Community-Aware Social Recommendation: A Unified SCSVD Framework

Jiewen Guan, Xin Huang, and Bilian Chen

Abstract—Recommender system provides personalized suggestions based on users' interests and social connections. However, most existing social recommendation models utilize social relationships in a direct manner, i.e., they only consider the user-user connections, neglecting the clustering nature of social networks. As social information recursively spreads in the social network, the community structure, which contains richer information in contrast to pure user-user relationships, would emerge. To dismiss these limitations, in this paper, we propose a unified recommendation framework named Simultaneous Community detection and Singular Value Decomposition (SCSVD), which utilizes the underlying community structure to regularize user latent preferences. We propose a well-designed iterative optimization algorithm to tackle social recommendation efficiently. In addition, we theoretically analyze the proposed algorithm in terms of convergence, time complexity, and also the unified process of community detection and user embedding learning. Extensive experiments are conducted on three benchmark real-world datasets of product reviews, demonstrating the effectiveness, robustness, and flexibility of SCSVD in both rating prediction and top-N recommendation tasks, compared to fifteen state-of-the-art approaches.

Index Terms—Social recommendation, community detection, optimization, theoretical analysis

1 Introduction

Recommender systems, which aim at mitigating the information overload problem [1], are now playing a more and more important role in providing users with information of their potential interests. Nowadays, the omnipresent social media allows users to issue many online social actions to others, such as following [2], trusting [3], etc. These activities provide additional information for recommender systems. Social correlation theories, especially the social homophily [4] and the social influence [5], imply that two socially-connected users are more likely to share their preferences. As a result, users would affect each other to be more similar, which shows potential to increase recommender systems. The exploitation of social networks for recommendation has attracted more and more attention during the last decade, and those derived models have indeed enhanced the capability and interpretability of recommender systems remarkably [6], [7], [8], [9], [10]. However, the vast majority of these models focus only on utilizing social information between connected user pairs, neglecting the community structure in a network, which is usually found in real networks and proven to be informative [11], [12]. Such multi-hop interpersonal relationships may provide another different perspective for improving recommendation.

Complex networks, such as social networks, communication networks, and financial networks, are abstract representations of complex systems, and they serve as a fundamental tool for analyzing the nature and functionality of complex systems. One of the most important properties of complex networks is their underlying community structure, where nodes are connected more densely within clusters than across clusters [13]. In the context of social networks, users form groups, and users within the same group will have more similar preferences, even they are not directly connected. Discovering the community structure in complex networks has been extensively investigated in the last two decades [14], and the community structure in social networks is also proven to be useful for other tasks, such as learning node embedding [15], link prediction [16], etc. However, only a few works, like [17], [18], have been proposed to apply community detection to increase the quality of recommender systems. Both studies [17], [18] adopt a two-stage method, i.e., first detecting communities in a social network, and then predicting ratings or recommending items based on those detected communities. However, although such a two-stage scheme could improve the performance of item recommendation or rating prediction, they are not really recommendation oriented, i.e., although users within those detected communities manifest similarity to some extent, those detected communities are irrelevant of the rating information, and it is not guaranteed that users are

- *Jiewen Guan and Bilian Chen are with the Department of Automation, Xiamen University, Xiamen 361005, China, and also with the Xiamen Key Laboratory of Big Data Intelligent Analysis and Decision-making, Xiamen 361005, China. E-mail: 23220191151283@stu.xmu.edu.cn, blchen@xmu.edu.cn.*
- *Xin Huang is with the Department of Computer Science, Hong Kong Baptist University, Kowloon Tong, Hong Kong. E-mail: xinhuang@comp.hkbu.edu.hk.*

Manuscript received 17 Dec. 2020; revised 11 Sept. 2021; accepted 1 Oct. 2021. Date of publication 6 Oct. 2021; date of current version 3 Feb. 2023.
The work was supported in part by the Youth Innovation Fund of Xiamenunder Grant 3502Z20206049, in part by the National Natural Science Foundation of China under Grant 61836005, and in part by the HK RGC under Grant 22200320.
(Corresponding author: Bilian Chen.)
Recommended for acceptance by R. C.-W. Wong.
Digital Object Identifier no. 10.1109/TKDE.2021.3117686

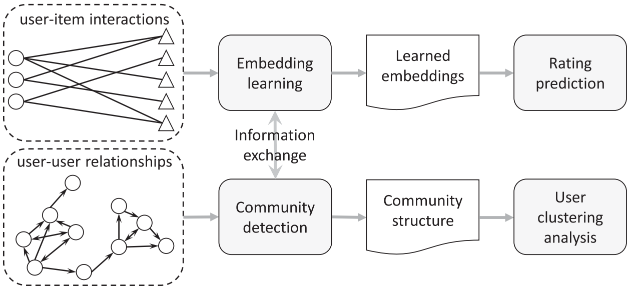


Fig. 1. An overview of the framework of SCSVD. SCSVD simultaneously learns latent vectors of entities and detects communities of users, where information could be exchanged during these two processes.

clustered towards a direction of better rating prediction or better item recommendation.

Based on the above discussions, in this paper, we propose a novel community detection based recommender system, named Simultaneous Community detection and Singular Value Decomposition (SCSVD), to deal with the social recommendation task. Fig. 1 illustrates an overview of our proposed SCSVD. Instead of merely applying the concept of community detection first and then constructing a recommendation model based on that, SCSVD adopts a one-stage unified framework to build a bridge between rating information and social relations. In our iterative optimization process, user latent preferences are regularized by the underlying community structure in the social network, so as to obtain a better user preference modeling. Besides, we derive an efficient optimization algorithm with convergence guarantee to solve the optimization problem of SCSVD.

To summarize, we make the following contributions:

- We analyze real-world recommendation and social network datasets in terms of various structural properties. Based on our observations, we propose a hypothesis of strong correlation between community properties and rating features, and then validate its correctness. (Section 3)
- We propose a novel Simultaneous Community detection and Singular Value Decomposition (SCSVD) framework for social recommendation, which utilizes community structural information to enhance the quality of recommendation. In our scheme, the learned user latent embeddings are regularized by the underlying community structure in the social network. (Section 4)
- We further derive an efficient learning algorithm to optimize SCSVD. Then we give theoretical analysis on the convergence and computational complexity of the learning algorithm. Moreover, the functionary mechanism of SCSVD is also theoretically analyzed. (Section 4)
- We conduct comprehensive experiments on three real-world benchmark datasets to validate the effectiveness and robustness of SCSVD against fifteen state-of-the-art competitive methods, on two recommendation tasks of rating prediction and item ranking. We also conduct parameter sensitivity analysis, empirical convergence analysis and embedding visualization to verify the effectiveness of our proposed SCSVD framework. (Section 5)

We review related work in Section 2 and conclude our work with future directions in Section 6.

2 Related Work

Our work is related to matrix factorization based representation learning, social recommendation and community detection.

Matrix Factorization Based Representation Learning. Matrix factorization aims to factorize a given data matrix into two factor matrices, which can be viewed as representations for different entities. Matrix factorization based representation learning has a wide range of applications, such as recommendation systems, image processing and data clustering. Li *et al.* [19] proposed a non-negative matrix factorization based method for robust image dimensionality reduction. Besides, they also proposed a matrix factorization based social image understanding model DCE to learn a unified representation space for both images and tags [20]. Recently, Li *et al.* [21] proposed a binary matrix factorization based hashing model for effective social image retrieve. All these models investigate how to utilize matrix factorization for social image understanding and retrieving, which study a different task but have close connections to our proposed model.

Social Recommendation. The ubiquitous social connections between users provide another auxiliary information to enhance recommendations [7], [22], [23]. Sociology studies showed that users would be influenced by their social neighbors [24], [25], so that a similar rating pattern could be found in a circle of friends [26]. In the following, we categorize various social recommender systems in terms of *traditional matrix factorization* [6], [7], [9], [10], [17] and *deep learning* [27], [28], [29].

Ma *et al.* [6] first proposed the matrix factorization based SoRec model to fuse social information by sharing latent vectors learned from rating matrix and social network, respectively. Jamali *et al.* [8] proposed the TrustWalker to adopt random walks to model trust propagation. Ma *et al.* [30] proposed the RSTE model, which integrated all user embeddings in a circle of friends to predict ratings. Ma *et al.* [7] proposed the SoReg model to regularize user latent preferences by social similarity. Yang *et al.* [31] proposed the TrustMF model, which differentiated trustors and trustees. Tang *et al.* [10] proposed the LOCABAL model, in which they jointly optimized a symmetric matrix tri-factorization to extract local social context and a weighted matrix factorization to exploit global social context and user preferences. Mirbakhsh *et al.* [32] proposed the CBSVD and CBASVD models, in which they first adopted k-means clustering to cluster users and items according to their latent preferences, and then used these obtained clusters to augment SVD++ and ASVD++ [33]. Zhang *et al.* [34] proposed the GeoSoCa model, in which they first learned statistics from historical rating records, social relations and geographical locations, and then made recommendations based on the learned statistics. Guo *et al.* [22] proposed the TrustSVD model to incorporate the trust influence from social neighbors on top of SVD++ [33]. Hu *et al.* [35] proposed the SSLSVD model, in which they adopted a transductive support vector machine to cluster social connections into trust and distrust, and then used the clustering results to augment TrustSVD. *All the above models are matrix factorization based, which aim to factorize rating matrix and social relation matrix simultaneously but in different ways.*

On the other hand, many studies on social recommendation have been developed using deep learning. Yang *et al.* [36] proposed a point-of-interest recommender system PACE, which simultaneously learned user preferences via a deep neural network and regularized learned preferences of socially connected users to be consistent. Fan *et al.* [27] proposed the GraphRec model to integrate deep neural networks and attention mechanism into social recommendation. Wu *et al.* [28] proposed the DANSER model to adopt dual graph attention networks to model user preferences from both rating information and social effects. Fan *et al.* [29] utilized a deep generative adversarial network to transfer users' information from the social domain to the item domain, while learning separated user representations in the two domains by adversarial learning. Recently, community structure has been incorporated into deep recommendation models. Li *et al.* [37] proposed to cluster users into communities to enhance graph neural networks for e-commerce recommendation. Li *et al.* [38] proposed to alternately stack graph neural network modules and perform k-means to cluster users and items, so as to obtain hierarchical embeddings of users and items. Maksimov *et al.* [39] introduced the multi-level item hierarchical graph to obtain different levels of item clusters to address the item cold-start problem. All these three models involve clustering users or items, but these clusters are obtained in the absence of social relations.

Community Detection. Traditional community detection methods mainly focus on partitioning a network using heuristic metrics, such as modularity, ratio cut, normalized cut, min-max cut, permanence, conductance, and so on [14], [40], [41]. Recently, Non-negative Matrix Factorization (NMF) has been widely adopted for community detection [42], [43], [44]. Psorakis *et al.* [42] proposed a Bayesian NMF model called BNMF to extract community information. Zhang *et al.* [45] proposed the Bounded Non-negative Matrix Tri-Factorization model (BNMTF) to detect overlapping communities in a network. Yang and Leskovec [46] developed an efficient NMF-based model named BigClam for overlapping community detection, which could handle large-scale networks. Wang *et al.* [15] proposed the M-NMF model to learn network embeddings based on modularity based community detection. Sun *et al.* [47] proposed a non-negative symmetric encoder-decoder approach for community detection. Ye *et al.* [48] proposed a novel deep autoencoder-like NMF model for community detection, which had extended Sun's work [47] to a deep neural network architecture.

Different from the above studies working on social recommendation and community detection independently, this paper investigates these two tasks simultaneously to give high-quality rating and ranking recommendations. There exist a very few studies involving community detection based social recommendation. The most related to ours is SoDimRec [17]. The SoDimRec model developed a two-stage recommendation scheme, i.e., first detecting communities between users, and then minimizing the embedding distances between users within the same community. However, in such a two-stage scheme, detected communities are not recommendation oriented, i.e., it is not guaranteed that those detected communities are the best for predicting ratings or recommending items, because they are detected in the absence of user-item interactions. In contrast, our proposed SCSVD model performs community detection and recommendation in a jointly learning manner, so that the detected communities can be adjusted to best regularize recommendation.

3 Preliminaries and Observations

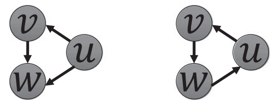
In this section, we first give preliminaries and then present our observations on real-life recommendation data.

