A Social Collaborative Filtering Method to Alleviate Data Sparsity Based on Graph Convolutional Networks

Haitao XIE†, Member, Qingtao FAN††, and Qian XIAO††a, Nonmembers

SUMMARY Nowadays recommender systems (RS) keep drawing attention from academia, and collaborative filtering (CF) is the most successful technique for building RS. To overcome the inherent limitation, which is referred to as data sparsity in CF, various solutions are proposed to incorporate additional social information into recommendation processes, such as trust networks. However, existing methods suffer from multi-source data integration (i.e., fusion of social information and ratings), which is the basis for similarity calculation of user preferences. To this end, we propose a social collaborative filtering method based on novel trust metrics. Firstly, we use Graph Convolutional Networks (GCNs) to learn the associations between social information and user ratings while considering the underlying social network structures. Secondly, we measure the direct-trust values between neighbors by representing multi-source data as user ratings on popular items, and then calculate the indirect-trust values based on trust propagations. Thirdly, we employ all trust values to create a social regularization in user-item rating matrix factorization in order to avoid overfitting. The experiments on real datasets show that our approach outperforms the other state-of-the-art methods on usage of multi-source data to alleviate data sparsity.

key words: collaborative filtering, data sparsity, trust metrics, graph convolutional network

1. Introduction

Recommender systems (RS) are widely applied in electronic commerce and social media websites to provide personalized information according to user preferences. Collaborative filtering (CF) is the most important and popular technique in RS. While content-based recommenders require feature descriptions of items, CF just relies on user-item rating matrix [1]. The underlying assumption of CF is that if users give similar ratings to the same items, they are likely to have similar preferences. Consequently, RS suffer from the problem of data sparsity, also called data scarcity, regardless of their various implementation approaches [2]. E.g., the density of Netflix data for movie recommendations is only about 1.2% [3], which means two random users usually have not rated any items in common. The lack of explicit and implicit rating data leads to the inaccuracy of recommendations, especially for CF based on matrix factorizations [4].

To overcome the inherent weakness of CF, diverse social information is incorporated into recommendation processes [5], [6], such as trust networks and user profiles consisting of age, gender, career, education, etc. These approaches are also referred to as social recommendations [7]. Since trust relationships reflect the preference similarities between users in social networks, trust networks are naturally regarded as the most important supplements to user ratings in CF [8]. E.g., Massa and Avesani proposed a notable trust metric (i.e., MoleTrust) [9] for building trust-aware RS based on trust scores given by users. MoleTrust is a depth-first graph walking algorithm with a tunable trust propagation horizon. Later, they integrated PageRank into MoleTrust to improve the accuracy of trust metrics [10].

Unfortunately, in the real world, most social media websites do not provide users any way to explicitly express their trusts in each other, and therefore researchers have to design trust inference mechanisms based on the information containing trust relationships implicitly [11]. To the best of our knowledge, although trust-aware RS have been widely studied in recent years, there are still two problems to be resolved efficiently: (1) how to derive trust values from various social information; (2) how to integrate the data of trusts with user ratings [12]. To address the above two issues, in this study, we propose a social collaborative filtering method, called RecGCN, which aims to integrate multi-source user data to alleviate data sparsity. Firstly, RecGCN uses GCNs to synthetically learn the associations between social information and user ratings, and then derive direct-trust values between neighbors from integrated data. Secondly, RecGCN calculates indirect-trust values based on trust propagations, while considering the diversity of recommendations. Thirdly, all trust values are employed in social regularizations as constraints of matrix factorizations to alleviate data sparsity.

Our main contributions are: (1) we propose a general framework to integrate multi-source user data by converting social information to user ratings on popular items; (2) we alleviate the data sparsity problem of CF by incorporating novel trust metrics in recommendation processes.

The rest of the paper is organized as: we start by introducing related works in Sect. 2. In Sect. 3, we present related concepts and preliminaries. We demonstrate the novel trust metrics in Sect. 4. In Sect. 5, we present the RecGCN method. Finally, we conduct experiments to evaluate our approach and compare it with other state-of-the-art approaches using the real-world dataset.

Manuscript received October 18, 2019.
Manuscript revised June 13, 2020.
Manuscript publicized August 28, 2020.
†The author is with Beijing United Information Center of Science-Technology-Economy, China.
††The author is with Beijing Institute of Science and Technology Information, China.
†††The author is with Beijing Institute of Graphic Communication, China.
a) E-mail: 493228928@qq.com (Corresponding author)
DOI: 10.1587/transinf.2019EDP7286

Copyright © 2020 The Institute of Electronics, Information and Communication Engineers
2. Related Work

To overcome the data sparsity of RS, researchers utilized contexts data as much as possible to improve the accuracy of recommendations. Since the available data is diverse, the data integration approaches are usually different from each other. E.g., Zhao et al. leveraged temporal information to solve the low accuracy problem of the recommender system for long term users based on recurrent neural network [13]. Yang et al. used multiple spatial information in POI (Point of Interests) recommendations, such as geographic locations and mobility. They proposed a semi-supervised learning framework to leverage unlabeled contexts data to alleviate data sparsity [14]. Ma et al. proposed a dynamic recommendation system named Hierarchical Temporal Collaborative Filtering (HTCF), which selects neighbors mainly based on hierarchical structure between items [15].

