] M. Jiang, P. Cui, F. Wang, W. Zhu, and S. Yang, "Scalable recommendation with social contextual information," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 11, pp. 2789–2802, Nov. 2014.

[11] G. Guo, J. Zhang, and N. Yorke-Smith, "A novel recommendation model regularized with user trust and item ratings," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 7, pp. 1607–1620, Jul. 2016.

[12] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

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

[14] L. Wu, P. Sun, Y. Fu, R. Hong, X. Wang, and M. Wang, "A neural influence diffusion model for social recommendation," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 235–244.

[15] S. Liu, I. Ounis, C. Macdonald, and Z. Meng, "A heterogeneous graph neural model for cold-start recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 2029–2032.

[16] L. Wu, J. Li, P. Sun, R. Hong, Y. Ge, and M. Wang, "DiffNet: A neural influence and interest diffusion network for social recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 10, pp. 4753–4766, Oct. 2022.

[17] R. He and J. McAuley, "Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering," in *Proc. 25th Int. Conf. World Wide Web*, 2016, pp. 507–517.

[18] W. Wang, F. Feng, X. He, L. Nie, and T.-S. Chua, "Denoising implicit feedback for recommendation," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, 2021, pp. 373–381.

[19] X. Liu et al., "Self-supervised learning: Generative or contrastive," *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 857–876, Jan. 2023.

[20] J. Yu, H. Yin, J. Li, Q. Wang, N. Q. V. Hung, and X. Zhang, "Self-supervised multi-channel hypergraph convolutional network for social recommendation," in *Proc. Web Conf.*, 2021, pp. 413–424.

[21] J. Yu, H. Yin, M. Gao, X. Xia, X. Zhang, and N. Q. Viet Hung, "Socially-aware self-supervised tri-training for recommendation," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 2084–2092.

[22] J. Wu, W. Fan, J. Chen, S. Liu, Q. Li, and K. Tang, "Disentangled contrastive learning for social recommendation," in *Proc. 31st ACM Int. Conf. Inf. Knowl. Manage.*, 2022, pp. 4570–4574.

[23] J. Wu et al., "Self-supervised graph learning for recommendation," in *Proc. 44th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2021, pp. 726–735.

[24] X. Cai, C. Huang, L. Xia, and X. Ren, "LightGCL: Simple yet effective graph contrastive learning for recommendation," in *Proc. 11th Int. Conf. Learn. Representations*, 2023, pp. 1–15.

[25] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[26] Y. You, T. Chen, Z. Wang, and Y. Shen, "Bringing your own view: Graph contrastive learning without prefabricated data augmentations," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, 2022, pp. 1300–1309.

[27] F. Wang and H. Liu, "Understanding the behaviour of contrastive loss," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 2495–2504.

[28] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9929–9939.

[29] J. Yu, H. Yin, X. Xia, T. Chen, L. Cui, and Q. V. H. Nguyen, "Are graph augmentations necessary? Simple graph contrastive learning for recommendation," in*Proc. 45th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2022, pp. 1294–1303.

[30] Z. Lin, C. Tian, Y. Hou, and W. X. Zhao, "Improving graph collaborative filtering with neighborhood-enriched contrastive learning," in *Proc. ACM Web Conf.*2022, pp. 2320–2329.

[31] B. Hu, N. Zhou, Q. Zhou, X. Wang, and W. Liu, "DiffNet: A learning to compare deep network for product recognition," *IEEE Access*, vol. 8, pp. 19336–19344, 2020.

[32] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9729–9738.

[33] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2019, pp. 165–174.

[34] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "LightGCN: Simplifying and powering graph convolution network for recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2020, pp. 639–648.

[35] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annu. Rev. Sociol.*, vol. 27, no. 1, pp. 415–444, 2001.

[36] J. Yu, X. Xia, T. Chen, L. Cui, N. Q. V. Hung, and H. Yin, "XSimGCL: Towards extremely simple graph contrastive learning for recommendation," 2022, *arXiv:2209.02544*.

[37] Y. Zhang, H. Zhu, Z. Song, P. Koniusz, and I. King, "Costa: Covariance-preserving feature augmentation for graph contrastive learning," in *Proc. 28th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2022, pp. 2524–2534.

[38] M. Liu, J. Li, G. Li, and P. Pan, "Cross domain recommendation via bi-directional transfer graph collaborative filtering networks," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 885–894.