3.1 Preliminaries

Throughout this paper, we use boldface capital letters to denote matrices, boldface lowercase letters to represent vectors, and italic lower case letters to express scalar values. For a given matrix $\mathbf{X}$, $\mathbf{x}^i$ and $\mathbf{x}_j$ are used to represent the ith row and jth column of $\mathbf{X}$, respectively, and x_{ij} is used to denote the (i,j)th entry of $\mathbf{X}$. Besides, we adopt $\mathrm{Tr}(\mathbf{X})$ and $\|\mathbf{X}\|_F$ to denote the trace and Frobenius norm of $\mathbf{X}$, respectively. The transpose of $\mathbf{X}$ is denoted by $\mathbf{X}^T$. The all-one vector is denoted as $\mathbf{1}$, the identity matrix whose diagonal elements are all ones is denoted as $\mathbf{I}$, and the Hadamard product (a.k.a. the element-wise product) is represented by $\odot$. Moreover, $\mathbf{X}^\dagger$ is used to denote the pseudo-inverse of $\mathbf{X}$. We use $\mathrm{diag}(\mathbf{x})$ to denote a diagonal matrix whose diagonal entries are composed of $\mathbf{x}$ in order, and we use $\mathbf{x} \parallel \mathbf{y}$ to denote that $\mathbf{x}$ and $\mathbf{y}$ are parallel. The inner product between $\mathbf{x}$ and $\mathbf{y}$ is denoted as $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^T\mathbf{y}$.

Let $\mathcal{U} = \{u_1, u_2, \ldots, u_n\}$ and $\mathcal{V} = \{v_1, v_2, \ldots, v_m\}$ be the sets of all users and items, respectively. Let $\mathbf{R} \in \mathbb{R}^{n\times m}$ be a user-item rating matrix, whose (i,j)th entry r_{ij} is the rating value, if u_i has given a rating to v_j, or 0 if not. We denote $\mathbf{u}_i \in \mathbb{R}^k$ and $\mathbf{v}_j \in \mathbb{R}^k$ as the user embedding for u_i and item embedding for v_j respectively, and $\mathbf{y}_j \in \mathbb{R}^k$ as the implicit item influence embedding [33] for v_j, where k is the number of latent factors. Besides, let $\mathbf{U} = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n] \in \mathbb{R}^{k\times n}$, $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_m] \in \mathbb{R}^{k\times m}$ and $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m] \in \mathbb{R}^{k\times m}$. Let p_i and q_j express the bias of u_i and v_j, respectively, and let μ represent the global mean rating of the dataset.

Let $\mathcal{G} = (\mathcal{U}, \mathcal{E})$ be a social network, whose number of users is $|\mathcal{U}| = n$ and number of edges is $|\mathcal{E}| = e$, where $\mathcal{U}$ represents the user set, and $\mathcal{E}$ represents the edge set, comprising all directed edges among all users. Typically, a social network is described by an adjacency matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, whose (i,j)th entry a_{ij} characterizes the directed social connection from user u_i to user u_j. In our social recommendation context, all relationships are unweighted, i.e., we have $a_{ij} = 1$, if there exists a directed edge between u_i and u_j, or $a_{ij} = 0$ otherwise. Besides, for an edge $\langle u_i, u_j \rangle \in \mathcal{E}$, we say u_i is an in-neighbor of u_j and u_j is an out-neighbor of u_i, and



(a) Trust triangle (b) Cycle triangle

Fig. 2. Illustration of trust triangle and cycle triangle.

we use $\mathcal{N}^+(u_i) = \{v : \langle u_i, v\rangle \in \mathcal{E}\}$ and $\mathcal{N}^-(u_i) = \{v : \langle v, u_i\rangle \in \mathcal{E}\}$ to represent the out-neighbors and in-neighbors of u_i, respectively.

3.2 Community Statistics and Observations

We start with an introduction of graph statistics in community structure, and then present our observations of these statistics over real-world datasets.

Triangle. A triangle is a clique of three vertices, which is known as a fundamental building block of undirected networks. There exist two kinds of triangles in directed graphs, including *trust triangles* and *cycle triangles*. A trust triangle $\triangle^T_{uvw}$ is composed of three vertices u, v, w having out degrees equal to two, one, and zero, respectively, as shown in Fig. 2a. A cycle triangle $\triangle^C_{uvw}$ is formed by three vertices having out-degrees of exactly one, respectively, as shown in Fig. 2b.

Average Clustering Coefficient. Average clustering coefficient evenly quantifies how close a node's neighbors are to being a clique. It is defined as $C = (1/n)\sum_{u\in\mathcal{U}} c_u$, where c_u is the individual clustering coefficient of node u defined as $c_u = T(u)/(k^{total}(u)(k^{total}(u)-1) - k^{\leftrightarrow}(u))$, where $T(u)$ is the number of all directed triangles (i.e., trust triangles and cycle triangles) containing vertex u, $k^{total}(u) = |\mathcal{N}^-(u)| + |\mathcal{N}^+(u)|$ is the sum of in-degree and out-degree of u and $k^{\leftrightarrow}(u) = |\mathcal{N}^-(u) \cap \mathcal{N}^+(u)|$ is the reciprocal degree of u. The upper bound of c_u is 1.

Expected Average Clustering Coefficient. Expected average clustering coefficient is the average clustering coefficient for a randomly generated directed graph whose nodes and node degree distributions are identical to the original graph.

We analyze social networks in three real-world datasets including Ciao, Epinions, and Flixster (for more details please refer to Section 5). Table 1 reports the detailed graph statistics of these datasets. We have the following observations:

- *Observation 1*. The average clustering coefficient is evidently greater than the expected average clustering coefficient on all these datasets, suggesting that there may exist a relatively compact cluster structure in all these datasets.
- *Observation 2*. A large number of trust triangles and cycle triangles widely exist in Ciao and Epinions datasets, demonstrating that close relationships within user triads are pervasive.
- *Observation 3*. A large amount of strongly and weakly connected components are involved in Epinions and Flixster datasets, indicating that there exist many close connections within not only user triads.

In summary, all the above observed evidences hint us a clustering perspective, i.e., considering the community structure in each social network.

TABLE 1
Graph Statistics of Three Real-World Datasets

Dataset	Ciao	Epinions	Flixster
Number of Users	7,375	40,163	10,000
Maximum Out-degree	804	1,723	28
Minimum In-degree	100	2,500	28
Number of Trust Triangles	799,556	4,291,145	3,575
Number of Cycle Triangles	172,147	921,583	1,181
Number of Strongly Connected Components	725	13,500	5,873
Number of Weakly Connected Components	65	904	5,740
Average Clustering Coefficient	**0.1397**	**0.1347**	**0.0283**
Expected Average Clustering Coefficient	0.0340	0.0194	0.0009

3.3 Hypothesis Testing

Most previous studies focus on utilizing pair-wise user interactions to improve recommendation quality. We are wondering if community structures, which are of a collective manner, are really gainful for providing useful information. Specifically, we are interested in the following question:

- Do users belonging to the same community have more similar rating patterns?

To answer the above question, we need to first define how communities are generated in our scheme, and what sort of users is called "having more similar rating patterns".

3.3.1 Community Detection

Generally, social relationships in product review sites are directed, i.e., users are not necessarily trusted mutually. With this in mind, in this paper, we adopt the conception of directed modularity optimization [49], which is one of the most widely used algorithms, to detect communities in a directed social network. Specifically, given a directed social network $\mathcal{G}$ with only two communities, the modularity is defined as $Q = (1/2e)\sum_{i,j=1}^{n}(a_{ij} - k_i^{\text{out}}k_j^{\text{in}}/e)h_ih_j$, where $k_i^{\text{out}} = |\mathcal{N}^+(u_i)|$ and $k_j^{\text{in}} = |\mathcal{N}^-(u_j)|$ are the out-degree of user u_i and in-degree of user u_j respectively, and h_i is the community indicator of u_i: $h_i = 1$ if u_i belongs to the first community, or $h_i = -1$ if u_i belongs to the second community. The quantity $k_i^{\text{out}}k_j^{\text{in}}/e$ is the expected number of edges between users u_i and u_j if edges are placed arbitrarily. According to the definition, modularity is the fraction of edges within communities minus the expected fraction of such edges when placed randomly. If we define a modularity matrix $\mathbf{B} \in \mathbb{R}^{n\times n}$ as $b_{ij} = a_{ij} - k_i^{\text{out}}k_j^{\text{in}}/e$, modularity Q can then be written as $Q = (1/2e)\mathbf{h}^T\mathbf{B}\mathbf{h}$, where the community membership vector $\mathbf{h} = [h_1, h_2, \ldots, h_n]^T$. However, the aforementioned scheme could only deal with two communities, which is almost meaningless in reality. In order to generalize such a model to more than two communities, we could define a membership indicator matrix $\mathbf{H} \in \mathbb{R}^{n\times c}$ whose rows are indicators specifying community assignments of users: $h_{ij} = 1$ if user i belongs to the jth community, or 0 if not. We denote $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_c]$, where $\mathbf{h}_i$ is a column vector indicating all members belonging to the ith community. It is noted that, according to the definition, each row of $\mathbf{H}$ has only one non-zero element, whose value

TABLE 2
t-Statistics and p-Values of t-Tests on Three Datasets

Dataset	Ciao [62]	Epinions [63]	Flixster [64]
t-statistic	25.7776	27.0802	8.4342
p-value	2.0797e-142	5.7510e-161	1.7897e-17

is 1, and all other elements are 0. Therefore, the community detection process for more than two communities could be boiled down to the following optimization problem:

$$\max_{\mathbf{H}} \ \mathrm{Tr}\big(\mathbf{H}^T\mathbf{B}\mathbf{H}\big) \qquad \text{s.t. each row of } \mathbf{H} \text{ is one-hot.}$$

Optimizing the above problem is NP-hard [50]. In practice, we could relax the constraint of the above optimization problem by allowing $\mathbf{H}$ to take any real value and imposing $\mathrm{Tr}\big(\mathbf{H}^T\mathbf{H}\big) = n$ on $\mathbf{H}$ [15], i.e., the original optimization problem is relaxed to be

$$\max_{\mathbf{H}} \ \mathrm{Tr}\big(\mathbf{H}^T\mathbf{B}\mathbf{H}\big) \qquad \text{s.t. } \mathrm{Tr}\big(\mathbf{H}^T\mathbf{H}\big) = n.$$

In addition, it is hard for us to tell whether zero entries in the adjacency matrix $\mathbf{A}$ correspond to total irrelevance or not. E.g., some users just have not issued such relationships which did exist. Therefore, we only select those non-zero entries in $\mathbf{A}$ to compute modularity, i.e., we aim to optimize the following problem in all our models

$$\max_{\mathbf{H}} \ \mathrm{Tr}\big(\mathbf{H}^T(\mathbf{A}\odot\mathbf{B})\mathbf{H}\big) \qquad \text{s.t. } \mathrm{Tr}\big(\mathbf{H}^T\mathbf{H}\big) = n. \tag{1}$$

Inspired by the optimization of the Rayleigh Quotient, the solution to the above problem is the eigenvectors corresponding to the c largest eigenvalues of $(\mathbf{A}\odot\mathbf{B} + (\mathbf{A}\odot\mathbf{B})^T)$.

In our hypothesis testing process, we set the number of communities to be detected, i.e., c, as the number of positive eigenvalues of $(\mathbf{A}\odot\mathbf{B} + (\mathbf{A}\odot\mathbf{B})^T)$, so as to optimize the modularity to be as large as possible. Besides, after obtaining real-valued $\mathbf{H}$ by eigenvalue decomposition, we conduct k-means clustering, where $k = c$, on rows of $\mathbf{H}$ to partition all users into c non-overlapping communities. These communities will be further utilized to examine whether users within the same community exhibit higher rating similarity. In the next section, we introduce the similarity metric used in our testing experiments.

3.3.2 User Similarity

Generally, the rating pattern of a user could be described by the user's rating records. Hence, we could adopt several rating similarity metrics to characterize the rating pattern similarity between users. In this paper, we adopt the Jaccard similarity to compute rating similarity between users u_i and u_j. Jaccard similarity is defined as $\mathrm{Jacc}(i,j) = |\mathcal{I}_i \cap \mathcal{I}_j|/|\mathcal{I}_i \cup \mathcal{I}_j|$, where $\mathcal{I}_i$ represents the set of items that u_i has consumed. According to the definition of Jaccard similarity, its value is ranging from 0 to 1, and a larger value indicates a closer relationship between the user pair. There are also a variety of other rating similarity metrics in the literature, such as Pearson correlation coefficient, cosine similarity, etc. However, in the following hypothesis testing, we only illustrate results of applying the Jaccard similarity, because we have verified that similar trends can be found by employing Pearson correlation coefficient or cosine similarity.