Although additional data is diverse, the usages of social information have recently attracted most attentions, especially for trusts. Duricic et al. applied a measure from network science, i.e. regular equivalence, to a trust network for generating similarity matrices in RS [16]. Li et al. explored user-item rating matrices to compute implicit social information, and then introduced an ISRec (Implicit Social Recommender) which integrates the implicit social information and the user-item rating matrix [17]. Feng et al. detected the latent associations among group members, in order to alleviate the data sparsity and improve the performance of group recommender systems [18].

Moreover, many researches particularly focused on the integration approaches of multi-source data. Ono et al. proposed methods to integrate a small amount of real situation data with a large amount of supposed situation data in context-aware users’ preference modeling to alleviate data sparsity [19]. Cheng et al. proposed a friend recommendation framework in social networks, where multiple sources have been integrated, including personal features, network structure features and social features. The framework is based on D-S evidence theory, which embodies the minimal conflicts among evidences [20].

Recently, several methods based on Graph Neural Networks (GNN) are proposed for integration of multi-source data in social recommendation scenarios. Wu et al. designed a neural architecture that organically combines the intrinsic relationship between social network structures and user-item interaction behaviors for social recommendation. They developed a model named collaborative neural social recommendation with a social embedding mechanism [21]. Later, Wu et al. proposed two improved models, called SocialGCN [22] and DiffNet [23], both models employ GCNs to capture how users’ preferences are influenced by the social diffusion process in social networks. Wang et al. developed a new recommendation framework Neural Graph Collaborative Filtering (NGCF), which exploits the user-item graph structure by propagating embeddings on it [24]. Fan et al. presented a novel graph neural network framework (GraphRec) to jointly capture interactions and opinions in the user-item graph for social recommendations [25].

In contrast with existing works, our approach tries to build a general framework, which converts available data to user ratings and leverages the social information naturally in the process of preference similarity calculations.

3. Preliminary

In this section, we introduce the semi-supervised deep learning framework GCNs and their usages in our study.

Many knowledge discovery tasks about social networks can be formally described with graphs. By analyzing graphs with Graph Convolutional Networks (GCNs), we can resolve node classification problems very well with high interpretability [26].

GCNs have similar theoretical basis with convolutional neural networks (CNNs). CNNs are comprised of several convolutional layers and fully connected layers, and well applied to almost all traditional problems of machine learning [27]. However, CNNs suffer a severe drawback that it is difficult to define operators on irregular (Non-Euclidean) data structures like graphs. By contrast, GCNs overcome the difficulty by two advantages [28]. Firstly, GCNs can extract information from graph structure without considering the orders of nodes in the input. Secondly, GCNs regard edges as the propagation paths of graph information rather than the features of nodes. In general, GCNs represent the state of nodes by a weighted sum of their adjacent nodes and collectively aggregate information from graph structures [29].

Although there are many variants of GCNs, we can briefly present the generalized framework of GCNs as follows:

In a graph, each node has hidden states, which are decided by many factors relating to the node. For each node \( v \), we denote its features, the features of its edges, the features of its neighbors and the hidden states of its neighbors as \( x_v, x_{e[v]}, x_{n[v]}, h_{n[v]} \) respectively. Let \( f_t \) be a local transition function. Then the hidden state \( h_v \) can be defined as follows:

\[
    h_v = f_t(x_v, x_{e[v]}, x_{n[v]}, h_{n[v]}) \tag{1}
\]

Where parameters of \( f_t \) are globally shared by all nodes. Moreover, we define a local output function \( o_v \), and then the output \( o_v \) of \( v \) is defined as follows:

\[
    o_v = f_o(x_v, h_v) \tag{2}
\]

Let \( H, O, X \) and \( X_N \) be the vectors constructed by stacking all the states, outputs, features and node features. Let \( F_i, F_o \) be the global function constructed by stacking \( f_i \) and \( f_o \) for all nodes. Then we have:

\[
    H = F_i(H, X) \tag{3}
\]

\[
    O = F_o(H, X_N) \tag{4}
\]

Based on the Banach’s fixed point theorem [30], we deem \( F_i \) and \( H \) as the contraction map and the fixed point
respectively. The convergence value of $H$ can be calculated by the following iterations when given any initial value of $H$:

$$H^{k+1} = F(H^k, X)$$  \hspace{1cm} (5)

Where $k$ represents k-th iteration.

Since $f_i$ and $f_o$ can be interpreted as the feedback neural network, we use gradient-descent method to learn their parameters. The loss function can be calculated by expected values and real output values of the supervised nodes.

