Article

SSGCL: Simple Social Recommendation with Graph Contrastive Learning

Zhihua Duan [1], Chun Wang [1,*] and Wending Zhong [2]

[1] Faculty of Data Science, City University of Macau, Macau 999078, China; d22091101472@cityu.edu.mo
[2] Faculty of Applied Sciences, Macao Polytechnic University, Macau 999078, China; p2209473@mpu.edu.mo
* Correspondence: chunwang@cityu.edu.mo

Abstract: As user–item interaction information is typically limited, collaborative filtering (CF)-based recommender systems often suffer from the data sparsity issue. To address this issue, recent recommender systems have turned to graph neural networks (GNNs) due to their superior performance in capturing high-order relationships. Furthermore, some of these GNN-based recommendation models also attempt to incorporate other information. They either extract self-supervised signals to mitigate the data sparsity problem or employ social information to assist with learning better representations under a social recommendation setting. However, only a few methods can take full advantage of these different aspects of information. Based on some testing, we believe most of these methods are complex and redundantly designed, which may lead to sub-optimal results. In this paper, we propose SSGCL, which is a recommendation system model that utilizes both social information and self-supervised information. We design a GNN-based propagation strategy that integrates social information with interest information in a simple yet effective way to learn user–item representations for recommendations. In addition, a specially designed contrastive learning module is employed to take advantage of the self-supervised signals for a better user–item representation distribution. The contrastive learning module is jointly optimized with the recommendation module to benefit the final recommendation result. Experiments on several benchmark data sets demonstrate the significant improvement in performance achieved by our model when compared with baseline models.

Keywords: recommendation system; collaborative filtering; social recommendation; contrastive learning; graph neural networks

MSC: 68T07

Citation: Duan, Z.; Wang, C.; Zhong, W. SSGCL: Simple Social Recommendation with Graph Contrastive Learning. *Mathematics* **2024**, *12*, 1107. https://doi.org/10.3390/math12071107

Academic Editor: Andrea Scozzari

Received: 7 March 2024
Revised: 3 April 2024
Accepted: 5 April 2024
Published: 7 April 2024

1. Introduction

With the rapid evolution of internet technology, users are inundated with exponential amounts of information every day. This leads to a situation where, although users seem to receive a greater volume of it, it becomes challenging to extract useful information from the large volume of redundant data. Recommendation systems play a crucial role in addressing this issue. At present, recommendation systems have already become a common solution to provide information filtering and prediction in various domains, such as social networks and short videos [1–3]. The critical function of recommendation systems is to provide users with items they may find interesting while avoiding recommending items that they do not, thereby saving them time and effort while boosting the revenue of various companies.

Collaborative filtering (CF) is one of the most popular technologies for recommendation which predicts users' interest in items by analyzing collaborative behaviors between users [4]. Many of the early CF models are based on matrix factorization (MF) [5–11]. These methods decompose the user–item matrix into two lower-dimensional latent representation matrices to capture the features of users and items. However, due to the inherent sparsity of data (a user is usually associated with only a few items, resulting in very few non-zero entries in the matrix), models based on MF often struggle to effectively capture feature embeddings for users and items.

To address this issue, some studies have aimed to involve social information and perform social recommendations. These methods are based on the idea that users in a social network may influence each other and share their preferences toward items. Therefore, the user–user interactions in the social network are worth exploring and are usually jointly learned with the user–item interactions in the network of interest. Through introducing social information, the sparsity issue is mitigated, and the efficiency and effectiveness of recommendations are enhanced.

In recent years, graph neural networks (GNNs) have emerged as effective methods for handling networked data. To learn better feature embedding, GNNs have also been successfully introduced into the recommendation task [12,13]. One of the reasons for the popularity of GNNs is their ability to capture high-order connectivity [14]. In GNN-based social recommendation systems, the user–item interactions form a bipartite graph, allowing for users to have two types of neighbors: directly associated items and two-step neighbor users sharing the same preferences. The user–user social interactions can also be easily turned into a graph. This property enables GNN-based recommendation system models [15–17] to extract collaboration signals from numerous interactions and obtain powerful node representations from high-order neighbor information.

Furthermore, some GNN-based recommendation systems have also aimed to employ self-supervision signals from autoencoders [18–20] or contrastive learning-based approaches [21,22]. Contrastive learning typically constructs pairs of positive and negative samples through data augmentation methods and then forces positive samples closer to and negative samples further from each other, thereby maximizing the discrimination between positive and negative pairs for representation learning. The contrastive loss is usually jointly optimized with the GNN loss to manipulate the embedding learning.

Although existing GNN-based social recommendation systems have combined social, interest, and even self-supervised information, achieving satisfactory results, some problems still limit the performance of these algorithms.

Sparse and unbalanced data sets: In practical applications, there is often a disparity in the number of users and items in the graph or the sparsity of the data set. Taking the Yelp data set as an example, the number of items is more than twice the number of users. This means that information related to a single item is sparser than that related to a user. Therefore, achieving the same high quality in learning item representations as in user representations is challenging. The inclusion of social information has made this inclination even worse. Unbalanced user and item representation learning will, without doubt, degrade the recommendation performance.

Complex and inefficient augmentation: Many social recommendation systems that incorporate graph neural networks (GNNs) tend to employ overly complex graph augmentation methods for contrastive learning, such as randomly dropping nodes or edges in the graph [23]. Research has shown that this type of operation can alter the original structure of the graph, leading to changes in the semantics of the user–item interactions [24]. Existing social recommendation models lack direct and simple ways of extracting information from the interactions. This dramatically affects the effectiveness and efficiency of the social recommendation.

Motivated by these observations, we address the following challenges in performing satisfactory social recommendations:

- In social recommendation, we have both social and interest information, which is sparse and unbalanced. How can we effectively learn useful user and item embedding from such complex information?
- Contrastive learning is a promising path toward better recommendation performance. How can we design a practical graph augmentation approach for contrastive learning?

To deal with the first challenge, we propose a diffusion module to effectively learn user and item embedding. Instead of the traditional GCN [25], a more straightforward

GNN-based strategy is designed to perform influence diffusion and learn informative user and item representation separately from the social and interest graph. Instead of aggregating and updating the user embedding learned from the interest and social graph at each GNN layer, we choose to update the user embedding from the two graphs separately by themselves and simply sum up the final embedding from the two networks. We argue that this method sufficiently extracts information related to both users and items without disturbing each other and obtains better embedding.

For the second challenge, we abandon complex graph enhancement methods and simply generate positive and negative samples for self-supervision by adding noise to the embedding. The experimental results suggest that such graph augmentation improves the model performance without compromising the original semantics of the graph. In such a simple way, our contrastive learning component can manipulate the embedding learning toward a smoother distribution, thereby improving the recommendation performance.

To summarize, in this paper, we propose a model called SSCGL specifically designed for social recommendation. SSCGL consists of three main modules: a diffusion module, a contrastive learning module, and a prediction module. The diffusion module learns informative user and item embedding separately from the social and interest information with a specially designed GNN-based architecture. Then, a contrastive learning module that generates contrastive views by adding noise to the graph is incorporated to further manipulate the embedding learning. Finally, the contrastive learning and the recommendation are jointly optimized to predict the final recommendation results. Experimental results on real-world social recommendation data sets validate the effectiveness of our designs.

Our work significantly contributes in the following aspects:

- We successfully integrate social information and self-supervised signals into the recommendation system, significantly improving the performance.
- While existing social recommendation works tend to design complex attention mechanisms to fuse the social and interest information during the GNN propagation, we abandon traditional GCN encoding and opt for a more straightforward encoder for layer-wise embedding updating. The representations from the interest and social networks are separately updated and, at last, combined with a simple sum. Our design demonstrates simple and powerful acquisition of better item and user representations.
- We abandon complex graph enhancement methods commonly used in traditional contrastive learning approaches and employ a simpler and more effective approach by adding noise to embeddings, further improving the model's performance.
- We evaluate our model on three real-world publicly available data sets. Our SSGCL demonstrates superior performance, compared with eight state-of-the-art baselines, in terms of social recommendations.

2. Related Work

In this section, we provide a brief overview of recommendation systems based on different kinds of approaches.