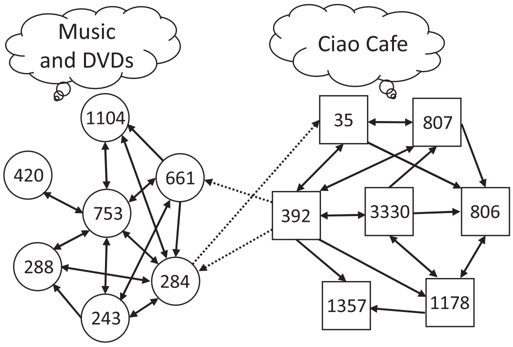


Fig. 3. Case study of two detected communities on Ciao.

3.3.3 Testing Settings and Results

To answer the question presented at the beginning of Section 3.3, we conduct Student's t-tests [51] on all the three datasets listed in Table 1. A detailed summary of testing procedure is described as follows.

1) Conducting community detection (optimization problem (1)) to obtain c non-overlapping communities in the social network. Denote the set of all c detected communities as $\mathcal{S} = \{s_1, s_2, \ldots, s_c\}$.
2) Computing intra-community and inter-community user similarities. Specifically, for each user $u_i \in \mathcal{U}$, we first retrieve the community that u_i belongs to, denoted as $s_{f(u_i)}$, where $f(\cdot)$ is a function that maps a user to the corresponding community index. Then, we compute two similarity values defined by Jaccard similarity:
   - The average similarity between u_i and all other users in $s_{f(u_i)}$, denoted as ζ_i.
   - The average similarity between u_i and $|s_{f(u_i)}| - 1$ randomly picked users where each user belongs to $\mathcal{S} \setminus s_{f(u_i)}$, denoted as ξ_i.
3) Conducting a two-sample t-test on $\boldsymbol{\zeta}$ and $\boldsymbol{\xi}$, where $\boldsymbol{\zeta} = [\zeta_1, \zeta_2, \ldots, \zeta_n]$ and $\boldsymbol{\xi} = [\xi_1, \xi_2, \ldots, \xi_n]$. The null hypothesis is $H_0 : \boldsymbol{\zeta} = \boldsymbol{\xi}$, and the alternative hypothesis is $H_1 : \boldsymbol{\zeta} > \boldsymbol{\xi}$.

The experimental results are shown as below.

For all these three datasets, the null hypothesis is rejected at significance level $\alpha = 0.01$ with p-value close to nothing. The results suggest a significant positive impact of detected communities in clustering users with more similar rating patterns. In the next section, we will present a case study to show the physical meaning of these detected communities and how they can help improve recommendation.

3.4 Case Study

To reveal physical meanings of the detected communities, we visualize two detected communities on Ciao dataset in Fig. 3. These two communities are highlighted in circles and squares, respectively. The number in each node represents its ID. The nodes in the same shape belong to the same community. The keywords associated with each community represent the common interests of community members.[1] The circle community is an interest group on Music and DVDs; on the other hand, the square community is an interest group on Ciao Cafe. The common interests are extracted by observing the detailed consumption histories of these community members, which can be found in our supplemental material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TKDE.2021.3117686. As observed, users within the same community tend to favor a specific type of products, which shows their homogeneous preferences. Besides, as the consumption histories in our supplemental material, available online, convey, nearly every user pair within the same community has common items (e.g., user #35 and user #392 purchased the same items #1336, #5236 and #2750), indicating that the users within the same community share more similar preferences. Similar observations can be found in other detected communities.

In conclusion, the physical meaning of the detected communities is that users within the same community share more similar preferences. As a consequence, the underlying community structure in a social network, which can be obtained by exploiting the modularity matrix $\mathbf{B}$, provides a plausible direction to regularize user embeddings: regularizing users within the same community with consistent latent factors. This motivation would lead to a better user preference modeling, and eventually benefit the performance of recommendation.

4 SCSVD Model

In this section, we first introduce the SCSVD model for community detection based social recommendation. We then develop optimization techniques for SCSVD and give theoretical analysis on SCSVD, in terms of convergence, algorithm complexity and also the correlation between community detection and embedding learning.

4.1 Backbones of a Latent Factor Model

Our SCSVD model is built on top of the SVD++ model [33], which is a state-of-the-art latent factor model. We adopt a combination of SVD++ and modularity maximization for community-aware social recommendation. Besides this combination, there are other choices for community detection based social recommendation. One approach is to combine simple spectral clustering with traditional matrix factorization, both of which can be optimized efficiently. Furthermore, one can also combine neural network based recommendation schemes [52] and neural network based community detection methods [53] to exploit higher-order correlations. We leave the exploration for other combinations as our future work. Nevertheless, the most widely used and successful approach for collaborative filtering is the latent factor model [33], [54], which maps both users and items into a common latent factor space. Therefore, we establish the foundations of a latent factor model and introduce SVD++ first.

For the most naive latent factor model, an unknown rating r_{ij} is modeled as the inner product of the corresponding user embedding $\mathbf{u}_i$ and item embedding $\mathbf{v}_j$, i.e., the rating is predicted as $\hat{r}_{ij} = \mathbf{u}_i^T\mathbf{v}_j$. Generally, a ridge regression loss function is adopted to learn $\mathbf{U}$ and $\mathbf{V}$, and stochastic gradient descent and alternating least squares are usually adopted to optimize the loss function. However, such a naive latent factor model suffers from a serious problem: it only considers the interactions between users and items, but neglects the biases of users and items. For instance, in a product review scenario, some users tend to give higher ratings but some do not, while some items are likely to receive higher ratings but some are in contrast, and among different e-commerce websites, each site's average rating may also be different. A simple but effective way to resolve the aforementioned problem is to integrate biases into the above latent factor model, i.e., predicted rating is computed as $\hat{r}_{ij} = \mu + p_i + q_j + \mathbf{u}_i^T\mathbf{v}_j$, where μ represents the global mean rating, and p_i and q_j stand for the bias of user u_i and item v_j respectively. Furthermore, the implicit feedbacks, i.e., users' purchase records without their concrete rating values, are also proven to be conducive for recommendation [33]. To incorporate the implicit feedbacks, another set of item latent vectors, $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m]$, is introduced, wherein $\mathbf{y}_j$ is related to v_j. These newly introduced item latent vectors are utilized to characterize users' preferences based on what they have bought, regardless of their ratings. The final model is called SVD++ [33], which takes the form

$$\hat{r}_{ij} = \mu + p_i + q_j + \mathbf{v}_j^T\left(\mathbf{u}_i + \frac{1}{\sqrt{|\mathcal{I}_i|}}\sum_{l\in\mathcal{I}_i}\mathbf{y}_l\right). \tag{2}$$

In the next section, we will present our SCSVD model, which utilizes modularity maximization to regularize SVD++.

4.2 A Unified Framework

We now present our unified framework SCSVD for community detection based social recommendation. Our model seeks to optimize SVD++ (Eq. (2)) and community detection (problem (1)) simultaneously, while building a bilateral connection between these two problems at the same time. In this way, the underlying community structure can be exploited to guide the process of user latent preference modeling. However, the constraint $\mathrm{Tr}(\mathbf{H}^T\mathbf{H}) = n$ in problem (1) is still difficult to optimize, therefore we relax again the constraint to the combination of $\mathbf{H}^T\mathbf{H} = \mathbf{I}$ and $\mathbf{H} \geq 0$. In order to build a connection between user embeddings $\mathbf{U}$ and community memberships $\mathbf{H}$, we adopt a matrix $\mathbf{C}$ to map them linearly [15]. Besides, as suggested by [55], [56], a weighted λ-regularization scheme is also introduced to our objective function to avoid over-fitting. In summary, the optimization problem of our SCSVD model is formulated as follows:

$$\begin{aligned}\min_{\mathbf{p},\mathbf{q},\mathbf{U},\mathbf{V},\mathbf{Y},\mathbf{C},\mathbf{H}} \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\big(r_{ij} - \hat{r}_{ij}\big)^2 + \frac{\alpha}{2}\left\|\mathbf{H} - \mathbf{U}^T\mathbf{C}\right\|_F^2 \\ & - \frac{\beta}{2}\mathrm{Tr}\big(\mathbf{H}^T(\mathbf{A}\odot\mathbf{B})\mathbf{H}\big) + \frac{\gamma}{2}(\|\mathbf{U}\mathbf{\Omega}_{\mathbf{U}}\|_F^2 \\ & + \|\mathbf{V}\mathbf{\Omega}_{\mathbf{V}}\|_F^2 + \|\mathbf{Y}\mathbf{\Omega}_{\mathbf{Y}}\|_F^2 + \|\mathbf{\Omega}_{\mathbf{U}}\mathbf{p}\|_2^2 + \|\mathbf{\Omega}_{\mathbf{V}}\mathbf{q}\|_2^2) \\ \text{s.t.} \quad & \mathbf{H}^T\mathbf{H} = \mathbf{I}, \mathbf{H} \geq 0,\end{aligned} \tag{3}$$

where $\mathbf{\Omega}_{\mathbf{U}} = \mathrm{diag}\ ([\sqrt{|\mathcal{I}_1|},\ \sqrt{|\mathcal{I}_2|},\ \ldots,\ \sqrt{|\mathcal{I}_n|}]^T)$, $\mathbf{\Omega}_{\mathbf{V}} = \mathrm{diag}\ ([\sqrt{|\mathcal{U}_1|},\ \sqrt{|\mathcal{U}_2|},\ \ldots,\ \sqrt{|\mathcal{U}_m|}]^T)$ and $\mathbf{\Omega}_{\mathbf{Y}} = (\mathrm{diag}\ (\mathbf{W}^T\mathbf{\Omega}_{\mathbf{U}}^2\mathbf{1}))^{\frac{1}{2}} = \mathrm{diag}([\sqrt{\sum_{i=1}^n w_{i1}|\mathcal{I}_i|},\ \sqrt{\sum_{i=1}^n w_{i2}|\mathcal{I}_i|},\ \ldots,\ \sqrt{\sum_{i=1}^n w_{im}|\mathcal{I}_i|}]^T)$. Besides, $\mathcal{I}_i$ is the set of all items that user u_i has interacted with, $\mathcal{U}_j$ is the set of all users who have consumed item v_j, and $\mathbf{W}$ is defined as $w_{ij} = 1$ if $r_{ij} > 0$, or 0 if not.

1. For a detailed list of all categories (such as DVDs), readers may refer to https://www.cse.msu.edu/~tangjili/datasetcode/catalog_ciao.txt.

The motivation of the term $\|\mathbf{H}-\mathbf{U}^T\mathbf{C}\|_F^2$ is to build a bridge between the user latent factor space and the community membership space. As we will discuss in Section 4.6, this term can guide user embeddings to preserve the underlying community structure, so as to obtain a better user preference modeling. However, since there is no any guarantee about the definiteness of $\mathbf{A}\odot\mathbf{B}$, the above optimization problem is non-convex with respect to $\mathbf{H}$, which is hard to optimize.

4.3 Optimization Scheme

Alternatively, we propose an equivalent framework of problem (3) as follows:

$$\begin{aligned}\min_{\mathbf{p},\mathbf{q},\mathbf{U},\mathbf{V},\mathbf{Y},\mathbf{C},\mathbf{H},\mathbf{F}} \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\big(r_{ij}-\hat{r}_{ij}\big)^2+\frac{\alpha}{2}\big\|\mathbf{H}-\mathbf{U}^T\mathbf{C}\big\|_F^2 \\ & -\frac{\beta}{2}\mathrm{Tr}\big(\mathbf{H}^T(\mathbf{A}\odot\mathbf{B})\mathbf{F}\big)+\frac{\eta}{2}\|\mathbf{F}-\mathbf{H}\|_F^2+\frac{\gamma}{2}(\|\mathbf{U}\mathbf{\Omega}_\mathbf{U}\|_F^2 \\ & +\|\mathbf{V}\mathbf{\Omega}_\mathbf{V}\|_F^2+\|\mathbf{Y}\mathbf{\Omega}_\mathbf{Y}\|_F^2+\|\mathbf{\Omega}_\mathbf{U}\mathbf{p}\|_2^2+\|\mathbf{\Omega}_\mathbf{V}\mathbf{q}\|_2^2) \\ \text{s.t.} \quad & \mathbf{H}^T\mathbf{H}=\mathbf{I}, \mathbf{F}\geq 0,\end{aligned} \tag{4}$$

where η is a large enough number to ensure the equality between $\mathbf{H}$ and $\mathbf{F}$.

Introducing an instrumental matrix $\mathbf{F}$ has an important benefit: each of $\mathbf{H}$ and $\mathbf{F}$ in problem (4) has only one single constraint, which alleviates the difficulty of optimization. Besides, the objective function in problem (4) is now convex in each single variable when other variables are fixed. We next separate the optimization of (4) into four subproblems, and they will be solved in an iterative manner.