In brief, since user ratings are usually integers within a particular range, we can regard the user ratings to an item as labels of nodes in networks. The change from social information to user ratings in social networks can be modeled as semi-supervised classifications of nodes with missing labels on graphs. GCNs leverage the social information contained in both user profiles and social relations as features of nodes. The components and steps of RecGCN can be described by the data flow in Fig. 1.

4. Trust Metrics Based on GCNs

4.1 Community Detection

At first we need to preprocess social networks by the technology of community detection, which aims to reduce the computation complexity of GCNs.

Our implementation of community detection is based on the betweenness measure. We perform a breadth-first search starting at vertex $s$ like Fig. 2, the weight $w_i$ on a vertex $i$ represents the number of distinct paths from the source vertex to $i$. If two vertices $i$ and $j$ are connected, with $j$ farther than $i$ from the source $s$, then the fraction of the path from $j$ through $i$ to $s$ is given by $w_i/w_j$.

Based on the above calculation process, we get the edge betweenness when source vertex is $s$. Iterations should be done by taking each vertex as source vertex, and then sum up betweenness results of all iterations in order to get the final edge betweenness.

We employ a top-down process by removing edges with high betweenness in order to get small size communities with suitable modularity values. Modularity is a measure of the structure of networks. Networks with high modularity have dense connections among nodes within communities but sparse connections among nodes in different communities. The value of the modularity lies in the range $[-1, 1][31]$. Consider an undirected network with $n$ nodes and $m$ edges such that the network can be partitioned into two communities using a membership variable $s$. If a node $v$ belongs to community 1, $s_v = 1$, or if $v$ belongs to community 2, $s_v = -1$. Let the adjacency matrix for the network be represented by $A$. Modularity $Q$ is then defined as the fraction of edges that fall within community 1 or 2, minus the expected number of edges within community 1 and 2 for a random graph with the same node degree distribution as the given network. Summing over all node pairs gives the equation for modularity. Let $k_v, k_w$ denote the degrees of nodes $v$ and $w$ respectively, and $Q$ is defined as follows:

$$Q = \frac{1}{2m} \sum_{v,w} (A_{vw} - \frac{k_v k_w}{2m}) s_v s_w + \frac{1}{2}$$  \hspace{1cm} (6)

In order to get communities with suitable size, we improve $Q$ as follows, where $n_t$ is a threshold of community size.

$$Q = \frac{1}{2m} \sum_{v,w} (A_{vw} - \frac{k_v k_w}{2m}) s_v s_w - \frac{1}{2} \frac{n - n_t}{n_t}$$  \hspace{1cm} (7)

After community detections, we generate a hierarchical tree, which consists of partitions and communities as follows. Moreover, we prune the trivial communities with unsuitable size or modularity for further reduction of computation complexity.

For each community, we build a GCN to retrieve knowledge from its history data. We choose items with global popularity to train the GCNs because there is sufficient labeled data (user ratings) to predict unlabeled data. Popular items can be easily picked out based on the mount of their ratings.

However, for each item, GCNs only predict the ratings of users within communities. Hence, we design an additional inference mechanism of default user ratings based on Bayesian network [32] for those who have been pruned.
We use Bayesian network to model the above hierarchical tree in order to learn the conditional probabilities of average ratings between each community and its sub communities. Let \( C, S_i \) denote the average ratings of a community and its \( i \)th sub community respectively, and \( S = \langle S_1, \ldots, S_k \rangle \). Let \( D \) represent a dataset with \( M \) history records. We use Maximum Likelihood Estimation (MLE) to learn the parameters \( \theta = \langle P(S_1 | C) \ldots P(S_k | C) \rangle \).

\[
L(\theta : D) = \prod_{m=1}^{M} P(C[m], S[m] : \theta) \\
= \prod_{m=1}^{M} P(C[m] : \theta) P(S[m] | C[m] : \theta) \\
= \left( \prod_{m=1}^{M} P(C[m] : \theta) \right) \left( \prod_{m=1}^{M} P(S[m] | C[m] : \theta) \right) \quad (8)
\]

\[
\tilde{\theta} = \arg\max_{\theta} L(\theta : D)
\]

After GCNs predicted ratings of leaf communities, we can infer the average ratings of their parent communities from the bottom up. We use the average ratings as default ratings of pruned users to alleviate the data sparsity.

\[
P(C | S) = \frac{P(S | C) P(C)}{P(S)} \\
= \frac{P(S_1 \ldots S_k | C) P(C)}{P(S_1) \ldots P(S_k) P(C)} \\
= \frac{P(S_1 | C) \ldots P(S_k | C) P(C)}{P(S_1) \ldots P(S_k)} \quad (9)
\]

4.2 Semi-Supervised Learning with GCNs

The inputs of GCNs include community structures, various user profiles and user ratings on items. For each item, the inputs can be regarded as graph structures, features of nodes and classifications of nodes respectively. Due to the data sparsity of user ratings, the classifications data is usually a small amount of labeled data with a large amount of unlabeled data. Since GCNs are semi-supervised learning methods, they are able to predict labels of unlabeled data as output, which are also user ratings on the item. In order to ensure the accuracy of predictions, GCNs only choose popular items, which own plenty of ratings as training data. The structure of a GCN is as Fig. 4.