[39] C. Zhao, C. Li, and C. Fu, "Cross-domain recommendation via preference propagation GraphNet," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, 2019, pp. 2165–2168.

[40] C. Gao et al., "Cross-domain recommendation without sharing user-relevant data," in *Proc. World Wide Web Conf.*, 2019, pp. 491–502.

[41] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," 2012, *arXiv:1205.2618*.

[42] M. Wang, X. Zheng, Y. Yang, and K. Zhang, "Collaborative filtering with social exposure: A modular approach to social recommendation," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 2516–2523.

[43] J.-H. Yang, C.-M. Chen, C.-J. Wang, and M.-F. Tsai, "Hop-rec: High-order proximity for implicit recommendation," in *Proc. 12th ACM Conf. Recommender Syst.*, 2018, pp. 140–144.

[44] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[45] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1024–1034.

[46] Y. Zhang, Y. Zhang, D. Yan, S. Deng, and Y. Yang, "Revisiting graph-based recommender systems from the perspective of variational auto-encoder," *ACM Trans. Inf. Syst.*, vol. 41, no. 3, pp. 1–28, 2023.

[47] S. Wu, F. Sun, W. Zhang, X. Xie, and B. Cui, "Graph neural networks in recommender systems: A survey," *ACM Comput. Surv.*, vol. 55, no. 5, pp. 1–37, 2022.

[48] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, 2019, pp. 555–563.

[49] J. Yu, H. Yin, J. Li, M. Gao, Z. Huang, and L. Cui, "Enhancing social recommendation with adversarial graph convolutional networks," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 8, pp. 3727–3739, Aug. 2022.

[50] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, 2020, Art. no. 2.

[51] Y. Wei et al., "Contrastive learning for cold-start recommendation," in *Proc. 29th ACM Int. Conf. Multimedia*, 2021, pp. 5382–5390.

[52] K. Zhou et al., "S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, 2020, pp. 1893–1902.

[53] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Deep graph contrastive representation learning," 2020, *arXiv: 2006.04131*.

[54] C. Zhou, J. Ma, J. Zhang, J. Zhou, and H. Yang, "Contrastive learning for debiased candidate generation in large-scale recommender systems," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discov. Data Mining*, 2021, pp. 3985–3995.

[55] T. Wang, L. Xia, and C. Huang, "Denoised self-augmented learning for social recommendation," in *Proc. 32nd Int. Joint Conf. Artif. Intell.*, 2023, pp. 2324–2331.

Lei Sang received the PhD degree from the Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, Australia, in 2021. He is currently a lecturer with the School of Computer Science and Technology, Anhui University, Anhui, China. His current research interests include natural language processing, data mining, and recommendation systems.

Mingyuan Liu received the bachelor's degree from the School of Physics and Materials Science, Anhui University, Hefei, China, in 2020. Now, he is currently working toward the master's degree with the School of Computer Science and Technology, Anhui University. His current research interests include graph neural network, personalized recommender systems, and self-supervised learning.

Yi Zhang received the bachelor's degree in computer science and technology from Anhui University, Hefei, China, in 2020, where he is currently working toward the PhD degree with the School of Computer Science and Technology. His current research interests include graph neural network, personalized recommender systems, and service computing.

Yuee Huang received the PhD degree in public health from the School of Earth and Environment, Anhui University of Science & Technology, in 2016. She is currently a full professor and a master instructor with the School of Public Health at Wannan Medical College, and the director of academic affairs office and the academic and technical leader of Wannan Medical College. Her research interests include maternal,child and adolescent health, epidemiology and health statistics, etc. Please see more information in our website https://ph.wnmc.edu.cn/.

Yiwen Zhang received the PhD degree in management science and engineering from the Hefei University of Technology, in 2013. He is currently a full professor with the School of Computer Science and Technology, Anhui University. He has published more than 70 papers in highly regarded conferences and journals, including *IEEE Transactions on Knowledge and Data Engineering*, *IEEE Transactions on Mobile Computing*, *IEEE Transactions on Services Computing*, *ACM Transactions on Information Systems*, *IEEE Transactions on Neural Networks and Learning Systems*, *ACM Transactions on Knowledge Discovery from Data*, ICSOC, ICWS, etc. His research interests include service computing, cloud computing, and Big Data analytics. Please see more information in our website http://bigdata.ahu.edu.cn/.