Updating $\mathbf{p},\mathbf{q},\mathbf{U},\mathbf{V},\mathbf{Y}$. The subproblem which only relates to $\mathbf{p},\mathbf{q},\mathbf{U},\mathbf{V},\mathbf{Y}$ is

$$\begin{aligned}\min_{\mathbf{p},\mathbf{q},\mathbf{U},\mathbf{V},\mathbf{Y}} \quad & \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\big(r_{ij}-\hat{r}_{ij}\big)^2+\frac{\alpha}{2}\big\|\mathbf{H}-\mathbf{U}^T\mathbf{C}\big\|_F^2 \\ +\frac{\gamma}{2}(\|\mathbf{U}\mathbf{\Omega}_\mathbf{U}\|_F^2 & +\|\mathbf{V}\mathbf{\Omega}_\mathbf{V}\|_F^2+\|\mathbf{Y}\mathbf{\Omega}_\mathbf{Y}\|_F^2+\|\mathbf{\Omega}_\mathbf{U}\mathbf{p}\|_2^2+\|\mathbf{\Omega}_\mathbf{V}\mathbf{q}\|_2^2).\end{aligned}$$

The only difference between the above subproblem and the SVD++ is the mapping term $\big\|\mathbf{H}-\mathbf{U}^T\mathbf{C}\big\|_F^2$. As suggested by [22], [33], [57], we adopt a gradient descent method to optimize $\mathbf{p},\mathbf{q},\mathbf{U},\mathbf{V},\mathbf{Y}$. Specifically, denote the objective function of problem (4) as $\mathcal{J}$, the gradient with respect to the aforementioned five variables are

$$\frac{\partial\mathcal{J}}{\partial p_i}=\sum_{j=1}^{m} w_{ij}\big(\hat{r}_{ij}-r_{ij}\big)+\gamma|\mathcal{I}_i|p_i,$$

$$\frac{\partial\mathcal{J}}{\partial q_j}=\sum_{i=1}^{n} w_{ij}\big(\hat{r}_{ij}-r_{ij}\big)+\gamma|\mathcal{U}_j|q_j,$$

$$\frac{\partial\mathcal{J}}{\partial \mathbf{u}_i}=\sum_{j=1}^{m} w_{ij}\big(\hat{r}_{ij}-r_{ij}\big)\mathbf{v}_j+\alpha\big(\mathbf{C}\mathbf{C}^T\mathbf{U}-\mathbf{C}\mathbf{H}^T\big)_i+\gamma|\mathcal{I}_i|\mathbf{u}_i,$$

$$\frac{\partial\mathcal{J}}{\partial \mathbf{v}_j}=\sum_{i=1}^{n} w_{ij}\big(\hat{r}_{ij}-r_{ij}\big)\left(\mathbf{u}_i+\frac{1}{\sqrt{|\mathcal{I}_i|}}\sum_{k\in\mathcal{I}_i}\mathbf{y}_k\right)+\gamma|\mathcal{U}_j|\mathbf{v}_j,$$
$$\frac{\partial\mathcal{J}}{\partial \mathbf{y}_k}=\sum_{i=1}^{n} w_{ik}\sum_{j=1}^{m} w_{ij}\big(\hat{r}_{ij}-r_{ij}\big)\frac{1}{\sqrt{|\mathcal{I}_i|}}\mathbf{v}_j+\gamma\sum_{\iota\in\mathcal{U}_k}|\mathcal{I}_\iota|\mathbf{y}_k.$$

It is noted that, except for $\mathbf{u}_i$, the update rules for all these variables could be disassembled into each training instance (u_i, v_j, r_{ij}). For $\mathbf{u}_i$, we update the term $\alpha(\mathbf{C}\mathbf{C}^T\mathbf{U}-\mathbf{C}\mathbf{H}^T)_i$ after all items in $\mathcal{I}_i$ are gone through, so as to make the update rule for $\mathbf{u}_i$ could be disassembled to each instance as well.

Updating $\mathbf{C}$. The subproblem that only associates to $\mathbf{C}$ and the corresponding analytical solution to this subproblem are

$$\min_{\mathbf{C}}\ \big\|\mathbf{H}-\mathbf{U}^T\mathbf{C}\big\|_F^2 \text{ and } \mathbf{C}=\big(\mathbf{U}\mathbf{U}^T\big)^{-1}\mathbf{U}\mathbf{H}.$$

Consider the fact that $n>k$ is consistently established in reality, the non-singularity of $\mathbf{U}\mathbf{U}^T$ is guaranteed, and this ensures the stability of the update rule for $\mathbf{C}$.

Updating $\mathbf{H}$. The subproblem that only relates to $\mathbf{H}$ is

$$\begin{aligned}&\min_{\mathbf{H}}\frac{\alpha}{2}\big\|\mathbf{H}-\mathbf{U}^T\mathbf{C}\big\|_F^2-\frac{\beta}{2}\mathrm{Tr}\big(\mathbf{H}^T(\mathbf{A}\odot\mathbf{B})\mathbf{F}\big)+\frac{\eta}{2}\|\mathbf{F}-\mathbf{H}\|_F^2\\ &\text{s.t. } \mathbf{H}^T\mathbf{H}=\mathbf{I}.\end{aligned}$$

After mathematical manipulation, the above problem can be converted to an equivalent problem as

$$\min_{\mathbf{H}}\left\|\mathbf{H}-\frac{\alpha\mathbf{U}^T\mathbf{C}+\frac{\beta}{2}(\mathbf{A}\odot\mathbf{B})\mathbf{F}+\eta\mathbf{F}}{\alpha+\eta}\right\|_F^2 \text{ s.t. } \mathbf{H}^T\mathbf{H}=\mathbf{I}. \tag{5}$$

To optimize problem (5), we introduce a lemma.

Lemma 1 ([58]). *Suppose we have two matrices* $\mathbf{P}\in\mathbb{R}^{n\times m}$ *and* $\mathbf{Q}\in\mathbb{R}^{n\times d}$. *The optimization problem*

$$\min_{\tilde{\mathbf{T}}}\ \big\|\mathbf{P}\tilde{\mathbf{T}}-\mathbf{Q}\big\|_F^2 \qquad \text{s.t. } \tilde{\mathbf{T}}\tilde{\mathbf{T}}^T=\mathbf{I},$$

has an analytical solution $\tilde{\mathbf{T}}=\mathbf{U}\mathbf{I}_{m,d}\mathbf{V}^T$, *where* $\mathbf{U}\in\mathbb{R}^{m\times m}$ *and* $\mathbf{V}\in\mathbb{R}^{d\times d}$ *are left and right eigenvectors of* $\mathbf{P}^T\mathbf{Q}$, *computed by singular value decomposition, respectively.*

The solution to problem (5) can be straightly obtained by applying Lemma 1 with $\mathbf{P}=\mathbf{I}$, $\tilde{\mathbf{T}}=\mathbf{H}^T$ and $\mathbf{Q}=\alpha\mathbf{C}^T\mathbf{U}/(\alpha+\eta)+\beta\mathbf{F}^T(\mathbf{A}\odot\mathbf{B})^T/2(\alpha+\eta)+\eta\mathbf{F}^T/(\alpha+\eta)$, as $\mathbf{H}=\mathbf{V}_\mathbf{H}\mathbf{I}_{n,c}\mathbf{U}_\mathbf{H}^T$, where $\mathbf{U}_\mathbf{H}\in\mathbb{R}^{c\times c}$ and $\mathbf{V}_\mathbf{H}\in\mathbb{R}^{n\times n}$ are left and right eigenvectors of $\mathbf{P}^T\mathbf{Q}$, computed by singular value decomposition, respectively.

Updating $\mathbf{F}$. The subproblem that is only relevant to $\mathbf{F}$ is

$$\min_{\mathbf{F}}-\frac{\beta}{2}\mathrm{Tr}\big(\mathbf{H}^T(\mathbf{A}\odot\mathbf{B})\mathbf{F}\big)+\frac{\eta}{2}\|\mathbf{F}-\mathbf{H}\|_F^2 \text{ s.t. } \mathbf{F}\geq 0.$$

After mathematical manipulation, the above problem can be converted to

$$\min_{\mathbf{F}}\left\|\mathbf{F}-\left(\mathbf{H}+\frac{\beta}{2\eta}(\mathbf{A}\odot\mathbf{B})^T\mathbf{H}\right)\right\|_F^2 \qquad \text{s.t. } \mathbf{F}\geq 0,$$

and the solution to this problem is straightforward, i.e.,

$$\mathbf{F}=\max\left(\mathbf{H}+\frac{\beta}{2\eta}(\mathbf{A}\odot\mathbf{B})^T\mathbf{H},0\right).$$

Based on the above analysis, we summarize the detailed optimization algorithm in Algorithm 1. As observed, Algorithm 1 iteratively optimizes SCSVD until certain stopping criteria are met. In line 1, it initializes all factors and constructs $\mathbf{A}$ and $\mathbf{B}$. From line 3 to line 14, it updates user and item embeddings by stochastic gradient descent. In line 15, it solves the least squares problem to update $\mathbf{C}$. In line 16, it optimizes $\mathbf{H}$ by singular value decomposition. In line 17, it updates $\mathbf{F}$ via a thresholding function.

Algorithm 1. The Optimization Algorithm for SCSVD

Input: The rating information $\mathbf{R}$, the social network $\mathcal{G}$, the number of latent factors k, the number of communities c, the number of maximum iterations Φ, learning rate θ, and parameters α, β, γ, η. (Denote $\hat{r}_{ij}$ as in Eq. (4).)

Output: $\mathbf{p}, \mathbf{q}, \mathbf{U}, \mathbf{V}, \mathbf{Y}$

1: Initialize $\mathbf{p}, \mathbf{q}, \mathbf{U}, \mathbf{V}, \mathbf{Y}, \mathbf{C}, \mathbf{H}, \mathbf{F}$ randomly, extract adjacency matrix $\mathbf{A}$ from $\mathcal{G}$, and compute modularity matrix $\mathbf{B}$
2: **while** number of iterations $\leq \Phi$ and not converged **do**
3:   **for** i in $\mathcal{U}$ **do**
4:     **for** j in $\mathcal{I}_i$ **do**
5:       Update $p_i \leftarrow p_i - \theta\big(\big(\hat{r}_{ij} - r_{ij}\big) + \gamma p_i\big)$
6:       Update $q_j \leftarrow q_j - \theta\big(\big(\hat{r}_{ij} - r_{ij}\big) + \gamma q_j\big)$
7:       Update $\mathbf{u}_i \leftarrow \mathbf{u}_i - \theta\big(\big(\hat{r}_{ij} - r_{ij}\big)\mathbf{v}_j + \gamma\mathbf{u}_i\big)$
8:       Update $\mathbf{v}_j \leftarrow \mathbf{v}_j - \theta((\hat{r}_{ij} - r_{ij})(\mathbf{u}_i + \sum_{k\in\mathcal{I}_i}\mathbf{y}_k/\sqrt{|\mathcal{I}_i|}) + \gamma\mathbf{v}_j)$
9:       **for** $k \in \mathcal{I}_i$ **do**
10:         Update $\mathbf{y}_k \leftarrow \mathbf{y}_k - \theta((\hat{r}_{ij} - r_{ij})\mathbf{v}_j/\sqrt{|\mathcal{I}_i|} + \gamma\mathbf{y}_k)$
11:       **end for**
12:     **end for**
13:     Update $\mathbf{u}_i \leftarrow \mathbf{u}_i - \theta\alpha(\mathbf{C}\mathbf{C}^T\mathbf{u}_i - (\mathbf{C}\mathbf{H}^T)_i)$
14:   **end for**
15:   Update $\mathbf{C} \leftarrow \big(\mathbf{U}\mathbf{U}^T\big)^{-1}\mathbf{U}\mathbf{H}$
16:   Compute $\mathbf{U}_\mathbf{H}$ and $\mathbf{V}_\mathbf{H}$ as the left and right eigenvectors of $\alpha\mathbf{C}^T\mathbf{U}/(\alpha+\eta) + \beta\mathbf{F}^T(\mathbf{A}\odot\mathbf{B})^T/2(\alpha+\eta) + \eta\mathbf{F}^T/(\alpha+\eta)$, and update $\mathbf{H} \leftarrow \mathbf{V}_\mathbf{H}\mathbf{I}_{n,c}\mathbf{U}_\mathbf{H}^T$
17:   Update $\mathbf{F} \leftarrow \max(\mathbf{H} + (\beta/2\eta)(\mathbf{A}\odot\mathbf{B})^T\mathbf{H}, 0)$
18: **end while**
19: **return** $\mathbf{p}, \mathbf{q}, \mathbf{U}, \mathbf{V}, \mathbf{Y}$

4.4 Convergence Analysis

We now give a convergence guarantee for Algorithm 1. For convenience, we denote the objective function in problem (4) as $\mathcal{J}(\mathbf{p}, \mathbf{q}, \mathbf{U}, \mathbf{V}, \mathbf{Y}, \mathbf{C}, \mathbf{H}, \mathbf{F})$.