2.1. Matrix Factorization (MF)-Based Recommendation Systems

In the current field of recommendation systems, many classic collaborative filtering algorithms adopt the matrix factorization (MF) approach [26–28]. Models based on MF map users and items to a lower-dimensional space using vector representations [29]. Satisfaction is then evaluated by computing the inner product of the vectors representing users and items. However, MF-based models struggle to effectively encode features of the user–item matrix, meaning that MF cannot capture more implicit factors. Moreover, traditional matrix factorization methods based on MF typically only leverage relationships between users and items for recommendations, neglecting user–user relationships. This implies that the model cannot capture information about higher-order neighbors, as simple matrix factorization essentially considers only first-order neighbor information and ignores potential relationships between users. These limitations result in sub-optimal recommendation performance for traditional MF models.

Early social recommendation models based on MF include SocialMF [26], TrustMF [27], and BPR-MF [28]. SocialMF is a recommendation approach based on social networks that assumes the existence of social networks between users and making recommendations based on direct or indirect social relationships between users. TrustMF moves a step further by considering the impact of social trust propagation. In TrustMF, the configuration of trust networks and trust matrices allows for a thorough analysis of the mutual influence between trustors and trustees. In contrast to the previous two models, BPR-MF introduces a pairwise BPR loss, leading to improved optimization of the model.

However, most of the aforementioned MF-based models are no longer effective in handling large-scale data sets and sparsely rated data sets compared with GNN-based methods.

2.2. Recommendation Systems Based on Graph Neural Networks

Methods based on graph neural networks (GNNs) have gradually become mainstream in the field of recommendation systems. The inherent structure of GNNs can effectively represent the graph between users and users, as well as between users and items, which aligns well with the high connectivity observed in social networks. The core objective of GNN-based recommendation systems is to learn representations for each node.

NGCF [30], for instance, aims to further learn embeddings for users and items in the latent space by exploring high-order connectivity relationships to refine the embedding representations. The refined user and item representations are then used for predictions. Diffnet++ [31] constructs a powerful and efficient diffusion network that combines the user–user graph and user–item graph to address the shortcomings of traditional MF-based methods in capturing high-order neighbor information. It incorporates carefully designed attention mechanisms in the model to resolve the inconsistency between the user–user graph and user–item graph relationships.

Approaching the issue of social inconsistency from a different angle, the authors of ref. [32] introduce the concept that social relationships may not necessarily align with rating predictions to create ConsisReg. Based on this insight, they propose a new framework to address this problem.

In many GNN-based models [31,32], graph convolutional networks (GCNs) are commonly used for embedding. However, the GCN was initially designed for homogeneous graphs and may not be suitable for GNN-based social models. LightGCN [33] improved upon the GCN by removing unnecessary non-linear activation functions and feature transformation matrices, making it more suitable for recommendation systems and achieving more advanced recommendation performance.

To further enhance the performance and robustness of recommendation systems, these GNN-based models strive to innovate and address specific challenges, showcasing the ongoing evolution in this area.

2.3. Recommendation System with Graph Contrastive Learning

A highly promising approach recently used in research involves integrating contrastive learning methods into graph-based recommendation systems to address challenges such as insufficient collaboration signals and data sparsity. In particular, SGL [23] combines contrastive learning with recommendation systems, employing three strategies—node dropping, edge dropping, and random walks—to enhance the graph. Contrastive learning is then utilized to obtain diverse self-supervised signals. The model is subsequently jointly trained with these signals as auxiliary modules, leading to improved performance.

SimGRACE [34] introduces perturbations to GCN encoders to generate new views for contrastive learning, while LightGCL [35] employs singular value decomposition to reconstruct different views for contrastive learning. These new views are then jointly optimized with the main model. SimGCL [36], on the other hand, recognizes that complex graph augmentation strategies may adversely affect the model. SimGCL opts for a simpler and more efficient approach by introducing noise into the embedding representations for contrastive learning.

Nevertheless, there is still a lack of a contrastive learning enhanced social recommendation method, where the self-supervision signal can effectively manipulate the learning toward better recommendation.

3. Methodology

This section presents our proposed recommendation model, SSGCL, which consists of four components. We first introduce some essential notation. In the second and third parts, the diffusion module and prediction layer employed in our model are introduced, respectively. The training approach of the model is presented in the last part. The framework of SSGCL is illustrated in Figure 1.

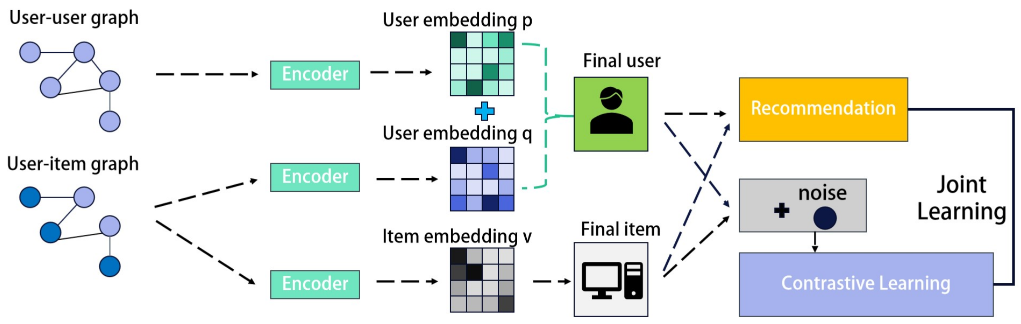

Figure 1. The conceptual framework of our proposed social recommendation system SSGCL. Through a GNN-based encoder, it learns a user embedding p from the user–user graph representing social information, a user embedding q, and an item embedding v from the user–item graph representing interest information. The two user embeddings p and q are summed to provide the final user embedding. Based on these, a contrastive learning module is jointly optimized with the recommendation task to learn the final embedding and recommendation result.

3.1. Notation

This study focuses on the social recommendation problem. Social recommendation systems generally have two sets of entities: a user set U consisting of m users and an item set V consisting of n items. Users may form two types of relationships: social connections between users and interest relationships toward items. These behaviors can be represented as a social graph and an interest graph: the social graph G_s contains nodes denoting users and edges denoting social connections, while the interest graph G_i contains nodes denoting both users and items, and edges denoting user preference toward the items.

These two graphs can be defined by two matrices: a user–user social relationship matrix $S \in \mathbb{R}^{M\times M}$ and a user–item interest relationship matrix $R \in \mathbb{R}^{M\times N}$.

In the user–user social relationship matrix S, if user a trusts user b or, in other words, user a's decision may be influenced by user b, $s_{ab} = 1$; otherwise, $s_{ab} = 0$. We use S_a to represent the set of all users that user a follows or trusts, i.e., $S_a = [b|s_{ab} = 1]$.

Similarly, in the user–item interest matrix R, $r_{ac} = 1$ means user a is interested in item c, and $r_{ac} = 0$ otherwise. R_a represents the set of items that user a is interested in, i.e., $R_a = [c|r_{ac} = 1]$, and R_c represents the set of users that voted for item c, i.e., $R_c = [a|r_{ca} = 1]$.

Our goal is to find unknown preferences from users toward items; that is, a predicted new user–item matrix $\hat{R}$ from R, with new preferences added.

3.2. Diffusion Module

As mentioned before, we design a diffusion module to learn comprehensive representations of users and items based on social and interest information. It has two parts, which acquire the representations from G_s and G_i.

First, representations are acquired from the user–item graph (interest graph G_i). While most GNN-based recommendation systems [12,31] commonly employ traditional GCN encoders to aggregate various information in the social and interest graphs, in this study, we follow [33] and employ a simple but efficient aggregation strategy.

Specifically, both user and item representations are acquired from the interest graph. The diffusion process of the interest graph G_i can be defined as

$$q_i^{(L)} = AGG_{items}(v_j^{(L-1)}, \forall j \in R_i) = \sum_{j \in N_i} \frac{v_j^{(L-1)}}{\sqrt{|R_i||R_j|}}, \tag{1}$$

$$v_j^{(L)} = AGG_{users}(q_i^{(L-1)}, \forall i \in R_j) = \sum_{i \in N_j} \frac{q_i^{(L-1)}}{\sqrt{|R_j||R_i|}}. \tag{2}$$

Here, L is the number of stacked diffusion layers. $q_i^{(L)}$ is the representation of user i obtained from the interest graph. Similarly, $v_j^{(L)}$ is the representation of item j from the interest graph. R_i represents the set of items j that user i is interested in, and R_j is the set of users i that voted for item j. It is worth mentioning that the input of the first layer $q_i^{(0)}$ and $v_j^{(0)}$ are randomly initialized.

Such an encoder is better suited for GNN-based recommendation systems, as it abandons the redundant non-linear transformations of traditional GCN encoders.