As mentioned above in Sect. 3, we define a multi-layer GCN with the following layer-wise propagation rule:

\[
H^{(k+1)} = \sigma(D^{-\frac{1}{2}} \tilde{A} D^{-\frac{1}{2}} H^{(k)} W^{(k)}) \quad (10)
\]

Where \( H^{(k)} \) is the matrix of activations in the \( k \)th layer, and \( H^{(0)} \) is the input features of all nodes in graphs. \( \sigma(\cdot) \) denotes a nonlinear activation function, i.e. \( \text{ReLU}(\cdot) = \max(0, \cdot) \). Let \( I_N \) and \( \tilde{A} \) denote the identity matrix and adjacency matrix of the undirected graphs respectively, then \( \tilde{A} = A + I_N \) and \( \tilde{D}_{ii} = \sum_j \tilde{A}_{ij} \). \( W^{(k)} \) is the trainable weight matrix of the \( k \)th layer.

In order to reduce the computational complexity, we achieve the effect of the above propagation rule via a first-order approximation of localized spectral filters on graphs [33].

For each layer, we define spectral convolutions on graphs as the multiplication of a signal \( x \in R_N \) (a scalar for every node) with a filter \( g_{\theta_0} \): 

\[
g_{\theta} * x = \theta \left( \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} x \right) \quad (11)
\]

Where \( \theta \) is a single free parameter, which is shared over the whole graph.

We can generalize the above definition to a signal \( X \in R^{N \times M} \) with \( M \) input channels (i.e. a \( M \)-dimensional feature vector for every node) and filters \( F \) as follows:

\[
Y = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X \Theta \quad (12)
\]

Where \( \Theta \in R^{M \times F} \) is a matrix of filter parameters and \( Y \in R^{N \times F} \) is the convolved signal matrix.

Based on the above description, we design a two-layer
GCN for semi-supervised user property predictions on a graph.

\[
Z = \text{softmax} \left( \hat{A} \text{ReLU}(\hat{A}XW(0)W(1)) \right)
\]

\[
\hat{A} = \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}}
\]

\[
\text{ReLU}(x) = \begin{cases} 
  x, & \text{if } x > 0 \\
  0, & \text{if } x \leq 0 
\end{cases}
\]

\[
\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum \exp(x_i)}
\]

Where \(W(0) \in R^{M \times H}\) is an input-to-hidden weight matrix for a hidden layer with \(H\) feature maps. \(W(1) \in R^{H \times P}\) is a hidden-to-output weight matrix.

For semi-supervised multiclass classification, we then evaluate the cross-entropy error over all labeled examples (supervised nodes). The weights \(W(0)\) and \(W(1)\) are trained using gradient descent. In this work, we perform batch gradient descent using the full dataset for every training iteration.

4.3 Metrics of Direct and Indirect Trusts

After GCNs complete the prediction of user ratings, we calculate the direct-trust values between any two neighbors as follows:

\[
T_{\text{direct}}(u, v) = \frac{R(u) \cdot R(v)}{||R(u)|| \cdot ||R(v)||}
\]

\[
= \frac{\sum_{i=1}^{n} u_i v_i}{\sqrt{\sum_{i=1}^{n} (u_i)^2} \times \sqrt{\sum_{i=1}^{n} (v_i)^2}}
\]

Where \(u, v\) are neighbor users, and \(R(\cdot)\) represents the vector of their ratings on \(n\) items.

Then, we design an indirect-trust inference method to measure the trust values for every pair of non-adjacent users within the same community. Given a community, transitivity is the assumption for propagating and inferring trusts. Our algorithm performs a modified breadth-first search in social networks to compute indirect-trust \(T_{\text{indirect}}(u, v)\) between a source user \(u\) and a target user \(v\).

The algorithm starts with \(u\), and finds all the shortest paths \(P\) to \(v\) within the scope of the community \(c\). Each path \(p \in P\) has length, strength, and several edges, we denote them as \(L(p), S(p)\) and \(E(p)\) respectively. We compute\(T_{\text{indirect}}(u, v)\) as follows:

\[
T_{\text{indirect}}(u, v) = \min(T_{\text{direct}}(e) \mid e \in E(p))
\]

\[
\delta = 1 - \frac{L(p)}{D(c)}
\]

Where \(D(c)\) represents the diameter of \(c\), and we use the parameter \(\delta\) to avoid the excessive propagation of trusts so as to keep the diversity of recommendations. E.g., In Fig. 5, if \(D(c) = 6\), then \(T_{\text{indirect}}(u, v) = 0.4\).