Theorem 2. *$\mathcal{J}(\mathbf{p}, \mathbf{q}, \mathbf{U}, \mathbf{V}, \mathbf{Y}, \mathbf{C}, \mathbf{H}, \mathbf{F})$ is convex to each variable. Besides, given a small enough number of learning rate θ, Algorithm 1 will monotonically decrease $\mathcal{J}(\mathbf{p}, \mathbf{q}, \mathbf{U}, \mathbf{V}, \mathbf{Y}, \mathbf{C}, \mathbf{H}, \mathbf{F})$ in each iteration, and converge to a local minimum of the problem.*

Proof. Please refer to our supplemental material, available online. □

4.5 Complexity Analysis

We now give the time complexity of Algorithm 1.

Theorem 3. *The overall time complexity of Algorithm 1 is $\mathcal{O}(\Phi(m^2nk + cn^2))$, where Φ is the iteration number, n is the number of users, m is the number of items, k is the number of latent factors, and c is the number of communities.*

Proof. Please refer to our supplemental material, available online. □

4.6 Theoretical Analysis on the Interaction Between Embedding Learning and Community Detection

In this section, we theoretically analyze the effects of the linear mapping term $\|\mathbf{H} - \mathbf{U}^T\mathbf{C}\|_F^2$ presented in problem (3). We use $\mathbf{h}_i$ and $\mathbf{h}^j$ to denote ith column and jth row of $\mathbf{H} \in \mathbb{R}^{n\times c}$ respectively. Recall that $\mathbf{H} = [\,\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_c\,] = \big[\,(\mathbf{h}^1)^T, (\mathbf{h}^2)^T, \ldots, (\mathbf{h}^n)^T\big]^T$, and $\mathbf{U} = [\,\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n\,]$. When $\|\mathbf{H} - \mathbf{U}^T\mathbf{C}\|_F^2$ is minimized after training, we have $\mathbf{H}^T \approx \mathbf{C}^T\mathbf{U}$, i.e., $\big[\,(\mathbf{h}^1)^T, (\mathbf{h}^2)^T, \ldots, (\mathbf{h}^n)^T\,\big] \approx \mathbf{C}^T[\,\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n\,]$, which implies that

$$\big(\mathbf{h}^1\big)^T \approx \mathbf{C}^T\mathbf{u}_1, \big(\mathbf{h}^2\big)^T \approx \mathbf{C}^T\mathbf{u}_2, \ldots \big(\mathbf{h}^n\big)^T \approx \mathbf{C}^T\mathbf{u}_n. \tag{6}$$

Next, we discuss two effects brought by the linear mapping.

On one hand, user embeddings $\mathbf{U}$ would be influenced by the community structure. According to [59], the orthogonal and non-negative constraints imposed on $\mathbf{H}$ make each row of $\mathbf{H}$ has only one non-zero element, i.e., ideally, $\mathbf{h}^i$ contains only one non-zero element which indicates the community index that u_i belongs to, but we can not expect the magnitude of that element. However, it insinuates a covert message that, community indicator vectors of those users in the same community are parallel. Specifically, denote a community as $\mathcal{C} = \big\{u_{g(1)}, u_{g(2)}, \ldots, u_{g(|\mathcal{C}|)}\big\}$, where $g(\cdot)$ is a function that maps user indices in $\mathcal{C}$ to user indices in $\mathcal{U}$. Then ideally, by Eq. (6), we have the following property:

$$\begin{cases} \kappa_1\mathbf{h}_\mathcal{C} = \Big(\mathbf{h}^{g(1)}\Big)^T \approx \mathbf{C}^T\mathbf{u}_{g(1)}, \\ \kappa_2\mathbf{h}_\mathcal{C} = \Big(\mathbf{h}^{g(2)}\Big)^T \approx \mathbf{C}^T\mathbf{u}_{g(2)}, \\ \cdots \\ \kappa_{|\mathcal{C}|}\mathbf{h}_\mathcal{C} = \Big(\mathbf{h}^{g(|\mathcal{C}|)}\Big)^T \approx \mathbf{C}^T\mathbf{u}_{g(|\mathcal{C}|)}, \end{cases} \tag{7}$$

where $\mathbf{h}_\mathcal{C}$ is any column vector paralleling to each community indicator $(\mathbf{h}^{g(i)})^T$ for all $u_{g(i)} \in \mathcal{C}$, and κ_i's are corresponding scale factors. From property (7), we could draw a conclusion that $\mathbf{C}^T\mathbf{u}_{g(i)}$'s are parallel to each other for all $u_{g(i)} \in \mathcal{C}$, denoted as $\mathbf{C}^T\mathbf{u}_{g(1)} \parallel \mathbf{C}^T\mathbf{u}_{g(2)} \parallel \cdots \parallel \mathbf{C}^T\mathbf{u}_{g(|\mathcal{C}|)}$. According to [60], linear transformations preserve parallelism. From this perspective, we define a linear transformation as $(\mathbf{C}^T)^\dagger = (\mathbf{C}\mathbf{C}^T)^{-1}\mathbf{C}$. Since $\mathbf{C} \in \mathbb{R}^{k\times c}$ and the latent dimensionality k is consistently smaller than the number of communities c, the non-singularity of $\mathbf{C}\mathbf{C}^T \in \mathbb{R}^{k\times k}$ is guaranteed and the existence of $(\mathbf{C}^T)^\dagger$ is assured. Transforming $\mathbf{C}^T\mathbf{u}_{g(i)}$'s by $(\mathbf{C}^T)^\dagger$, we can obtain that

$$\mathbf{u}_{g(1)} \parallel \mathbf{u}_{g(2)} \parallel \cdots \parallel \mathbf{u}_{g(|\mathcal{C}|)}.$$

The above property is important, since it demonstrates that the model is forcing user embeddings corresponding to the same community to be parallel, i.e., user latent vectors corresponding to the same community are forced to have homogeneous preference distributions on the k latent factors. Ideally, despite that the absolute magnitudes of users' preferences on a certain latent factor would be different, the overall preference distributions of these users on all k latent factors are forced to be identical. Compared to the SoReg [7], which forces user embeddings corresponding to a circle of friends to be close in terms of the euclidean distance, our approach enhances the expression ability of recommendation models. E.g., latent vectors with different lengths but

TABLE 3
Overall Statistics of Three Real Datasets

Statistic	Ciao	Epinions	Flixster	Statistic	Ciao	Epinions	Flixster
# of Users	7,375	40,163	10,000	# of Trustors	6,792	27,681	4,508
# of Items	105,114	139,738	19,342	# of Trustees	7,297	39,003	4,544
# of Ratings	284,086	664,823	457,882	# of Relations	111,781	442,979	12,144
Rating Density	0.037%	0.012%	0.237%	Relation Density	0.226%	0.041%	0.059%

similar directions could capture user biases in another phase while preserving their preference similarity. Although the graph structure of communities detected by SCSVD may be not as cohesive as the community-oriented algorithms (e.g., BigClam [46]), our communities can capture similar features in users' preferences, which indeed improves the recommendation task of rating prediction.

On the other hand, $\mathbf{H}$ could also assimilate information embedded in user latent preferences. As discussed, under ideal circumstances, $\mathbf{h}^i$ contains only one positive element, hence we could measure the correlation between $\mathbf{h}^i$ and $\mathbf{h}^j$ by their inner product. Ideally, when $\langle \mathbf{h}^i, \mathbf{h}^j \rangle$ is a positive number, user u_i and user u_j would be grouped together; when $\langle \mathbf{h}^i, \mathbf{h}^j \rangle$ is zero, the two users would be isolated. Therefore, in reality, a bigger value of $\langle \mathbf{h}^i, \mathbf{h}^j \rangle$ indicates a higher propensity that u_i and u_j are clustered together. Let us consider the term $\langle \mathbf{h}^i, \mathbf{h}^j \rangle \approx \mathbf{u}_i^T (\mathbf{C}\mathbf{C}^T) \mathbf{u}_j$ achieved by Eq. (6). Since $\mathbf{C}\mathbf{C}^T$ is a positive semi-definite matrix, it could be diagonalized by eigenvalue decomposition with all non-negative eigenvalues, and all its eigenvectors could be chosen to be orthonormal. From this, we could obtain another relation as

$$\mathbf{u}_i^T (\mathbf{C}\mathbf{C}^T) \mathbf{u}_j = \mathbf{u}_i^T (\mathbf{Q}^T \mathbf{\Lambda} \mathbf{Q}) \mathbf{u}_j = (\mathbf{Q}\mathbf{u}_i)^T \mathbf{\Lambda} (\mathbf{Q}\mathbf{u}_j),$$

where $\mathbf{Q}$ is an orthogonal matrix whose rows are the eigenvectors of $\mathbf{C}\mathbf{C}^T$, and $\mathbf{\Lambda}$ is a diagonal matrix whose entries are the eigenvalues of $\mathbf{C}\mathbf{C}^T$ [61]. By extracting diagonal elements in $\mathbf{\Lambda}$, the above relation could be further derived as

$$(\mathbf{Q}\mathbf{u}_i)^T \mathbf{\Lambda} (\mathbf{Q}\mathbf{u}_j) = (\mathbf{\Lambda}\mathbf{1})^T (\mathbf{Q}\mathbf{u}_i \odot \mathbf{Q}\mathbf{u}_j) = \sum_{f=1}^{k} \lambda_f (\mathbf{Q}\mathbf{u}_i)_f (\mathbf{Q}\mathbf{u}_j)_f,$$

where $\mathbf{\Lambda}\mathbf{1}$ is a column vector comprising all diagonal entries in $\mathbf{\Lambda}$ (i.e., all eigenvalues), and λ_f is the fth eigenvalue of $\mathbf{\Lambda}$. Since $\mathbf{\Lambda}\mathbf{1}$ is a positive vector and $\mathbf{Q}$ represents an orthogonal transformation which preserves lengths of vectors and angles between them, the term $\sum_{f=1}^{k} \lambda_f (\mathbf{Q}\mathbf{u}_i)_f (\mathbf{Q}\mathbf{u}_j)_f$ could be interpreted as a weighted inner product that characterizes the similarity between user u_i and user u_j with different attention, which is described by the spectrum of $\mathbf{C}\mathbf{C}^T$, paid to different rotated latent factors. So far, we have derived an important relation that

$$\langle \mathbf{h}^i, \mathbf{h}^j \rangle \approx \sum_{f=1}^{k} \lambda_f (\mathbf{Q}\mathbf{u}_i)_f (\mathbf{Q}\mathbf{u}_j)_f.$$

The bigger the similarity, the larger the value $\langle \mathbf{h}^i, \mathbf{h}^j \rangle$ and the higher the tendency of the two users being clustered together. Therefore, from this perspective, similar users under our criteria are likely to be clustered together. This mechanism reasonably incorporates the rating information into the detected communities, thereby increasing their homogeneity, not only in terms of social connections, but also in terms of similarities defined in the k-dimensional latent factor space.

5 Experimental Results

5.1 Experimental Settings

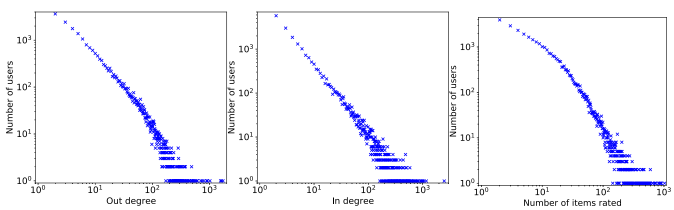
5.1.1 Datasets

We use the following three real-world product review datasets in our experiments: Ciao [62], Epinions [63] and Flixster [64]. Due to the limited space in our local machine, 10,000 users are selected in our Flixster dataset. The overall statistics of the three datasets are presented in Table 3. From Table 3, we could observe that all the datasets are very large-scale and extremely sparse, thence they are very suitable for testing different models.

Besides, the distributions of out-degree, in-degree and rating number on the Epinions dataset are illustrated in Fig. 4. As observed, a power law distribution, which is ubiquitous in online applications, is obeyed on the Epinions dataset, where most users have few trustors, trustees, or items rated, while a few users have a large number of them. Although, as Fig. 4 conveys, rating information may be inadequate for high quality personalized recommendation, social information could alternatively serve as another additional information source even though it is also highly sparse.

5.1.2 Evaluating Methodology

We conduct a five-fold cross validation to test the effectiveness and robustness of recommendation models. Besides, we adopt two perspectives to test models' capabilities of rating prediction [22], [65], one is the *all users* view, where every user presented in the testing fold would be involved in calculating metrics, while the other is the *cold-start users* view, where only those tested users who have rated less than or equal to five items would be involved in calculating metrics.