The second part involves learning representations from the social graph G_s. Unlike the user–item graph G_i, the social graph only contains nodes representing users, and we can learn only user representations from it. The propagation of user i based on the social graph G_s at the Lth layer could be defined as

$$p_i^{(L)} = AGG_{users}(p_t^{(L-1)}, \forall t \in S_i) = \sum_{t \in S_i} \frac{p_t^{(L-1)}}{\sqrt{|S_i||S_t|}}, \tag{3}$$

where $p_i^{(L)}$ is the embedding for user i at layer L and S_i is the set of socially connected neighbor users of user i.

Upon completion of the aggregation through Equations (1) and (2), representations $q_i^{(L)}$ are obtained for user i after L layers of iterations and $v_j^{(L)}$ for item j after L layers of iterations based on the user–item graph. Through Equation (3), the social-relation-based user representation denoted as $p_i^{(L)}$ is obtained. To aggregate user representation learned from the social graph and the interest graph, previous works [31] designed complex attention mechanisms and combined the two view information from every layer. However, we opt for a simple yet effective sum aggregation function to obtain the final user representation:

$$u_i^{(L)} = q_i^{(L)} + p_i^{(L)} \tag{4}$$

With our aggregation approach, both $q_i^{(L)}$ and $p_i^{(L)}$ effectively learn user representations from their respective views, making full use of the information in each graph without being subject to additional interference.

3.3. Prediction Layer

In the previous section, it is shown that the final user embedding $u_i^{(L)}$ and the final item embedding $v_j^{(L)}$ are obtained from Equations (2) and (4). Then, the prediction results of our model are calculated by taking the inner product of the final user embedding and item embedding:

$$\hat{y}_{ij} = < u_i^{(L)}, v_j^{(L)} > . \tag{5}$$

where $\hat{y}_{ij}$ is user i's predicted preference score for item j, according to our model.

3.4. Model Training

In order to capture more collaborative signals, we employed two types of losses for joint learning to optimize the model: a supervised loss suitable for recommendation systems and a self-supervised loss to capture more collaborative signals.

3.4.1. Supervised Loss

We opted not to use point-wise loss functions [37] which are widely used for recommendation, and instead chose the Bayesian personalized ranking (BPR) pairwise loss function instead. The BPR loss function is specialized for the ranking task. In Bayesian personalized ranking, instances with an interaction are considered positive samples, while those without interaction are considered negative samples. The objective is to maximize the margin between positive and negative samples, which is similar to contrastive learning in principle. This matches our need to determine which items are more worthy of recommendation to the users (potential positive samples with high ranking). Additionally, recommendation data are sparse, with positive samples (instances where users interact with items) being far fewer than negative samples. The sampling-based BPR loss can satisfactorily resolve this problem and lead to better performance. The BPR loss is defined as

$$L_{bpr} = \sum_{(i,j^+,j^-)\in O} -\log \sigma(\hat{y}_{ij^+} - \hat{y}_{ij^-}), \tag{6}$$

where $O = [(i, j^+, j^-)|(i, j^+) \in O^+, (i, j^-) \in O^-]$ is the training data, O^+ represents the observed interaction information, and O^- denotes unobserved interactions. σ is the sigmoid function. The BPR loss provides positive and negative signals to model the user–item interaction but still lacks the ability to enhance the embedding based on the nodes themselves. This could potentially result in sub-optimal representations. To address this issue, we employed a specially designed contrastive loss as assistance.

3.4.2. Self-Supervised Loss

To create contrastive views, traditional GNN-based contrastive learning involves preprocessing steps, such as randomly dropping nodes or edges, performing random walks, or even constructing an entirely new graph [23]. Such strategies tend to be over-designed. According to the findings of previous work, steps such as randomly dropping nodes and edges are likely to result in the loss of important information [36]. Moreover, enhancing the graph in these ways can be very time-consuming.

Therefore, here, we adopt a simpler and more efficient contrastive learning approach. Inspired by [36], traditional graph enhancement methods are abandoned and instead positive and negative samples are constructed by adding noise to the learned embedding of users and items. In particular, for a user embedding u_i, samples $u_i^{'}$ and $u_i^{''}$ are generated as follows:

$$u_i^{'} = u_i + \beta_i^{'}, u_i^{''} = u_i + \beta_i^{''}. \tag{7}$$

Here, for user i with learned representation u_i, $\beta^{'}$ and $\beta^{''}$ represent the added vector noises. These noises are constrained by $||\beta||_2 = \varepsilon$ and $\beta = \bar{\beta} \odot sign(u_i)$, $\bar{\beta} \in \mathbb{R}^d \sim U(0, 1)$. ε is a hyperparameter and $\odot$ is the Hadamard product. These constraints control the magnitude and the orthant of the noise, which could otherwise lead to significant biases and disrupt the original semantics of the node. Figure 2 briefly illustrates our contrastive learning method described in Equation (7).

We consider samples generated from the same node as positive pairs and samples from different nodes as negative ones for contrastive learning. This simple strategy can smoothly manipulate the learned representation toward a better distribution. The InfoNCE contrastive loss [38] is employed to force positive pairs close to each other and negative pairs distant from each other,

$$L_{cl}^{user} = \sum_{i \in U} -\log \frac{\exp(cos < u_i^{'}, u_i^{''} > /\tau)}{\sum_{k \in U} \exp(cos < u_i^{'}, u_k > /\tau)}. \tag{8}$$

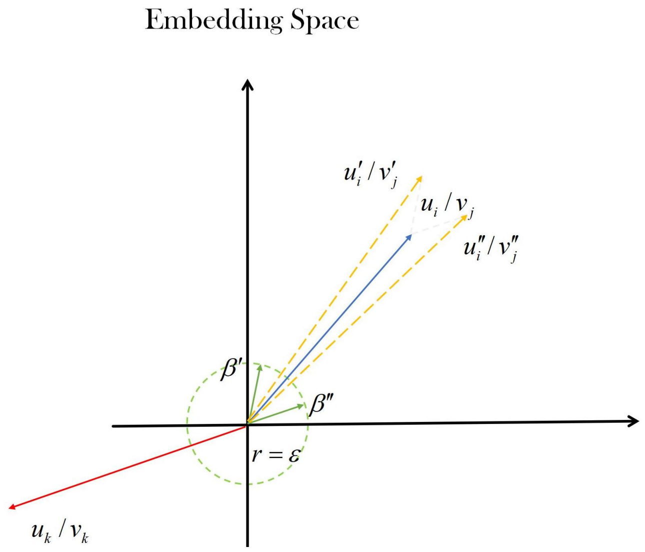

Figure 2. Illustration of data augmentation with added noise. $\beta^{'}$ and $\beta^{''}$ are the noises we add to the learned representations, constrained by hyperparameter ε. $u_i^{'}/v_j^{'}$ and $u_i^{''}/v_j^{''}$ are the samples generated from u_i/v_j with the noises. They are close to the original representation and can be regarded as positive samples, compared with the representations of other nodes u_k/v_k.

Here, $cos <,>$ denotes the cosine similarity between two vectors. τ is a temperature hyperparameter. The denominator is summed over all possible users k in the user set U. This contrastive loss can be regarded as a softmax classifier that classifies $u^{'}$ to $u^{''}$.

Similarly, for item embedding v_j, the following is the item-based noised samples and contrastive loss:

$$v_j^{'} = v_j + \beta_j^{'}, v_j^{''} = v_j + \beta_j^{''}. \tag{9}$$

$$L_{cl}^{item} = \sum_{j \in V} -\log \frac{\exp(cos < v_j^{'}, v_j^{''} > /\tau)}{\sum_{k \in V} \exp(cos < v_j^{'}, v_k > /\tau)}. \tag{10}$$

The final self-supervised loss is the sum of the user-based contrastive loss and the item-based contrastive loss:

$$L_{ssl} = L_{cl}^{item} + L_{cl}^{user} \tag{11}$$

3.4.3. Final Loss

We incorporated our supervised BPR loss (refer to Equation (6)) with our self-supervised contrastive loss (refer to Equation (11)) for joint learning and to optimize our model. The final loss function is defined as

$$L_{final} = L_{bpr} + \lambda_1 L_{ssl} + \lambda_2 ||\theta||_2, \tag{12}$$

where $||\theta||_2$ is the regularization term of L_{final} introduced to prevent model overfitting. λ_1 and λ_2 are two hyperparameters used to control the balance. The learning process is conducted in a supervised manner. The BPR loss is the primary supervised component that optimizes the user and item embedding, while the self-supervised learning module merely serves as an auxiliary part that manipulates the learned embedding toward a better distribution.