5. RecGCN Based on Matrix Factorization

We improve the traditional CF method based on matrix factorization by adding a social regularization in order to avoid overfitting and alleviate data sparsity.

Let \(R_{N \times D}\) denote an existing rating matrix, which contains ratings of \(N\) users to \(D\) items. We can factorize \(R_{N \times D}\) as follows:

\[
R_{N \times D} = \begin{bmatrix} p_1 & \cdots & q_1 \\ p_2 & \cdots & q_2 \\ \vdots & \vdots & \vdots \\ p_N & \cdots & q_D \end{bmatrix}^T
\]

The prediction of user ratings and the loss function can be calculated as follows:

\[
r_{ij} = \mu_j + w_j \sum_{k=1}^{K} p_{ik} q_{jk}
\]

\[
Loss_{P,Q} = \frac{1}{2} \sum_{i,j} \left( r_{ij} - \sum_{k=1}^{K} p_{ik} q_{jk} \right)^2
\]

To avoid overfittings, we use L2-Norm as regularizations.

\[
Loss_{P,Q} = \frac{1}{2} \sum_{i,j} \left( r_{ij} - \sum_{k=1}^{K} p_{ik} q_{jk} \right)^2 + \frac{\lambda_p}{2} \sum_{i=1}^{N} \|p_i\|_2^2 + \frac{\lambda_q}{2} \sum_{j=1}^{D} \|q_j\|_2^2
\]

Considering the \(\mu\): global bias, \(m_i\): user bias and \(w_j\): item bias, we modify the expression of user ratings and loss function as follows:

\[
r_{ij} = \mu + m_i + w_j + \sum_{k=1}^{K} p_{ik} q_{jk}
\]

\[
Loss_{P,Q,M,W} =
\]

\[
\frac{1}{2} \sum_{i,j} \left( r_{ij} - \mu - m_i - w_j - \sum_{k=1}^{K} p_{ik} q_{jk} \right)^2 + \frac{\lambda_p}{2} \sum_{i=1}^{N} \sum_{k=1}^{K} p_{ik}^2 + \frac{\lambda_q}{2} \sum_{j=1}^{D} \sum_{k=1}^{K} q_{jk}^2 + \frac{\lambda_m}{2} \sum_{i=1}^{N} m_i^2 + \frac{\lambda_w}{2} \sum_{j=1}^{D} w_j^2
\]

In order to incorporate social information into the recommendation process, we add a social regularization into
the loss function by considering the association between user trusts and their ratings.

\[
\text{Loss}_{PQM,W} = \\
\frac{1}{2} \sum_{i,j} \left( r_{ij} - \mu - m_i - w_j - \sum_{k=1}^{K} p_k q_{jk} \right)^2 \\
+ \frac{\lambda}{2} \sum_{i=1}^{N} \sum_{k=1}^{K} p_{ik}^2 \\
+ \frac{\lambda}{2} \sum_{j=1}^{D} \sum_{k=1}^{K} q_{jk}^2 \\
+ \frac{\lambda}{2} \sum_{i=1}^{N} \sum_{j=1}^{D} \left( T(u,v) - \text{Sim}_r(u,v) \right)^2
\]

Where \( T(u,v) \) and \( \text{Sim}_r(u,v) \) represents the trust values between \( u \) and \( v \), and the similarity of their ratings respectively.

\[
\text{Sim}_r(u,v) = \frac{R(u) \cdot R(v)}{||R(u)|| \cdot ||R(v)||} \\
= \frac{\sum_{l=1}^{n} u_l v_l}{\sqrt{\sum_{l=1}^{n} (u_l)^2} \cdot \sqrt{\sum_{l=1}^{n} (v_l)^2}}
\]

\[
T(u,v) = \begin{cases} 
T_{\text{direct}}(u,v), & v \in \text{neighbor}(u) \\
T_{\text{indirect}}(u,v), & v \notin \text{neighbor}(u)
\end{cases}
\]

Gradient descent method is used to calculate the parameters.

\[
p_{ik}^{t+1} = p_{ik}^t - \alpha \frac{\partial \text{Loss}_{PQM,W}}{\partial p_{ik}} \\
q_{jk}^{t+1} = q_{jk}^t - \alpha \frac{\partial \text{Loss}_{PQM,W}}{\partial q_{jk}} \\
m_{i}^{t+1} = m_{i}^t - \alpha \frac{\partial \text{Loss}_{PQM,W}}{\partial m_{i}} \\
w_{j}^{t+1} = w_{j}^t - \alpha \frac{\partial \text{Loss}_{PQM,W}}{\partial w_{j}}
\]

Where \( \alpha \) represents the rate of learning, and we should choose \( \alpha \) according to the following principle:

\[
\alpha_{\text{opt}} = \min \left\{ \alpha \left| \frac{\partial \text{Loss}_{PQM,W}}{\partial \alpha} \right| \right\}
\]

After we get the result of matrix factorization, the prediction of user ratings can be calculated and used in recommendation processes.