(a) Out-degree (b) In-degree (c) Rating number

Fig. 4. Statistics of the Epinions dataset.

5.1.3 Evaluating Metrics

We evaluate on two recommendation tasks of *rating prediction* and *item ranking*, using various evaluation metrics as follows.

Rating Prediction. For evaluating the predictive quality of the recommendation algorithms, two widely used metrics are adopted: Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) [66]. Smaller values of MAE and RMSE indicate better predictive accuracy. Specifically, denote the testing set as $\mathcal{T}$, the MAE and RMSE are defined as $\mathrm{MAE}(\mathcal{T}) = (1/|\mathcal{T}|)\sum_{(u_i,v_j)\in\mathcal{T}}|r_{ij} - \hat{r}_{ij}|$, and $\mathrm{RMSE}(\mathcal{T}) = \sqrt{(1/|\mathcal{T}|)\sum_{(u_i,v_j)\in\mathcal{T}}(r_{ij} - \hat{r}_{ij})^2}$.

Item Ranking. For evaluating the top-N recommendation quality of the recommendation algorithms, we use two widely adopted ranking based metrics: Normalized Discounted Cumulative Gain (NDCG) and Mean Reciprocal Rank (MRR). In these two metrics, the personalized top-N recommendation list is composed of each user's top-N estimated highest-rated items in the testing set, the positive items are those items presented in the testing set with ratings higher than a pre-defined threshold ϵ, and the negative items are the remaining in the testing set. Specifically, NDCG and MRR are defined as $\mathrm{NDCG@}N = (1/|\mathcal{U}_\mathcal{T}|)\sum_{u\in\mathcal{U}_\mathcal{T}}(\sum_{j=1}^{N}\frac{2^{\mathrm{rel}(j,u,L_u^N)}-1}{\log_2(j+1)})/(\sum_{j=1}^{N}\frac{2^{\mathrm{rel}(j,u,L_u^N)}-1}{\log_2(j+1)})$ and $\mathrm{MRR@}N = (1/|\mathcal{U}_\mathcal{T}|)\sum_{u\in\mathcal{U}_\mathcal{T}}(1/\mathrm{Rank}_u)$, where $\mathcal{U}_\mathcal{T}$ represents those users appearing in the testing set $\mathcal{T}$, L_u^N and L_u^N stand for the real and ideal top-N recommendation list for u respectively, $\mathrm{rel}(\cdot)\in\{0,1\}$ is a function indicating whether the item at rank j in u's top-N recommendation list is a positive item, and Rank_u is the rank of the first positive item in u's top-N recommendation list. In our experiments, we set $\epsilon = 3.5$ for all the datasets.

Besides, we have also adopted another well-known ranking based metric: receiver operating characteristic (ROC) curve [67], which is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various N settings. The ROC curve could provide a comprehensive description of ranking performance.

5.2 Comparative Approaches

To evaluate the performance comprehensively, we compare our SCSVD model[2] with three groups of methods, including traditional recommender systems [32], [33], [68], traditional social recommender systems [6], [7], [9], [22], [31], [35], and deep neural network based recommender systems [27], [28], [52], [69], [70], [71]. We summarize these methods in detail as below.

CBSVD [32]. CBSVD first clusters users and items according to their latent preferences, and then uses these learned user and item clusters to augment SVD++.

SVD++ [33]. SVD++ is a basic but state-of-the-art recommendation model which uses both user-specific and item-based user embeddings to represent a user's preferences.

PMF [68]. Probabilistic matrix factorization utilizes user-item rating matrix only and models latent factors of users and items by Gaussian distributions.

2. The source code of SCSVD is publicly available at https://github.com/Kwan1997/SCSVD.

SoRec [6]. SoRec simultaneously factorizes the user-item rating matrix and the user-user trust matrix.

SoReg [7]. SoReg models user trust relationships as regularization terms to constrain the rating matrix factorization.

SocialMF [9]. SocialMF considers the trust information and its propagating property in rating matrix factorization.

TrustSVD [22]. TrustSVD incorporates the trust influence from social neighbors on top of SVD++. It shows state-of-the-art performance on social recommendation results.

TrustMF [31]. TrustMF maps users into two parts of truster space and trustee space, and factorizes the rating matrix and trust network simultaneously for rating prediction.

SSLSVD [35]. SSLSVD adopts a transductive support vector machine to cluster social relations, and then uses the clustering results to augment TrustSVD.

GraphRec [27]. GraphRec employs an attention mechanism based deep neural network to learn representations of each user from both rating information and social relations, to obtain reliable rating predictions.

DANSER [28]. DANSER adopts dual graph attention networks to collaboratively learn representations for two-fold social effects, where one is modeled by a user-specific attention weight and the other is modeled by a dynamic and context-aware attention weight.

NeuMF [52]. NeuMF is a state-of-the-art matrix factorization model with neural network architecture. The original implementation is for the recommendation ranking task, and we have modified its loss function to the mean squared error to apply it to the rating prediction task.

NGCF [69]. NGCF is an item ranking model, which adopts a graph convolutional network to propagate user embeddings and item embeddings on the user-item bipartite graph to model higher-order user-item interactions.

LightGCN [70]. LightGCN is an item ranking model, which adopts a simple and neat graph convolutional network to model user-item interactions.

MHCN [71]. MHCN is an item ranking model, which uses a well-designed multichannel hypergraph convolutional network to incorporate higher-order user connections.

We have fine-tuned all comparative models carefully. Table 4 lists all parameters used in the experiments. Note that there is no tunable parameters except the embedding size k in the GraphRec model. Besides, for SCSVD, we set $\eta = 10^6$ and the learning rate $\theta = 0.01$ for all the experiments. We have also adopted a learning rate decay strategy, where we set $\eta \leftarrow 0.8\eta$ per 10 iterations, to match the speed of gradient descent with other subproblems. Note that we only evaluate three deep learning methods of NGCF, LightGCN and MHCN in the item recommendation task, as they are item ranking models. Besides, as NGCF, LightGCN and MHCN can only handle implicit feedback data, we binarize all the datasets for them by reassigning all ratings by a threshold of 3.5.

5.3 Model Comparisons

5.3.1 Rating Prediction Task

The experimental results of the rating prediction task are presented in Tables 5 and 6, where the best results are in

TABLE 4
Parameter Settings of All Methods

Dataset	TrustSVD		SocialMF		SoRec		SoReg		PMF	SVD++	TrustMF		NeuMF	GraphRec	DANSER			NGCF			LightGCN			MHCN			CBSVD			SSLSVD			SCSVD			
	λ	λ_t	λ	λ_t	λ	λ_c	λ	β	λ	λ	λ	λ_t	λ	\	τ	λ	C_t	λ	b	s	λ	b	s	λ	b	s	λ	α	k	λ	λ_t	α	λ	α	β	c
Ciao	0.6	1.5	0.001	1.0	0.001	0.01	0.001	0.1	0.33	0.1	0.001	1.0	0.0000001	\	0.5	0.001	10	0.001	0.2	0.2	0.001	0.2	0.2	0.01	0.2	0.2	0.15	0.15	50	0.6	1.5	0.3	0.15	40	200	300
Epinions	0.9	0.5	0.001	1.0	0.001	0.01	0.001	0.1	0.2	0.2	0.001	1.0	0.01	\	0.5	0.001	10	0.001	0.2	0.2	0.001	0.2	0.2	0.01	0.2	0.2	0.2	0.15	50	0.9	0.5	0.3	0.09	40	200	200
Flixster	0.8	0.5	0.001	1.0	0.001	0.01	0.001	0.1	0.05	0.04	0.001	1.0	0.00001	\	0.5	0.00005	10	0.001	0.2	0.2	0.001	0.2	0.2	0.01	0.2	0.2	0.04	0.15	50	0.8	0.5	0.3	0.06	25	100	400

TABLE 5
Rating Prediction Performance Comparisons of Different Models on *All Users*

Dataset		Metric	CBSVD	SSLSVD	TrustSVD	SocialMF	SoRec	SoReg	PMF	SVD++	TrustMF	NeuMF	GraphRec	DANSER	SCSVD
Ciao	$k=5$	MAE	0.7214	0.7245	0.7253	0.7483	0.7649	0.7679	0.8861	0.7220	0.7470	0.7258	0.8171	0.7368	**0.7204**
		RMSE	0.9541	0.9570	0.9578	0.9795	0.9887	1.0382	1.1061	0.9581	0.9800	0.9716	1.0153	0.9659	**0.9530**
	$k=10$	MAE	0.7215	0.7252	0.7256	0.7553	0.7663	0.7659	0.8976	0.7229	0.7573	0.7251	0.7765	0.7463	**0.7209**
		RMSE	0.9543	0.9585	0.9589	0.9874	1.0124	1.0386	1.1140	0.9600	1.0125	0.9681	0.9893	0.9733	**0.9539**
Epinions	$k=5$	MAE	0.8023	0.8016	0.8016	0.8288	0.8384	0.8807	0.9120	0.8028	0.8200	0.7989	0.9337	0.8472	**0.7973**
		RMSE	1.0412	1.0455	1.0448	1.0810	1.0738	1.1564	1.1805	1.0487	1.0716	1.0480	1.1293	1.0846	**1.0402**
	$k=10$	MAE	0.8023	0.8023	0.8025	0.8375	0.8374	0.8480	0.9431	0.8052	0.8294	**0.7951**	0.8534	0.8472	0.7974
		RMSE	1.0412	1.0466	1.0460	1.0961	1.1022	1.1183	1.2107	1.0516	1.1053	1.0493	1.0798	1.0847	**1.0408**
Flixster	$k=5$	MAE	0.6662	0.6649	0.6642	0.6853	0.6961	0.7115	0.6995	0.6670	0.6869	0.6701	0.7933	0.6899	**0.6639**
		RMSE	0.8899	0.8864	0.8854	0.9081	0.9253	0.9446	0.9338	0.8926	0.9073	0.8914	0.9952	0.9185	**0.8839**
	$k=10$	MAE	0.6641	0.6638	0.6632	0.6876	0.7037	0.6977	0.7013	0.6659	0.6893	0.6655	0.7349	0.6930	**0.6615**
		RMSE	0.8878	0.8865	0.8853	0.9139	0.9390	0.9290	0.9368	0.8929	0.9134	0.8879	0.9492	0.9217	**0.8815**

TABLE 6
Rating Prediction Performance Comparisons of Different Models on *Cold-Start Users*

Dataset		Metric	CBSVD	SSLSVD	TrustSVD	SocialMF	SoRec	SoReg	PMF	SVD++	TrustMF	NeuMF	GraphRec	DANSER	SCSVD
Ciao	$k=5$	MAE	**0.7281**	0.7337	0.7344	0.7295	0.7691	0.7477	1.0643	0.7852	0.9928	0.7409	0.8192	0.7365	0.7290
		RMSE	0.9745	0.9670	**0.9667**	0.9727	0.9800	0.9781	1.2836	0.9985	1.2773	0.9888	1.0149	0.9712	0.9761
	$k=10$	MAE	**0.7290**	0.7349	0.7347	0.7319	0.7364	0.7482	0.8804	0.7388	1.0098	0.7381	0.7762	0.7408	0.7299
		RMSE	0.9749	0.9662	**0.9655**	0.9735	1.0258	0.9856	1.1240	1.0318	1.2813	0.9876	0.9941	0.9696	0.9763
Epinions	$k=5$	MAE	0.8493	0.8510	0.8498	0.8560	0.8699	0.8723	1.3046	0.8805	1.1904	**0.8376**	0.9752	0.8655	0.8463
		RMSE	1.0953	1.0935	1.0916	1.1260	1.0996	1.1273	1.5453	1.1137	1.5162	1.0970	1.1715	1.0981	**1.0888**
	$k=10$	MAE	0.8493	0.8505	0.8518	0.8554	0.8344	0.8522	1.0519	**0.8324**	1.2434	0.8326	0.9002	0.8655	0.8458
		RMSE	1.0955	1.0924	1.0928	1.1246	1.1341	1.1341	1.2964	1.1464	1.5660	1.0973	1.1522	1.0981	**1.0886**
Flixster	$k=5$	MAE	0.7999	0.8352	0.8335	0.8061	0.9271	0.9012	1.0916	0.8521	0.8906	**0.7908**	0.9081	0.9074	0.7988
		RMSE	1.0191	1.0480	1.0451	1.0242	1.1623	1.1453	1.3206	1.1134	1.2091	**1.0159**	1.1210	1.1272	1.0176
	$k=10$	MAE	0.8003	0.8292	0.8319	0.8094	0.9288	0.9105	0.9260	0.8709	0.9084	**0.7838**	0.8384	0.9286	0.7906
		RMSE	1.0195	1.0442	1.0447	1.0276	1.1731	1.1716	1.1399	1.1434	1.2016	1.0111	1.0803	1.1523	**1.0099**

boldface, while the top-2 results are in the shade. We have the following observations:

- In the view of *all users* (Table 5), our proposed SCSVD outperforms all other comparative approaches in terms of both MAE and RMSE in all cases except one (i.e., NeuMF performs best on the Epinions dataset when $k=10$ in terms of RMSE), reflecting the strong effectiveness of SCSVD. Besides, it is noted that, SVD++, which only relies on rating information, could beat most matrix factorization based social recommendation models. This implies that the SVD++ is highly underestimated, and this point is consistent with recent findings [22], [57].
- In the view of *cold-start users* (Table 6), the prediction accuracy of all recommendation models are worse than the counterparts in the view of *all users*. Besides, our proposed SCSVD achieves the top-2 performance in 75% cases, which is the highest among all models. This has illustrated the robustness of SCSVD since it is stable even in a cold-start scenario.
- With the embedding size k increases, the rating prediction performance is not necessarily improved. E.g., the performance of PMF consistently deteriorates given a larger k. This is mainly because that, although a longer vector has a greater ability to encode more information, it would also be more vulnerable to the over-fitting problem.