4. Experiments

In this section, to demonstrate the effectiveness and superiority of our model, we conducted extensive experiments and addressed the following questions from various perspectives:

- **RQ1:** How does our model perform compared with various state-of-the-art (SOTA) models on different data sets?
- **RQ2:** How do the self-supervised learning module and diffusion modules' propagation mechanisms impact the model?
- **RQ3:** How do different parameter settings affect the model?

4.1. Benchmark Data Sets

We utilized three real-world publicly available data sets: Yelp [23], Last.fm [39], and Douban-book [36]. Below, detailed information about the data sets is provided.

- **Yelp**: Yelp is a geographically based online review platform that encourages users to share their perspectives and experiences by writing reviews and providing ratings. The data set encompasses interactions between users and various locations, including visits and comments. Additionally, by leveraging user friend lists, we can infer user relationships, enabling further analysis of influencing factors within the social network.
- **Douban-book**: Douban-book is a highly influential book-related online social platform in China that gathers extensive information on books and user reviews. This platform enables users to search for books, share their book reviews, rate works, and create personal booklists. Additionally, users can follow other readers, thereby establishing a social network to receive more book recommendations and enhance their interactive experiences.
- **Last.fm**: Last.fm is a music-related data set that includes user, artist, song, and social relationship data. It is widely used in research on music recommendation systems, social network analysis, and music data mining. The data set comprises information such as users' listening histories, social relationships, tags, and events, providing valuable resources for studying music consumption behavior and music social networks.

The three data sets are publicly available for download. All processed data sets were obtained from https://recbole.io/cn/dataset_list.html (accessed on 6 March 2024). A summary of the data set details is provided in Table 1. The data sets were all pre-processed [40] and have been widely used for the recommendation task. Our focus was solely on the social interest information within the data sets, while other user attributes and item details were not within the scope of our consideration.

Table 1. Benchmark graph data sets.

Data Set	Users	Avg. Actions of Users	Items	Avg. Actions of Items	Interactions	Sparsity
Yelp	17,236	12.065	37,379	5.563	207,945	99.968
Douban-book	13,025	60.812	22,348	35.443	792,062	99.728
Last.fm	1893	49.067	17,633	5.265	92,834	99.722

4.2. Baseline Methods

In the experiments, we compared a total of eight recommendation system models with our proposed SSGCL. The main differences between these baselines are summarized in Table 2.

- **MF-BPR** [28] is a recommendation system model. Built upon the idea of matrix factorization, it aims to learn latent representations of users and items for personalized item recommendations. The model is optimized using Bayesian personalized ranking (BPR) loss.

- **Social-MF** [26] is a model designed for recommendation systems. It is also based on the idea of matrix factorization, integrating social information with purchase data to provide personalized recommendations.
- **NGCF** [30] is a deep learning model for recommendation systems, combining traditional collaborative filtering methods with graph neural networks to enhance the effectiveness of personalized recommendations. The core idea involves representing users and items as nodes in a graph and using graph neural networks to learn their latent relationships. Through multiple layers of neural networks, NGCF can better capture the complex interactions between users and items, thereby improving the model performance.
- **Diffnet++** [31] is a graph diffusion model for social recommendations. It combines social and interest information in a recursive manner and employs an attention mechanism during the propagation and aggregation process to enhance the model's performance.
- **MHCN** [41] is a model based on hyper-graph convolutional networks for recommendation systems. The model utilizes social and other information to model edges in the hyper-graph, thereby enhancing the recommendation performance of the model.
- **LightGCN** [33] is a lightweight graph convolutional model. In comparison with traditional graph convolutional networks (GCNs), it simplifies the structure and parameters. By directly updating user and item embedding vectors, it enhances the learning efficiency and effectiveness in social networks and recommendation systems.
- **SGL** [23] is a graph convolutional model that incorporates self-supervised learning for recommendation systems. The algorithm enhances the graph to transform the original graph and create different views by using uniform node and edge dropout strategies. Subsequently, it employs contrastive learning methods to learn from these views. The losses are jointly optimized to obtain the final representations of nodes.
- **SEPT** [42] is a recommendation system model that incorporates self-supervised learning. By constructing complementary views of the graph, pseudo-labels are obtained, and triplet training is employed to assist the training process.

Table 2. Algorithm comparison.

	Interest Information	Social Information	GNN	Self-Supervised Learning	Diffusion Module	BPR
MF-BPR	★					★
Social-MF	★	★				
NGCF	★		★			★
Diffnet++	★	★	★		★	★
MHCN	★	★		★		★
LightGCN	★		★			★
SGL	★		★	★		★
SEPT	★	★	★	★		★
SSCGL	★	★	★	★	★	★

4.3. Experimental Setup

For the effectiveness and fairness of the experiments, we adjusted the hyperparameters for all baselines following the strategies in the original papers. We ran all experiments with the Adam optimizer. The data sets were randomly divided into training, validation, and test sets in an 8:1:1 ratio. Meanwhile, we employed two advanced and effective evaluation metrics that are widely used in top conferences [31,33,36,43]—namely Recall@K and NDCG@K (normalized discounted cumulative gain)—to assess the model performance. In our SSGCL model, we stacked three GNN layers for the influence diffusion, and the embedding size for both users and items was set to 64. The L2 norm was employed for the regularization. The trade-off hyperparameters of the contrastive learning module λ_1 and the regularization term λ_2 were set to 1×10^{-5} and 1×10^{-2}, respectively. The learning rate α was set to 1×10^{-3}. The temperature coefficient τ was 0.15. The noise radius ε is 0.1. Additionally, all our experiments were conducted on an RTX 2070 (8 GB) GPU, and the results of the first 50 epochs are illustrated in Figure 3.

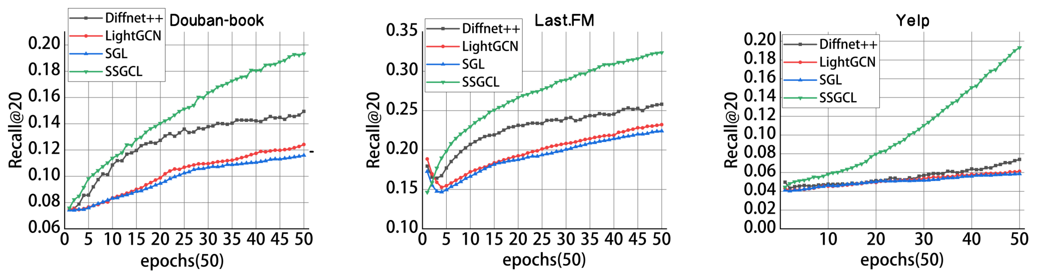

Figure 3. The performance curves in the first 50 epochs.

4.4. Performance Comparison (RQ1)

This study compared eight baseline models with two variants of SSGCL, namely SSGCL-M and SSGCL-L, across three public data sets and two evaluation metrics. SSGCL-M was the model that only aggregated the user representation of the two graphs in the last GNN layer, as described in our Methodology section. SSGCL-L was a variant of our model in which we followed previous works and aggregated the user representation at each layer of the GNN influence diffusion. The learned user representation that combined two sides of information was then updated into the two graphs for the influence diffusion of the next GNN layer.

We conducted the experiment by evaluating the rankings for the top 10, top 20, and top 30 recommendations. As shown in Tables 3–5, our SSGCL-M (final aggregation) consistently outperformed the optimal baseline in the top@k experiments, while SSGCL-L (intermediate aggregation) achieved a performance close to the baseline. We summarize our findings as follows:

1. **The exploration of social information could benefit the recommendation system.**

The tables show that many recommendation systems that incorporate social information performed better than other methods proposed during the same time period, such as Diffnet++, SEPT, and our method. However, the advantages of social information were not apparent when compared with more up-to-date methods. This was probably because taking full advantage of social information is difficult. Early approaches lack effective ways to jointly learn social interactions with the interest information.

2. **The approaches based on GNNs exhibited excellent performance.**

From the tables, it is evident that GNN-based models generally exhibited a significant improvement over traditional approaches, such as matrix factorization-based recommendation systems. For example, LightGCN and SGL performed quite well without social information, and mainly benefited from their well-designed GNN layers. MHCN involves both social information and self-supervised signals but has yet to achieve satisfactory results, which may be because it fails to model multifaceted information sufficiently without the assistance of GNNs. This demonstrates a GNN's effectiveness in handling such interest and social networks, as well as modeling the complex interactions in the recommendation task.