6. Experiments

6.1 Experiment Setup

We carried out experiments based on real datasets. We collected data from a website of BUCT (my.buct.edu.cn), which helps students to make their career plans and recommends related MOOC (massive open online courses). In this website, users follow each other and provide their personal information, such as age, gender, hobby and major. They also take Holland Career Test and give ratings to the recommended resources. Hence, we have plenty of social information to verify our method. Our dataset contains social information of 6511 students, and covers a 6 month period from June 2018 to December 2018. All users give ratings to 21000 resources, and the density of our data is about 2.3%.

In addition, we also compared RecGCN with various state-of-the-art methods based on a subset of the Yelp dataset (www.yelp.com/dataset), which is commonly used in social recommender researches. Yelp is an online location-based social websites, and users express their experiences by ratings. The subset includes 3400 users and 7100 items. The rating density and link density of the subset are 0.027% and 0.046% respectively.

We implemented GCNs with an ADAM optimizers and softmax cross entropy loss functions based on an open source GCN framework implemented in Tensorflow, which can be accessed from https://github.com/tkipf/gcn. The following analyses are carried out on a server equipped with Intel i7-6850K and NVIDIA GTX1080TI. The pseudo code of building a GCN is as Fig. 6.

6.2 Accuracy of GCNs Predictions

For each community, GCNs predict user ratings on popular items by semi-supervised learning. We selected 5 GCNs to calculate the accuracy of user rating predictions. We picked a part of user ratings on 100 popular items as training set (labeled data) and used the rest ratings to verify the accuracy. The results are as Table 1.

The above results in Table 1 show GCNs can predict user ratings accurately. Both the density of existing user ratings and the complexity of community structures can help improve the prediction accuracy of GCNs.

6.3 Quality of Data Transformation

Since GCNs transform the original social information into user ratings, we should ensure the correctness of the transformation. The generated ratings should be consistent with the existing ratings, both of which reflect the actual user
Table 1  Accuracy of GCN predictions

| Size of Community | Clustering Coefficient | Ratio of Labeled data | Accuracy |
|-------------------|------------------------|-----------------------|----------|
| 86                | 0.176                  | 40%                   | 0.64     |
| 105               | 0.184                  | 40%                   | 0.49     |
| 127               | 0.269                  | 40%                   | 0.57     |
| 143               | 0.272                  | 40%                   | 0.63     |
| 279               | 0.287                  | 40%                   | 0.69     |
| 86                | 0.176                  | 60%                   | 0.52     |
| 105               | 0.184                  | 60%                   | 0.60     |
| 127               | 0.269                  | 60%                   | 0.74     |
| 143               | 0.272                  | 60%                   | 0.75     |
| 279               | 0.287                  | 60%                   | 0.82     |

Similarities. Hence, we picked 795 pairs of users randomly
to calculate the similarities of existing ratings and generated
ratings respectively.

Figure 7 shows there is a consistency between existing
ratings and generated ratings, which means the transforma-
tion from social information to user ratings is reasonable.

If we take the distance between each pair of users into
consideration, and then we get Fig. 8 as follows. Figure 8
shows the generated ratings can reflect the real trust rela-
tionships in real world.

6.4 RecGCN vs. Traditional CF

In order to compare RecGCN and traditional CF (CF based
on matrix factorization), we firstly used 4 indexes to evalu-
ate them, including accuracy, precision, recall and F-
measure. Let T, F, P and N denote True, False, Positives
and Negatives respectively, and the indexes are calculated
as follows:

\[
\begin{align*}
\text{accuracy} & = \frac{TP + TN}{P + N} \\
\text{precision} & = \frac{TP}{TP + FP} \\
\text{recall} & = \frac{TP}{TP + FN} \\
F - \text{measure} & = \frac{2}{\frac{1}{\text{precision}} + \frac{1}{\text{recall}}} 
\end{align*}
\]

(25)

We used the same dataset to train and test RecGCN
and a traditional CF based on matrix factorization, which
only take the existing ratings as inputs, and then we got the
results as Table 2.

Table 2 shows RecGCN has better performances than
the traditional collaborative filtering method. RecGCN is
able to absorb more information to infer user ratings.

Besides, since RecGCN propagates trust relations
among users in the process of trust inference, it may lead to
excessive similarity on recommendations of different users.
Hence we compared the diversities of RecGCN and CF by
calculating the average similarities of their predicted ratings
between pairs of users with different distances. The selected
users are coming from the same communities and the results
are as Fig. 9.

Figure 9 shows RecGCN has an approximate diversity
on recommendations in contrast with traditional CF.