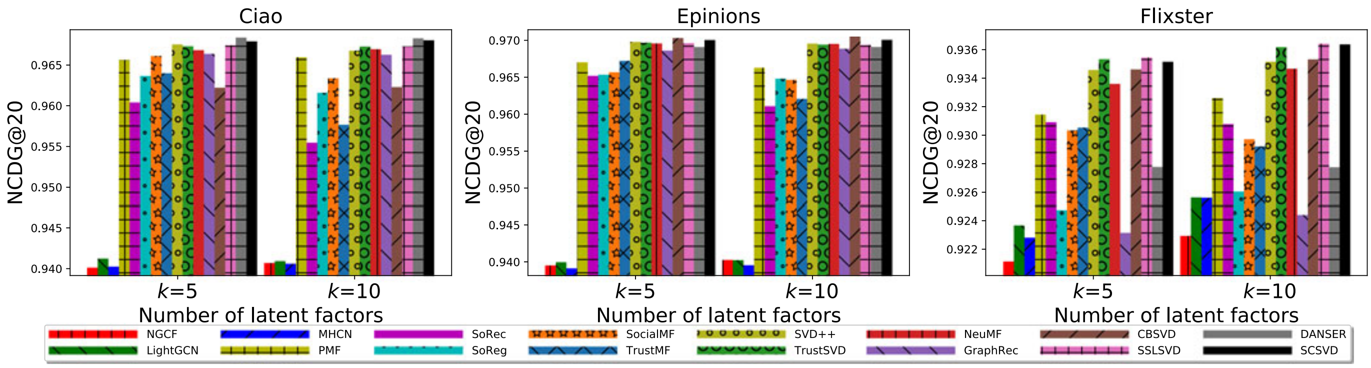


Fig. 5. NDCG@20 performance with different k on the three datasets.

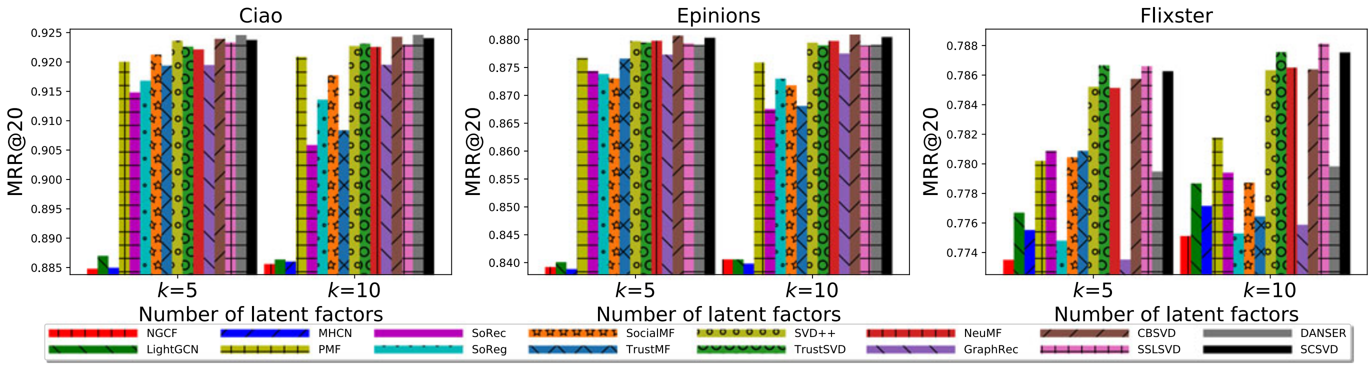


Fig. 6. MRR@20 performance with different k on the three datasets.

- Deep learning based recommender systems are not always better than traditional matrix factorization based recommender systems. E.g., GraphRec performs nearly the worst in the view of *all users*. Part of the reason is that GraphRec, as the authors claimed in their paper, is designed for completely warm users and items, thus it fails in a regular recommendation scene.

5.3.2 Item Recommendation Task

The experimental results of the item recommendation task are presented in Figs. 5, 6 and 7 (bars and curves in black represent our SCSVD model). We have fixed the number of recommended items $N = 20$ in Figs. 5 and 6, and we have fixed the dimensionality of latent space $k = 10$ in Fig. 7. We have the following observations:

- In terms of $\mathrm{NDCG@20}$ and $\mathrm{MRR@20}$ in Figs. 5 and 6 respectively, our proposed SCSVD attains top-3 performance on all the datasets. This has illustrated not only effectiveness but also flexibility of SCSVD, since it could switch freely between the rating prediction task and the item ranking task, and achieve considerable performance.
- Similar to the previous task, a larger embedding size k may not bring a better $\mathrm{NDCG@20}$ or $\mathrm{MRR@20}$.
- The experimental performance of GraphRec, DANSER and SoReg are not very consistent across different datasets. This is mainly because different models have their own preferred datasets with different statistics. For these three methods, when the rating data size (i.e., # users and # items) and the degree of social engagement (i.e., the proportion of trustors and trustees) are at a low level, their performance would deteriorate. On the contrary, the performance of SCSVD is relatively stable across different datasets, although it is not always the best. Another exciting observation is that SCSVD performs better when rating information is sparser, and it is not sensitive to the density of social information.
- In terms of the ROC curve in Fig. 7, we can observe that our proposed SCSVD lies at the top of all models on the Flixster dataset. On the Ciao and Epinions datasets, our SCSVD consistently achieves top-2 performance. This shows that our proposed SCSVD is eminent not just in static scenes, but also in dynamic settings.
- Three deep learning models NGCF, LightGCN and MHCN have the worst performance on all three datasets, in terms of $\mathrm{NDCG@20}$ and $\mathrm{MRR@20}$. This is because they can only deal with implicit datasets,

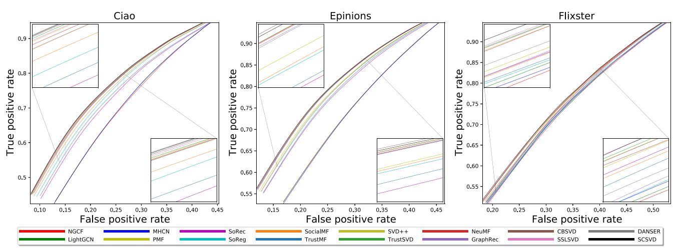


Fig. 7. ROC curves with $k = 10$ on the three datasets.

TABLE 7
Ablation Evaluation of SCSVD and SCSVDp

Dataset	Ciao				Epinions				Flixster			
Metric	MAE		RMSE		MAE		RMSE		MAE		RMSE	
# Factors	$k=5$	$k=10$	$k=5$	$k=10$	$k=5$	$k=10$	$k=5$	$k=10$	$k=5$	$k=10$	$k=5$	$k=10$
SCSVD	**0.7204**	**0.7209**	**0.9530**	**0.9539**	**0.7973**	**0.7974**	**1.0402**	**1.0408**	**0.6639**	**0.6615**	**0.8839**	**0.8815**
SCSVDp	0.7214	0.7222	0.9549	0.9561	0.7983	0.8002	1.0429	1.0454	0.6643	0.6623	0.8853	0.8836

thus the concrete rating information cannot be exploited to guide the ranking process. Our proposed recommendation model handling explicit ratings clearly wins these three competitors.

5.4 Ablation Study

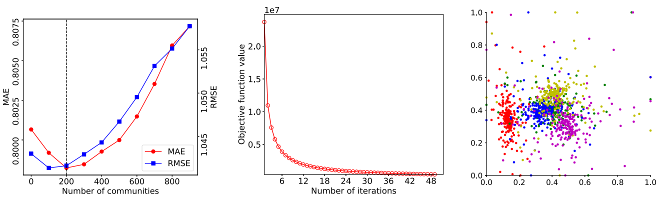
We conduct an ablation study to show the effectiveness of community detection in our SCSVD recommendation model. Specifically, we implement a variant of SCSVD without using community detection, denoted as SCSVDp, which removes modularity maximization and its corresponding linear mapping from the objective function of SCSVD (i.e., let $\alpha = \beta = 0$). We compare SCSVD with its pruned version, SCSVDp. Table 7 shows the MAE and RMSE results of SCSVD and SCSVDp on all datasets, in terms of *all users* with k varied in $\{5, 10\}$. As observed, SCSVD consistently outperforms SCSVDp, illustrating the usefulness of community detection in our SCSVD framework.

5.5 Community Sensitivity Evaluation

We evaluate the effect of changing the number of communities c on the Epinions dataset. In this experiment, c is investigated in the range from 0 to 1000 with a step size of 100, where $c = 0$ refers to the SVD++ model. The experimental results are illustrated in Fig. 8a, where the vertical dashed lines represent the c values used in previous comparative experiments. As Fig. 8a depicts, the community structure has indeed enhanced the SVD++ model (corresponding to $c = 0$). Besides, Fig. 8a also indicates that a very large or small number of communities in our scheme is not effective in improving prediction performance.

5.6 Convergence Rate Evaluation

We have already proven the convergence of Algorithm 1 in Section 4.4, and now we empirically study its convergence property. Specifically, we run SCSVD on the Epinions dataset with all parameters fixed as in Table 4, and the experimental results are shown in Fig. 8b. As observed, the objective function value monotonically decreases and converges rapidly within dozens of iterations. This phenomenon has empirically validated the correctness of our convergence theory.



(a) Sensitivity of c (b) Convergence curve (c) Visualization of $\mathbf{U}$

Fig. 8. Community sensitivity analysis, converge analysis and embedding visualization on the Epinions dataset.

5.7 User Embedding Visualization

As discussed in Section 4.6, users belonging to the same community would be modeled as having similar latent preferences. Therefore, user embeddings might manifest a clustering structure eventually. To validate this hypothesis, we conduct the following experiment on the Epinions dataset. We first feed the learned user embedding matrix $\mathbf{U}$ into the standard PCA tool [72] to reduce the dimensionality of user embeddings from k to 2. Then, we normalize user embeddings to turn their values into a common scale. Finally, we visualize all user embeddings belonging to the top-5 biggest communities in $\mathbb{R}^2$. The visualization results are shown in Fig. 8c, where nodes belonging to the same community share the same color. As observed, low-dimensional user embeddings show a relatively clear clustering structure, which has validated our hypothesis.

5.8 Efficiency and Scalability Evaluations

We evaluate the efficiency and scalability of SCSVD, compared with four methods SVD++ [33], DANSER [28], GraphRec [27] and MHCN [71]. For scalability test, we first generate five synthetic recommendation datasets with different sizes. The statistics of the five synthetic datasets are as follows: # users varies in the increased order whose value is selected from $\{100, 500, 1000, 5000, 10000\}$; # item, # ratings, and # relations are all set as five times of # users; all ratings and relations are randomly generated. We denote these five synthetic datasets as Syn-1, ..., Syn-5, respectively. Table 8 reports the model training time of all comparison methods on the five synthetic datasets and three real datasets Ciao, Epinions and Flixster. As observed, SCSVD achieves a scalable efficiency on five synthetic datasets with

TABLE 8
Running Time Comparison (in Second/Epoch)

Dataset		SVD++	SCSVD	DANSER	GraphRec	MHCN
Name	# Users					
Syn-1	100	0.04	0.04	2.93	1.01	0.15
Syn-2	500	0.14	0.33	3.26	6.77	2.34
Syn-3	1,000	0.29	0.51	4.26	10.13	12.27
Syn-4	5,000	1.74	3.04	12.49	104.27	78.12
Syn-5	10,000	3.06	7.71	18.82	242.11	160.35
Ciao	7,375	183.95	229.28	1267.38	4885.23	824.35
Flixster	10,000	1091.74	1297.74	5986.37	13411.63	2019.32
Epinions	40,163	196.67	279.82	2353.07	14098.38	3546.42

the increased number of users. Besides, SCSVD takes a slightly more time than the basic model SVD++, due to the fact that we incorporate an additional community detection module in our method. Moreover, our method SCSVD runs much faster than three deep learning methods DANSER, GraphRec and MHCN on five synthetic datasets and three real datasets, validating its efficiency superiority.