3. **Self-supervised learning shows great potential in recommendation system tasks.**

It can be observed from the tables that recent recommendation system approaches, such as SGL and SEPT, tended to employ self-supervised learning and obtained satisfactory performance. This was because the self-supervised learning module could help to learn more informative representations and make the user and item representation uniform, thereby helping to achieve better recommendation results.

4. **The superiority of our SSGCL.**

Our SSGCL sufficiently explored the social and interest network with a GNN-based influence diffusion layer. The learned representation was further optimized with a contrastive learning module that extracts self-supervised signals. This should be the reason why our SSGCL performed well compared with the baseline methods.

Table 3. Experimental results for the Douban-book data set.

Model	Recall@10	Recall@20	Recall@30	NDCG@10	NDCG@20	NDCG@30
MF-BPR	0.0901	0.1362	0.1685	0.0706	0.0838	0.1059
Social-MF	0.0246	0.0399	0.0534	0.0122	0.0162	0.0192
NGCF	0.1061	0.1559	0.1905	0.0831	0.0968	0.1068
Diffnet++	0.1285	0.1822	0.2181	0.0998	0.1151	0.1251
MHCN [1]	-	-	-	-	-	-
LightGCN	0.1962	0.2576	0.3306	0.1666	0.1828	0.1947
SGL	0.1941	0.2581	0.3025	0.1681	0.1828	0.1972
SEPT	0.2370	0.3046	0.3523	0.1957	0.2138	0.2271
SSGCL-L	0.2407	0.3042	0.3468	0.1961	0.2129	0.2248
SSGCL-M [2]	**0.2765**	**0.3367**	**0.3757**	**0.2298**	**0.2460**	**0.2572**

[1] The Douban-book data set was too large for the MHCN algorithm. [2] In the table, data in bold indicates optimal performance.

Table 4. Experimental results for the Last.fm data set.

Model	Recall@10	Recall@20	Recall@30	NDCG@10	NDCG@20	NDCG@30
MF-BPR	0.1786	0.2438	0.2958	0.1681	0.1958	0.2145
Social-MF	0.1510	0.2196	0.2676	0.1400	0.1690	0.1863
NGCF	0.1958	0.2738	0.3261	0.1854	0.2185	0.2372
Diffnet++	0.2545	0.3521	0.4184	0.2512	0.2921	0.3159
MHCN	0.3357	0.4684	0.5499	0.2995	0.3557	0.3848
LightGCN	0.3929	0.5097	0.5823	0.3808	0.4308	0.4571
SGL	0.3831	0.4991	0.5741	0.3739	0.4231	0.4501
SEPT	0.3774	0.4818	0.5562	0.3753	0.4195	0.4463
SSGCL-L	0.4844	0.6092	0.6770	0.4669	0.5202	0.5448
SSGCL-M	**0.5495**	**0.6650**	**0.7241**	**0.5510**	**0.6005**	**0.6219**

Table 5. Experimental results for the Yelp data set.

Model	Recall@10	Recall@20	Recall@30	NDCG@10	NDCG@20	NDCG@30
MF-BPR	0.0222	0.0367	0.0512	0.0112	0.0151	0.0182
Social-MF	0.0240	0.0400	0.0543	0.0122	0.0163	0.0195
NGCF	0.0327	0.0571	0.0747	0.0164	0.0227	0.0265
Diffnet++	0.2077	0.2678	0.3081	0.1456	0.1613	0.1702
MHCN	0.1368	0.1996	0.2470	0.0787	0.0951	0.1055
LightGCN	0.5721	0.6631	0.7136	0.4318	0.4561	0.4674
SGL	0.5832	0.6695	0.7194	0.4478	0.4708	0.4822
SEPT	0.4101	0.5065	0.5638	0.2816	0.3068	0.3195
SSGCL-L	0.6772	0.7690	0.8143	0.5050	0.5296	0.5505
SSGCL-M	**0.7143**	**0.7910**	**0.8296**	**0.5546**	**0.5755**	**0.5846**

4.5. Ablation Study (RQ2)

Ablation experiments helped us gain a deeper understanding of the contributions of different components of the model to the overall performance. Through gradually removing specific parts of the model or altering their configurations, we assessed the impacts of these parts on the model's performance, thus revealing key factors in the model design.

From the above experiments, we could observe that both SSGCL-L and SSGCL-M performed well. The variant SSGCL-L adopts a more complicated influence diffusion strategy that aggregates the user representation at each layer like in many other algorithms that take social information into account. However, it cannot beat SSGCL-M, which has a simple final aggregation strategy.

In this subsection, to further investigate the effectiveness of the model's simple influence diffusion strategy and the contrastive learning module, we conducted ablation experiments to address the question of whether simpler aggregation mechanisms and contrastive learning make sense. For this purpose, we compared four variants of the model, namely SSGCL-M utilizing final layer aggregation with contrastive learning, SSGCL-L employing intermediate layer aggregation with contrastive learning, SSGCL-N using final layer aggregation without contrastive learning, and SSGCL-O using intermediate layer aggregation without contrastive learning. To ensure the fairness of the ablation experiments, we used the same optimizer and other parameters as mentioned before.

From Tables 6–8, it is evident that SSGCL-M achieved the best performance on all three public data sets. This further shows the effectiveness of our simple final layer aggregation and the contrastive learning module. SSGCL-M performed better than SSGCL-L, and SSGCL-N performed better than SSGCL-O, both of which demonstrated that the precisely designed propagation of previous social recommendation systems did not have much of an effect on the final performance; furthermore, our final aggregation strategy was simple and effective, and it lead to better representations and recommendation results. SSGCL-M and SSGCL-L clearly outperformed the other two variants, which shows the superiority of our contrastive learning module.

Table 6. Ablation study of the Douban-book data set.

Model	Recall@10	Recall@20	Recall@30	NDCG@10	NDCG@20	NDCG@30
SSGCL-M	**0.2765**	**0.3367**	**0.3757**	**0.2298**	**0.2460**	**0.2572**
SSGCL-L	0.2407	0.3042	0.3468	0.1961	0.2129	0.2248
SSGCL-N	0.1947	0.2613	0.3053	0.1619	0.1792	0.1914
SSGCL-O	0.1717	0.2335	0.2796	0.1400	0.1560	0.1856

Table 7. Ablation study of the Last.fm data set.

Model	Recall@10	Recall@20	Recall@30	NDCG@10	NDCG@20	NDCG@30
SSGCL-M	**0.5495**	**0.6650**	**0.7241**	**0.5510**	**0.6005**	**0.6219**
SSGCL-L	0.4844	0.6092	0.6770	0.4669	0.5202	0.5448
SSGCL-N	0.3162	0.4321	0.5100	0.3049	0.3541	0.3821
SSGCL-O	0.2997	0.4115	0.4897	0.2893	0.3370	0.3648

Table 8. Ablation study of the Yelp data set.

Model	Recall@10	Recall@20	Recall@30	NDCG@10	NDCG@20	NDCG@30
SSGCL-M	**0.7143**	**0.7910**	**0.8296**	**0.5546**	**0.5755**	**0.5846**
SSGCL-L	0.5772	0.7690	0.8143	0.5050	0.5298	0.5505
SSGCL-N	0.4133	0.5140	0.5748	0.2808	0.3072	0.3212
SSGCL-O	0.3706	0.4708	0.5454	0.2455	0.2732	0.2884

4.6. Hyperparameter Analysis (RQ3)

In this subsection, we present the results of the experiments that were conducted to explore the sensitivity of several key hyperparameters in our model. Analyzing the parameter sensitivity helped us to better understand the behavior and performance of the model. Through experimenting with and evaluating different parameter values, we can understand how the model's performance changed under different settings, thereby determining the optimal parameter combination.

The contrastive learning coefficient λ_1: We conducted experiments on the contrastive learning coefficient λ_1, and the results are shown in Figure 4. The model performance peaked on the Last.fm and Douban-book data sets when the contrastive learning coefficient reached 1×10^{-5}. For the Yelp data set, when the contrastive learning coefficient reached 1×10^{-4}, the model performance peaked. This shows that adjusting the parameters on different data sets can optimize model performance, as different data sets exhibit varying sensitivities to parameters.