6.5 RecGCN vs. Other Existing Methods

To evaluate the performance, we compared RecGCN with
existing methods including a traditional recommendation
method (matrix factorization, MF), a traditional social rec-
ommendation method (ISRec [17]), and Graph Neural Net-
works based methods (SocialGCN [22], DiffNet [23]). All
the methods above gave detailed explanations of their im-
plementations, and we carried out experiments with Yelp
dataset, and then we got the results as Table 3.

Table 3 shows deep learning methods have better performances than traditional methods. Additionally, RecGCN performs as well as other state-of-the-art methods. RecGCN not only learn the associations between social information and user ratings but also capture the process of how users’ preferences are influenced by GCNs and trust propagations.

### 6.6 Effect of Alleviating Data Sparsity

The target of this research is trying to alleviate data sparsity by incorporating additional social information. We delete ratings of the same dataset in different proportions, and then evaluate the accuracy of CF and RecGCN as Fig. 10.

Figure 10 shows RecGCN is able to alleviate data sparsity effectively. In addition, we validated the extendibility of RecGCN by incorporating different numbers of dimensions of social information. We used the above datasets and got the results as Fig. 11.

Figure 11 shows RecGCN has a good extendibility in incorporating additional information into recommendation processes.

In order to understand what’s the effectiveness of each part in the RecGCN, we removed indirect-trust computation in RecGCN as RecGCN*, and removed all trust computation in RecGCN as RecGCN**. Then, we removed some attributes from the Yelp dataset about users such as “review_count”, “cool”, “useful” and “funny” (RecGCN***), and we got the results as Table 4.

### 7. Conclusions

In this paper, we aims to overcome the inherent limitation, which is referred to as data sparsity in CF. We proposed an improved CF method, called RecGCN, based on Graph Convolutional Networks. RecGCN integrates multi-source social information by employing a novel trust metrics in social networks. RecGCN transforms the social information into user ratings in order to create a social regularization in user-item rating matrix factorization. The experiments on real datasets show that RecGCN outperforms the traditional CF and some existing social recommendation methods by using additional data to alleviate data sparsity. RecGCN also has a good Extendibility.

### Acknowledgments

This study was supported by national social science fund of China (No. 15CXW005).

### References

[1] S.-C. Lee, S.-W. Kim, S. Park, and D.-K. Chae, “An approach to effective recommendation considering user preference and diversity simultaneously,” IEICE Trans. Inf. & Syst., vol. E101-D, no.1, pp. 244–248, 2018.
[2] T.H. Ma, J. Zhou, M. Tang, Y. Tian, A. Al-Dhelaan, M. Al-Rodhaan, and S. Lee “Social network and tag sources based augmenting collaborative recommender system,” IEICE Trans. Inf. & Syst., vol. E98-D, no.4, pp. 902–910, 2015.
[3] R.M. Bell and Y. Koren, “Lessons from the netflix prize challenge,” ACM SIGKDD Explorations Newsletter, vol. 9, no. 2, pp. 75–79, 2007.
[4] Y. Koren, R.M. Bell, and C. Volinsky, “Matrix factorization techniques for recommender systems,” Computer, vol. 42, no. 8, pp. 30–37, 2009.
[5] L. Quijano-Sanchez, J.A. Recio-Garcia, B. Diaz-Agudo, and G. Jimenez-Diaz, “Social factors in group recommender systems,” ACM Transactions on Intelligent Systems and Technology, vol. 4, no. 1, pp. 1–30, 2013.
[6] Z.-K. Zhang, C. Liu, Y.-C. Zhang, and T. Zhou, “Solving the
cold-start problem in recommender systems with social tags,” Europhysics Letters, vol.92, no.2, pp.28002–28003, 2010.

[7] Z. Wang, L.F. Sun, W.W. Zhu, S. Yang, H. Li, and D. Wu, “Joint social and content recommendation for user-generated videos in online social network,” IEEE Trans. Multimedia, vol.15, no.3, pp.698–710, 2013.

[8] J. Golbeck, Computing and Applying Trust in Web-based Social Networks, University of Maryland College Park, 2005.

[9] P. Massa and P. Avesani, “Trust-aware recommender systems,” Proc. RecSys 2007, New York, USA, pp.17–24, 2007.

[10] P. Massa and P. Avesani, Trust Metrics in Recommender Systems, Computing with Social Trust, pp.259–285, Springer, London, 2009.

[11] B. Pal and M. Jemani, “Trust inference using implicit influence and projected user network for item recommendation,” Journal of Intelligent Information Systems, vol.52, no.2, pp.425–450, 2019.

[12] F.E. Walter, S. Battiston, and F. Schweitzer, “A model of a trust-based recommendation system on a social network,” Autonomous Agents and Multi-Agent Systems, vol.16, no.1, pp.57–74, 2008.

[13] Z.Y. Zhao, M. Zhu, Y.Q. Sheng, and J. Wang, “A top-n-balanced sequential recommendation based on recurrent network,” IEICE Trans. Inf. & Syst., vol.E102-D, no.4, pp.737–744, 2019.