6 Conclusion and Future Work

In this paper, we proposed a novel community detection based recommendation framework, SCSVD, to improve recommender systems. To optimize SCSVD, an efficient optimization algorithm was also derived, and its convergence and computational complexity were also theoretically analyzed. Furthermore, we provided theoretical analysis towards the functionary mechanism of SCSVD. Comprehensive experimental results demonstrated that our proposed SCSVD outperformed both traditional matrix factorization based (social) recommendation models and advanced neural network based (social) recommendation models, in terms of both predictive accuracy and top-N ranking precision. This work also opens several interesting directions, including 1) developing faster and interpretable methods by incorporating heuristic graph-based community detection into social recommendation, and also 2) designing novel overlapping community detection methods to capture multiple community memberships.

References

[1] F. Ricci, L. Rokach, and B. Shapira, *Introduction to Recommender Systems Handbook*. Berlin, Germany: Springer, 2011.

[2] N. Ben-Lhachemi and E. H. Nfaoui, "Using tweets embeddings for hashtag recommendation in Twitter," *Procedia Comput. Sci.*, vol. 127, pp. 7–15, 2018.

[3] J. Tang, X. Hu, and H. Liu, "Social recommendation: A review," *Social Netw. Anal. Mining*, vol. 3, no. 4, pp. 1113–1133, 2013.

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

[5] P. V. Marsden and N. E. Friedkin, "Network studies of social influence," *Sociol. Methods Res.*, vol. 22, no. 1, pp. 127–151, 1993.

[6] H. Ma, H. Yang, M. R. Lyu, and I. King, "SoRec: Social recommendation using probabilistic matrix factorization," in *Proc. 17th ACM Conf. Inf. Knowl. Manage.*, 2008, pp. 931–940.

[7] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proc. 4th ACM Int. Conf. Web Search Data Mining*, 2011, pp. 287–296.

[8] M. Jamali and M. Ester, "TrustWalker: A random walk model for combining trust-based and item-based recommendation," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 397–406.

[9] 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.

[10] J. Tang, X. Hu, H. Gao, and H. Liu, "Exploiting local and global social context for recommendation," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2013, pp. 2712–2718.

[11] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, 2002.

[12] X. Huang, L. V. Lakshmanan, and J. Xu, *Community Search Over Big Graphs*. San Rafael, CA, USA: Morgan Claypool Pub., 2019.

[13] D. Jin *et al.*, "Incorporating network embedding into Markov random field for better community detection," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 160–167.

[14] S. Fortunato, "Community detection in graphs," *Phys. Rep.*, vol. 486, no. 3–5, pp. 75–174, 2010.

[15] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 203–209.

[16] J. Tang, H. Gao, X. Hu, and H. Liu, "Exploiting homophily effect for trust prediction," in *Proc. 6th ACM Int. Conf. Web Search Data Mining*, 2013, pp. 53–62.

[17] J. Tang *et al.*, "Recommendation with social dimensions," in *Proc. 30th AAAI Conf. Artif. Intell.*, 2016, pp. 251–257.

[18] C. Cao, Q. Ni, and Y. Zhai, "An improved collaborative filtering recommendation algorithm based on community detection in social networks," in *Proc. 17th Annu. Conf. Genet. Evol.*, 2015, pp. 1–8.

[19] Z. Li, J. Tang, and X. He, "Robust structured nonnegative matrix factorization for image representation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1947–1960, May 2018.

[20] Z. Li, J. Tang, and T. Mei, "Deep collaborative embedding for social image understanding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2070–2083, Sep. 2019.

[21] Z. Li, J. Tang, L. Zhang, and J. Yang, "Weakly-supervised semantic guided hashing for social image retrieval," *Int. J. Comput. Vis.*, vol. 128, pp. 2265–2278, 2020.

[22] G. Guo, J. Zhang, and N. Yorke-Smith, "TrustSVD: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 123–129.

[23] 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.

[24] R. Zafarani, M. A. Abbasi, and H. Liu, *Social Media Mining: An Introduction*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[25] 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.

[26] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, and J. Tang, "DeepInf: Modeling influence locality in large social networks," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 2110–2119.

[27] W. Fan *et al.*, "Graph neural networks for social recommendation," in *Proc. 28th Int. Conf. World Wide Web*, 2019, pp. 417–426.

[28] Q. Wu *et al.*, "Dual graph attention networks for deep latent representation of multifaceted social effects in recommender systems," in *Proc. 28th Int. Conf. World Wide Web*, 2019, pp. 2091–2102.

[29] W. Fan, T. Derr, Y. Ma, J. Wang, J. Tang, and Q. Li, "Deep adversarial social recommendation," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, 2019, pp. 1351–1357.

[30] H. Ma, I. King, and M. R. Lyu, "Learning to recommend with social trust ensemble," in *Proc. 32nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2009, pp. 203–210.

[31] B. Liu and J. Liu, "Social collaborative filtering by trust," in *Proc. 23rd Int. Joint Conf. Artif. Intell.*, 2013, pp. 52–56.

[32] N. Mirbakhsh and C. X. Ling, "Clustering-based factorized collaborative filtering," in *Proc. 7th ACM Conf. Recommender Syst.*, 2013, pp. 315–318.

[33] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 426–434.

[34] J.-D. Zhang and C.-Y. Chow, "GeoSoCa: Exploiting geographical, social and categorical correlations for point-of-interest recommendations," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2015, pp. 443–452.

[35] Z. Hu *et al.*, "SSL-SVD: Semi-supervised learning–based sparse trust recommendation," *ACM Trans. Internet Technol.*, vol. 20, no. 1, pp. 1–20, 2020.

[36] C. Yang, L. Bai, C. Zhang, Q. Yuan, and J. Han, "Bridging collaborative filtering and semi-supervised learning: A neural approach for POI recommendation," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2017, pp. 1245–1254.

[37] C. Li, K. Jia, D. Shen, C.-J. R. Shi, and H. Yang, "Hierarchical representation learning for bipartite graphs," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, 2019, pp. 2873–2879.

[38] Z. Li *et al.*, "Hierarchical bipartite graph neural networks: Towards large-scale e-commerce applications," in *Proc. 36th IEEE Int. Conf. Data Eng.*, 2020, pp. 1677–1688.

[39] I. Maksimov, R. Rivera-Castro, and E. Burnaev, "Addressing cold start in recommender systems with hierarchical graph neural networks," in *Proc. IEEE Int. Conf. Big Data*, 2020, pp. 5128–5137.

[40] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proc. 19th Int. Conf. World Wide Web*, 2010, pp. 631–640.
[41] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: The state-of-the-art and comparative study," *ACM Comput. Surv.*, vol. 45, no. 4, pp. 1–35, 2013.
[42] I. Psorakis, S. Roberts, M. Ebden, and B. Sheldon, "Overlapping community detection using Bayesian non-negative matrix factorization," *Phys. Rev. E*, vol. 83, no. 6, 2011, Art. no. 066114.
[43] H. Zhang, I. King, and M. R. Lyu, "Incorporating implicit link preference into overlapping community detection," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 396–402.
[44] H. Zhang, T. Zhao, I. King, and M. R. Lyu, "Modeling the homophily effect between links and communities for overlapping community detection," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2016, pp. 3938–3944.
[45] Y. Zhang and D.-Y. Yeung, "Overlapping community detection via bounded nonnegative matrix tri-factorization," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2012, pp. 606–614.
[46] J. Yang and J. Leskovec, "Overlapping community detection at scale: A nonnegative matrix factorization approach," in *Proc. 6th ACM Int. Conf. Web Search Data Mining*, 2013, pp. 587–596.
[47] B.-J. Sun, H. Shen, J. Gao, W. Ouyang, and X. Cheng, "A non-negative symmetric encoder-decoder approach for community detection," in *Proc. 26th ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 597–606.
[48] F. Ye, C. Chen, and Z. Zheng, "Deep autoencoder-like nonnegative matrix factorization for community detection," in *Proc. 27th ACM Conf. Inf. Knowl. Manage.*, 2018, pp. 1393–1402.
[49] E. A. Leicht and M. E. J. Newman, "Community structure in directed networks," *Phys. Rev. Lett.*, vol. 100, no. 11, 2008, Art. no. 118703.
[50] U. Brandes *et al.*, "Maximizing modularity is hard," 2006, *arXiv: physics/0608255v2*.
[51] A. C. Elliott and W. A. Woodward, *Statistical Analysis Quick Reference Guidebook: With SPSS Examples*. Thousand Oaks, CA, USA: Sage, 2007.
[52] 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.
[53] Z. Chen, J. Bruna, and L. Li, "Supervised community detection with line graph neural networks," in *Proc. 7th Int. Conf. Learn. Representations*, 2019. [Online]. Available: https://iclr.cc/Conferences/2019/Schedule?type=Poster and https://nyuscholars.nyu.edu/en/publications/supervised-community-detection-with-line-graph-neural-networks
[54] Y. Koren, "Collaborative filtering with temporal dynamics," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 447–456.
[55] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, "Large-scale parallel collaborative filtering for the Netflix prize," in *Proc. 4th Int. Conf. Algorithmic Aspects Inf. Manage.*, 2008, pp. 337–348.
[56] B. Yang, Y. Lei, J. Liu, and W. Li, "Social collaborative filtering by trust," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1633–1647, Aug. 2017.
[57] S. Rendle, W. Krichene, L. Zhang, and J. Anderson, "Neural collaborative filtering vs. matrix factorization revisited," in *Proc. 14th ACM Conf. Recommender Syst.*, 2020, pp. 240–248.
[58] T. Viklands, "Algorithms for the weighted orthogonal procrustes problem and other least squares problems," Ph.D. dissertation, Dept. Comput. Sci., Umeã Univ., Umeã, Sweden, 2006.
[59] C. Ding, X. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proc. 5th SIAM Int. Conf. Data Mining*, 2005, pp. 606–610.
[60] M. Berger, *Geometry I*. Berlin, Germany: Springer, 2009.
[61] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2012.
[62] J. Tang, H. Gao, and H. Liu, "mTrust: Discerning multi-faceted trust in a connected world," in *Proc. 5th ACM Int. Conf. Web Search Data Mining*, 2012, pp. 93–102.
[63] P. Massa and P. Avesani, "Trust-aware recommender systems," in *Proc. 1st ACM Conf. Recommender Syst.*, 2007, pp. 17–24.
[64] R. Zafarani and H. Liu, "Social computing data repository at ASU," *School Compu., Inform. Decis. Syst. Eng.*, Arizona State University, 2009. [Online]. Available: http://socialcomputing.asu.edu.
[65] G. Guo, J. Zhang, and N. Yorke-Smith, "Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems," *Knowl.-Based Syst.*, vol. 74, pp. 14–27, 2015.
[66] B. Chen, S. Yu, J. Tang, M. He, and Y. Zeng, "Using function approximation for personalized point-of-interest recommendation," *Expert Syst. Appl.*, vol. 79, pp. 225–235, 2017.
[67] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, 2006.
[68] A. Mnih and R. R. Salakhutdinov, "Probabilistic matrix factorization," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1257–1264.
[69] 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.
[70] 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.
[71] 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. 30th Int. Conf. World Wide Web*, 2021, pp. 413–424.
[72] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics Intell. Lab. Syste.*, vol. 2, no. 1–3, pp. 37–52, 1987.

Jiewen Guan received the BEng degree from the Zhejiang University of Technology in 2019. He is currently working toward the MEng degree with Xiamen University. He has authored or coauthored several papers in the *IEEE Transactions on Neural Networks and Learning Systems.* His research interests include data mining and optimization algorithms.

Xin Huang received the PhD degree from The Chinese University of Hong Kong in 2014. He is currently an assistant professor with Hong Kong Baptist University. His research interests include graph data management and mining.

Bilian Chen received the PhD degree from The Chinese University of Hong Kong in 2012. She is currently an associate professor with Xiamen University. She has authored or coauthored several papers in the *IEEE Transactions on Neural Networks and Learning Systems*, the *SIAM Journal on Optimization*, the *Journal of Global Optimization*, and the *Information Sciences*. Her research interests include machine learning, optimization theory, and recommendation system.