The regularization coefficient λ_2: We conducted experiments on the regularization coefficient λ_2. The main purpose of the regularization parameter λ_2 is to control the complexity of the model, thereby helping to prevent overfitting. When the model becomes overly complex, it may perform well on the training data but poorly on unseen data, indicating a lack of generalization to new samples. By introducing the regularization parameter, a penalty term was added to the loss function, penalizing the size of the model parameters, which made the model simpler and improved its generalization capability. The results are shown in Figure 5. We observed that when the regularization parameter was set to 1×10^{-2}, our model achieved the best performance on all data sets, which was different from the case of the contrastive learning coefficient. Additionally, we found that when we set the parameter between 1×10^{-4} and 1×10^{-8}, the experimental results on the Douban-book and Last.fm data sets remained stable, with the graph lines leveling off. This trend was more rapid on the Yelp data set. When the parameter was set in the range of $[1 \times 10^{-2}, 1 \times 10^{-8}]$, the experimental results stabilized, and the decrease in model performance was minimal. These findings provide guidance for selecting and adjusting regularization parameters, emphasizing the importance of finding optimal parameters within a certain range to achieve stable and excellent model performance.

The temperature coefficient of InfoNCE τ: The temperature coefficient τ plays a crucial role in adjusting the scale of the contrastive size. The main function of temperature coefficient τ is to control the proportion of positive and negative sample pairs. From a mathematical perspective, the temperature coefficient τ in Equation (8) is positioned in the denominator. Figure 6 clearly shows that, when the model parameter was set to 0.15, the best performance was achieved on all three data sets. On the Douban-book and Last.fm data sets, the model performance showed an upward trend as the value of parameter τ increased within the range [0.1, 0.15]. However, when the parameter was set to [0.15, 1], the model performance started to decline. On the Yelp data set, the model performance remained relatively stable within the range [0.1, 0.15], but exhibited a declining trend within the range of [0.15, 1]. Overall, the optimal performance was achieved when the model parameter was set to 0.15.

Number of GNN layers: Due to the excessive smoothing issue of GNNs, we conducted experiments to explore the best number of GNN layers for our model. The experimental results are shown in Figure 7. For the Last.fm and Yelp data sets, as the number of GNN layers increased, the model's performance improved, reaching its peak when the number of GNN layers was set to three. However, on the Douban-book data set, the model performance peaked when the GNN layers reached two. This indicates that different data sets exhibited varying sensitivities to the number of GNN layers. The Douban-book data set, being larger and more intricate in its interrelations compared with the other two data sets, further underscored the issue of excessive smoothing in graph neural networks (GNNs). When data sets exhibit denser and more diverse interaction patterns, increasing the number of GNN layers may lead to over-smoothing of the features, resulting in a loss of detail and discriminative power in the learned features.

Noise radius ε: We abandoned traditional graph augmentation methods, such as random node deletion or random edge deletion, opting instead to introduce uniform noise into the samples to construct sample pairs and simultaneously enhance the model's robustness. However, the introduction of noise must be constrained, as excessive noise can severely impact the model. Therefore, we imposed a radius constraint on the noise introduced to ensure the effectiveness and stability of the model. We conducted experiments with the noise radius parameter ε set to 0.01, 0.05, 0.1, 0.2, 0.3, and 0.4. We found that the model achieved optimal performance across all three data sets when the noise radius r was set to 0.1. Moreover, as the noise radius ε increased, the decreasing trend in performance tended to stabilize. This indicates that our model has a certain level of robustness. The experimental results are shown in Figure 8.

Through the parameter sensitivity experiments, we found that the data distribution varied across each data set, leading to the requirement of different parameter configurations for the model to achieve optimal performance. Therefore, we had to adjust the parameters to obtain the best performance. This also implies that in recommendation systems, we can enhance the generalization performance by adjusting the parameters. In other words, parameter adjustment can improve both the applicability and performance of the recommendation system.

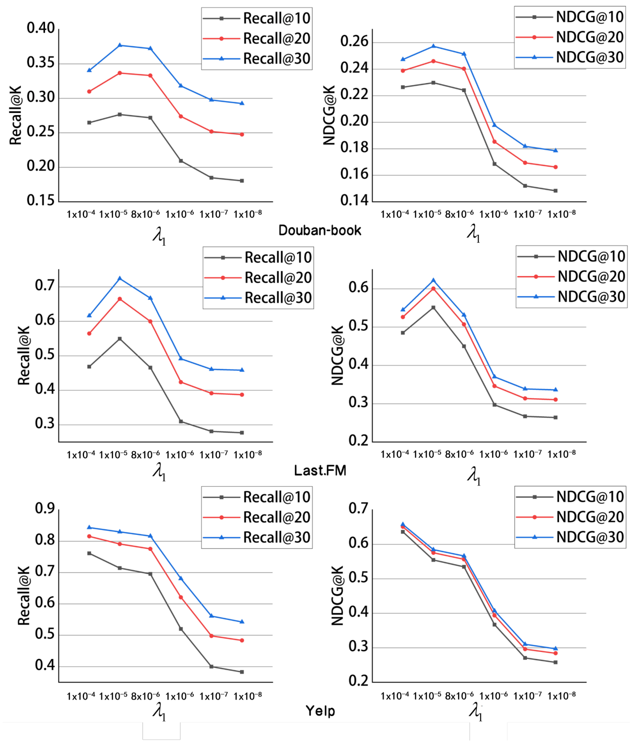

Figure 4. Sensitivity analysis of λ_1.

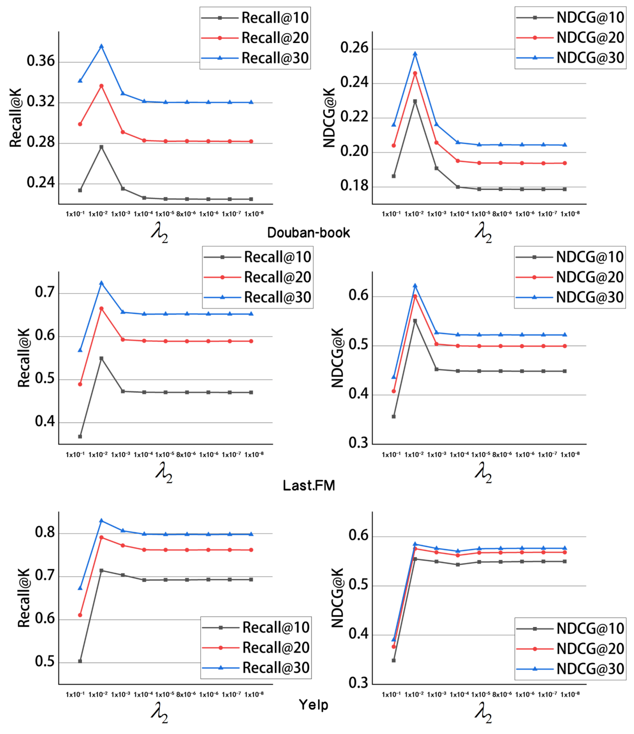

Figure 5. Sensitivity analysis of λ_2.

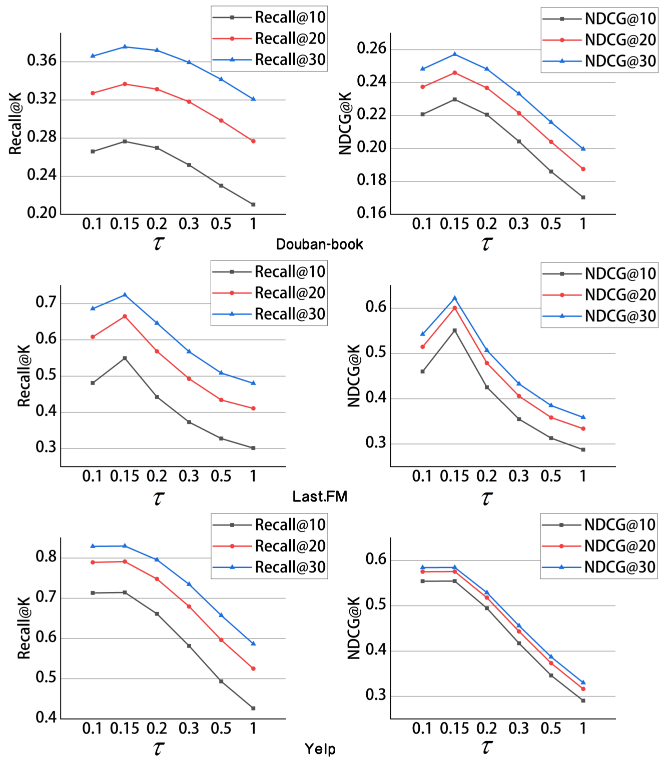

Figure 6. Sensitivity analysis of τ.