[14] C. Yang, L. Bai, C. Zhang, Q. Yuan, and J. Han, “Bridging collaborative filtering and semi-supervised learning: a neural approach for POI recommendation,” ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Nova Scotia, Canada, pp.1245–1254, 2017.

[15] T.H. Ma, L.M. Guo, M.L. Tang, Y. Tian, M. Al-Rodhaan, and A. Al-Dheelaan, “A collaborative filtering recommendation algorithm based on hierarchical structure and time awareness,” IEICE Trans. Inf. & Syst., vol.E99-D, no.6, pp.1512–1520, 2016.

[16] T. Duricic, E. Lacic, D. Kovald, and E. Lex, “Trust-based collaborative filtering: tackling the cold start problem using regular equivalence,” Proc. RecSys 2018, New York, USA, pp.446–450, 2018.

[17] Y.S. Li, M.N. Song, and H.H. E, “Recommender system using implicit social information,” IEICE Trans. Inf. & Syst., vol.E98-D, no.2, pp.346–354, 2015.

[18] S.S. Feng, H.X. Zhang, L. Wang, L. Liu, and Y. Xu, “Detecting the latent associations hidden in multi-source information for better group recommendation,” Knowledge Based Systems, vol.171, no.1, pp.56–68, 2019.

[19] G. Oono, Y. Takishima, Y. Motomura, H. Asoh, Y. Shinagawa, M. Imai, and Y. Anzai, “Context-aware users’ preference models by integrating real and supposed situation data,” IEICE Trans. Inf. & Syst., vol.E91-D, no.11, pp.2552–2559, 2008.

[20] S.L. Cheng, B.F. Zhang, G.B. Zhou, M. Huang, and Z. Zhang, “Friend recommendation in social networks based on multi-source information fusion,” International Journal of Machine Learning and Cybernetics, vol.10, no.5, pp.1003–1024, 2019.

[21] L. Wu, P. Sun, R. Hong, Y. Ge, and M. Wang, “Collaborative neural social recommendation,” IEEE Trans. Syst., Man, Cybern., Syst., vol.99, no.1, pp.1–13, 2018.

[22] L. Wu, P. Sun, R. Hong, et al., “SocialGCN: An efficient graph convolutional network based model for social recommendation,” Proc. AAAI 2019, Honolulu, USA, pp.571–578, 2019.

[23] L. Wu, P. Sun, Y. Fu, R. Hong, X. Wang, and M. Wang, “A Neural Influence Diffusion Model for Social Recommendation,” Proc. SIGIR 2019, Paris, France, pp.20–30, 2019.

[24] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, “Neural Graph Collaborative Filtering,” Proc. SIGIR 2019, Paris, France, pp.121–131, 2019.

[25] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, “Graph Neural Networks for Social Recommendation,” Proc. WWW 2019, San Francisco, USA, pp.47–58, 2019.

[26] J. Zhou, G. Cui, Z. Zhang, et al., “Graph neural networks: A review of methods and applications,” arXiv: Learning, 2018.

[27] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” Nature, vol.521, no.7553, pp.436–438, 2015.

[28] M.M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, “Geometric deep learning: going beyond euclidean data,” IEEE Signal Process. Mag., vol.34, no.4, pp.18–42, 2017.

[29] T.N. Kipf and M. Welling, “Semi-supervised classification with graph convolutional networks,” ICLR 2017, Toulon, France, pp.1–9, 2017.

[30] M.A. Khamsi and W.A. Kirk, An Introduction to Metric Spaces and Fixed Point Theory, John Wiley & Sons, New York, 2011.

[31] M.E.J. Newman, “Modularity and community structure in networks,” Proc. National Academy of Sciences of the United States of America, vol.103, no.23, pp.8577–8606, 2006.

[32] D. Koller and N. Friedman, Probabilistic Graphical Models: Principles and Techniques, The MIT Press, Boston, 2009.

[33] M. Defferrard, X. Bresson X, and P. Vandergheynst, “Convolutional neural networks on graphs with fast localized spectral filtering,” Advances in Neural Information Processing Systems, vol.29, no.1, pp.3844–3852, 2016.

Haitao Xie received the Ph.D. degree in computer science from the Beijing University of Posts and Telecommunications, Beijing, China, in 2012. He is currently a researcher at Beijing United Information Center of Science-Technology-Economy. His main research interests include deep learning and large-scale data mining.

Qingtao Fan received the M.S. degree from Beijing Institute of Technology, Beijing, China, in 2008. Now he works in Beijing Institute of Science and Technology information, his main interests includes management of information resource, evaluation of S&T, data mining and scientometrics.

Qian Xiao received the B.S. and Ph.D. degrees from the Beijing University of Posts and Telecommunications, Beijing, China, in 2007 and 2012 respectively. She is currently an assistant professor at Beijing Institute of Graphic Communication. Her main research interest include social media and social network analysis.