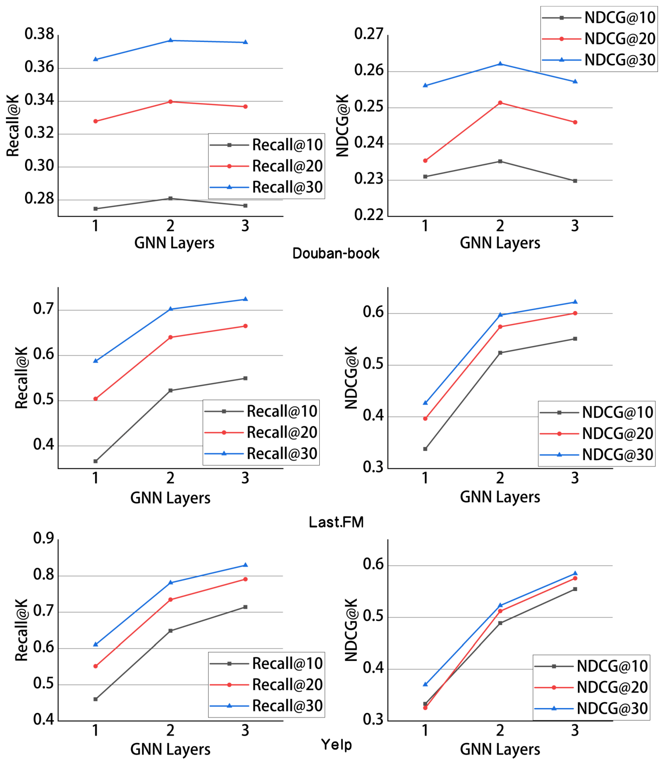

Figure 7. Sensitivity analysis of GNN layers.

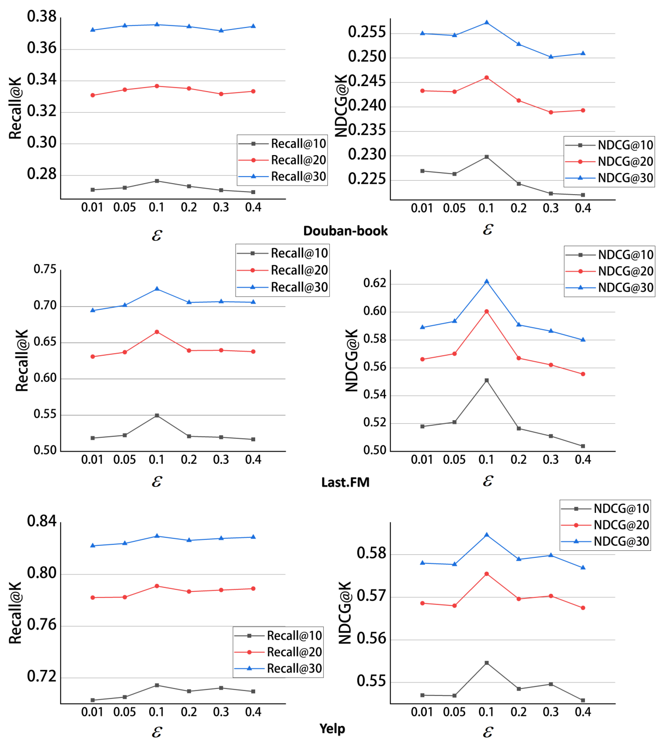

Figure 8. Sensitivity analysis of ε.

5. Conclusions and Future Work

This paper proposes a simpler and more effective social recommendation system called SSGCL. Specifically, we explore simpler aggregation methods and design a GNN-based influence diffusion module that learns informative user and item representations from both the interest network and the social network. A special contrastive learning module is then employed, which generates positive and negative samples by simply adding noises to the original node embedding to manipulate the learned representations for a better distribution. The experiments demonstrate that some of the strategies in previous social recommendation models are over-designed, while our model is effective and simple. We achieve state-of-the-art results on three public data sets in terms of two evaluation metrics.

While our recommendation model has made significant progress in terms of performance, there are other issues and limitations inherent in recommendation systems that should be recognized. For instance, most existing recommendation systems face the risk of data leakage because they utilize users' private data and do not consider privacy protection, making these data vulnerable to malicious attacks. Some companies choose to completely safeguard data or withhold labels during training to prevent data leakage. However, while this approach ensures data security, it sacrifices data interactivity, leading to sub-optimal model performance during training. Similar issues are addressed in other scenarios but not well explored for recommendation systems yet. Furthermore, social influence and algorithmic bias are also common issues in recommendation systems. Recommendation systems may excessively filter user information, causing users to only encounter information related to their interests, neglecting other important information and leading to societal fragmentation. Ensuring more diverse recommendations while improving recommendation system performance remains a challenge and a focal point for future efforts.

Although these issues are outside the scope of this paper's consideration, we intend to address them in our future work. In our future research, we aim to broaden our focus to explore the robustness of recommendation systems and their security and privacy concerns to better align with real-world applications and offer numerous intriguing avenues for future exploration. Additionally, we would also like to introduce more information, whereas this study focused solely on social recommendation systems, primarily incorporating social information. For example, there has been a surge of interest in knowledge graphs [44] within the field of recommendation systems. In this way, we can potentially uncover more latent relationships and achieve superior performance.

Regarding future work, as mentioned, there has been a surge of interest in knowledge graphs [44] within the field of recommendation systems. Knowledge graphs offer the capability to integrate various types of data, whereas this study focused solely on social recommendation systems, primarily incorporating social information. Introducing knowledge graphs into recommendation systems could potentially enrich the system with a broader range of information, thereby uncovering more latent relationships and achieving superior performance. On the other hand, incorporating additional information into recommendation systems may pose challenges, such as compromising model robustness and introducing excessive noise [19]. In our future research endeavors, we aim to broaden our focus beyond solely self-supervised learning of graphs. Instead, we intend to explore a wider array of topics, including the semi-supervised learning of graphs [45], the security of graph data [46], and privacy concerns in graph recommendation systems [47]. This expansion will enable our recommendation systems to better align with real-world applications and offer numerous intriguing avenues for future exploration.

Author Contributions: Methodology, W.Z.; Writing – original draft, Z.D.; Writing—review & editing, C.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data sets used in our experiments are all publicly available.We have made the code publicly available on github and implemented it using PyTorch. https://github.com/0Nagatuki0/SSGCL.git.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Gong, X.; Feng, Q.; Zhang, Y.; Qin, J.; Ding, W.; Li, B.; Jiang, P.; Gai, K. Real-time Short Video Recommendation on Mobile Devices. In Proceedings of the 31st ACM International Conference on Information & Knowledge Management, Atlanta, GA, USA, 17–21 October 2022; pp. 3103–3112.
2. Fan, W.; Ma, Y.; Li, Q.; He, Y.; Zhao, E.; Tang, J.; Yin, D. Graph neural networks for social recommendation. In Proceedings of the World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 417–426.
3. Volokhin, S.; Collins, M.D.; Rokhlenko, O.; Agichtein, E. Augmenting Graph Convolutional Networks with Textual Data for Recommendations. In Proceedings of the European Conference on Information Retrieval, Dublin, Ireland, 2 April 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 664–675.
4. Shi, Y.; Larson, M.; Hanjalic, A. List-wise learning to rank with matrix factorization for collaborative filtering. In Proceedings of the Fourth ACM Conference on Recommender Systems, Barcelona, Spain, 26–30 September 2010; pp. 269–272.
5. Koren, Y.; Bell, R.; Volinsky, C. Matrix factorization techniques for recommender systems. *Computer* **2009**, *42*, 30–37. [CrossRef]
6. Abdollahi, B.; Nasraoui, O. Explainable matrix factorization for collaborative filtering. In Proceedings of the 25th International Conference Companion on World Wide Web, Montreal, QC, Canada, 11–15 April 2016; pp. 5–6.
7. Liu, X.; Aggarwal, C.; Li, Y.F.; Kong, X.; Sun, X.; Sathe, S. Kernelized matrix factorization for collaborative filtering. In Proceedings of the 2016 SIAM International Conference on Data Mining, Miami, FL, USA, 5–7 May 2016; pp. 378–386.
8. Baltrunas, L.; Ludwig, B.; Ricci, F. Matrix factorization techniques for context aware recommendation. In Proceedings of the Fifth ACM Conference on Recommender Systems, Chicago, IL, USA, 23–27 October 2011; pp. 301–304.
9. Nguyen, J.; Zhu, M. Content-boosted matrix factorization techniques for recommender systems. *Stat. Anal. Data Min. ASA Data Sci. J.* **2013**, *6*, 286–301. [CrossRef]
10. Yu, H.F.; Hsieh, C.J.; Si, S.; Dhillon, I.S. Parallel matrix factorization for recommender systems. *Knowl. Inf. Syst.* **2014**, *41*, 793–819. [CrossRef]
11. Kumar, R.; Verma, B.; Rastogi, S.S. Social popularity based SVD++ recommender system. *Int. J. Comput. Appl.* **2014**, *87*, 33–37. [CrossRef]
12. Wu, S.; Sun, F.; Zhang, W.; Xie, X.; Cui, B. Graph neural networks in recommender systems: A survey. *ACM Comput. Surv.* **2022**, *55*, 1–37. [CrossRef]
13. Scarselli, F.; Gori, M.; Tsoi, A.C.; Hagenbuchner, M.; Monfardini, G. The graph neural network model. *IEEE Trans. Neural Netw.* **2008**, *20*, 61–80. [CrossRef]
14. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph neural networks: A review of methods and applications. *AI Open* **2020**, *1*, 57–81. [CrossRef]

15. Yin, R.; Li, K.; Zhang, G.; Lu, J. A deeper graph neural network for recommender systems. *Knowl.-Based Syst.* **2019**, *185*, 105020. [CrossRef]
16. Huang, T.; Dong, Y.; Ding, M.; Yang, Z.; Feng, W.; Wang, X.; Tang, J. MixGCF: An improved training method for graph neural network-based recommender systems. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Online, 14–18 August 2021; pp. 665–674.
17. Xia, L.; Huang, C.; Xu, Y.; Dai, P.; Bo, L. Multi-behavior graph neural networks for recommender system. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *35*, 5473–5487. [CrossRef]
18. Wang, C.; Pan, S.; Hu, R.; Long, G.; Jiang, J.; Zhang, C. Attributed graph clustering: A deep attentional embedding approach. *arXiv* **2019**, arXiv:1906.06532.
19. Berahmand, K.; Daneshfar, F.; Salehi, E.S.; Li, Y.; Xu, Y. Autoencoders and their applications in machine learning: A survey. *Artif. Intell. Rev.* **2024**, *57*, 28. [CrossRef]
20. Daneshfar, F.; Soleymanbaigi, S.; Nafisi, A.; Yamini, P. Elastic deep autoencoder for text embedding clustering by an improved graph regularization. *Expert Syst. Appl.* **2024**, *238*, 121780. [CrossRef]
21. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. In Proceedings of the International Conference on Machine Learning, PMLR, Online, 13–18 July 2020; pp. 1597–1607.
22. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9729–9738.
23. Wu, J.; Wang, X.; Feng, F.; He, X.; Chen, L.; Lian, J.; Xie, X. Self-supervised graph learning for recommendation. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Online, 11–15 July 2021; pp. 726–735.
24. Wang, H.; Zhang, J.; Zhu, Q.; Huang, W. Augmentation-free graph contrastive learning with performance guarantee. *arXiv* **2022**, arXiv:2204.04874.
25. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
26. Jamali, M.; Ester, M. A matrix factorization technique with trust propagation for recommendation in social networks. In Proceedings of the Fourth ACM Conference on Recommender Systems, Barcelona, Spain, 26–30 September 2010; pp. 135–142.
27. Yang, B.; Lei, Y.; Liu, J.; Li, W. Social collaborative filtering by trust. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1633–1647. [CrossRef] [PubMed]
28. Rendle, S.; Freudenthaler, C.; Gantner, Z.; Schmidt-Thieme, L. BPR: Bayesian personalized ranking from implicit feedback. *arXiv* **2012**, arXiv:1205.2618.
29. Yang, L.; Cao, X.; He, D.; Wang, C.; Wang, X.; Zhang, W. Modularity based community detection with deep learning. In Proceedings of the IJCAI, New York, NY, USA, 9–15 July 2016; Volume 16, pp. 2252–2258.
30. Wang, X.; He, X.; Wang, M.; Feng, F.; Chua, T.S. Neural graph collaborative filtering. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, Paris, France, 21–25 July 2019; pp. 165–174.
31. Wu, L.; Li, J.; Sun, P.; Hong, R.; Ge, Y.; Wang, M. Diffnet++: A neural influence and interest diffusion network for social recommendation. *IEEE Trans. Knowl. Data Eng.* **2020**, *34*, 4753–4766. [CrossRef]
32. Yang, L.; Liu, Z.; Dou, Y.; Ma, J.; Yu, P.S. ConsisRec: Enhancing GNN for social recommendation via consistent neighbor aggregation. In Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Online, 11–15 July 2021; pp. 2141–2145.
33. He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; Wang, M. 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, Online, 25–30 July 2020; pp. 639–648.
34. Xia, J.; Wu, L.; Chen, J.; Hu, B.; Li, S.Z. SimGRACE: A simple framework for graph contrastive learning without data augmentation. In Proceedings of the ACM Web Conference 2022, Online, 25–29 April 2022; pp. 1070–1079.
35. Cai, X.; Huang, C.; Xia, L.; Ren, X. LightGCL: Simple Yet Effective Graph Contrastive Learning for Recommendation. *arXiv* **2023**, arXiv:2302.08191.
36. Yu, J.; Yin, H.; Xia, X.; Chen, T.; Cui, L.; Nguyen, Q.V.H. Are graph augmentations necessary? Simple graph contrastive learning for recommendation. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–15 July 2022; pp. 1294–1303.
37. Chai, T.; Draxler, R.R. Root mean square error (RMSE) or mean absolute error (MAE)?–Arguments against avoiding RMSE in the literature. *Geosci. Model Dev.* **2014**, *7*, 1247–1250. [CrossRef]
38. Oord, A.v.d.; Li, Y.; Vinyals, O. Representation learning with contrastive predictive coding. *arXiv* **2018**, arXiv:1807.03748.
39. Karakayali, N.; Kostem, B.; Galip, I. Recommendation systems as technologies of the self: Algorithmic control and the formation of music taste. *Theory, Cult. Soc.* **2018**, *35*, 3–24. [CrossRef]
40. Zhao, W.X.; Mu, S.; Hou, Y.; Lin, Z.; Chen, Y.; Pan, X.; Li, K.; Lu, Y.; Wang, H.; Tian, C.; et al. RecBole: Towards a unified, comprehensive and efficient framework for recommendation algorithms. In Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Online, 1–5 November 2021; pp. 4653–4664.
41. Yu, J.; Yin, H.; Gao, M.; Xia, X.; Zhang, X.; Viet Hung, N.Q. Socially-aware self-supervised tri-training for recommendation. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Online, 14–18 August 2021; pp. 2084–2092.

42. Yu, J.; Yin, H.; Li, J.; Wang, Q.; Hung, N.Q.V.; Zhang, X. Self-supervised multi-channel hypergraph convolutional network for social recommendation. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 413–424.
43. Gao, Y.; Du, Y.; Hu, Y.; Chen, L.; Zhu, X.; Fang, Z.; Zheng, B. Self-guided learning to denoise for robust recommendation. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–15 July 2022; pp. 1412–1422.
44. Yang, Y.; Huang, C.; Xia, L.; Li, C. Knowledge graph contrastive learning for recommendation. In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, 11–15 July 2022; pp. 1434–1443.
45. Daneshfar, F.; Soleymanbaigi, S.; Yamini, P.; Amini, M.S. A survey on semi-supervised graph clustering. *Eng. Appl. Artif. Intell.* **2024**, *133*, 108215. [CrossRef]
46. Sun, L.; Dou, Y.; Yang, C.; Zhang, K.; Wang, J.; Philip, S.Y.; He, L.; Li, B. Adversarial attack and defense on graph data: A survey. *IEEE Trans. Knowl. Data Eng.* **2022**, *35*, 7693–7711. [CrossRef]
47. Zhang, S.; Yin, H.; Chen, T.; Huang, Z.; Cui, L.; Zhang, X. Graph embedding for recommendation against attribute inference attacks. In Proceedings of the Web Conference 2021, Ljubljana, Slovenia, 19–23 April 2021; pp. 3002–3014.