SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •
Special Focus on Natural Language Processing and Social Computing

November 2017, Vol. 60 110104:1–110104:22
doi: 10.1007/s11432-017-9243-7

Integrating a weighted-average method into the random walk framework to generate individual friend recommendations

Jibing GONG[1,2,4,5], Xiaoxia GAO[1,2*], Hong CHENG[3], Jihui LIU[1,2], Yanqing SONG[1,2], Mantang ZHANG[1,2] & Yi ZHAO[1,2]

[1]*School of Information Science and Engineering, Yanshan University, Qinhuangdao* 066044*, China;*
[2]*The Key Lab for Computer Virtual Technology and System Integration, Yanshan University, Qinhuangdao* 066044*, China ;*
[3]*Department of Systems Engineering and Engineering Management, The Chinese University of Hong Kong, Hong Kong, China;*
[4]*State Key Lab of Mathematical Engineering and Advanced Computing, Wuxi* 214000*, China;*
[5]*Key Laboratory for Software Engineering of Hebei Province, Yanshan University, Qinhuangdao* 066044*, China*

Received August 15, 2017; accepted September 8, 2017; published online October 13, 2017

Abstract Friend recommendation is a fundamental service in both social networks and practical applications, and is influenced by user behaviors such as interactions, interests, and activities. In this study, we first conduct in-depth investigations on factors that affect recommendation results. Next, we design Friend++, a hybrid multi-individual recommendation model that integrates a weighted average method (WAM) into the random walk (RW) framework by seamlessly employing social ties, behavior context, and personal information. In Friend++, the first plus signifies recommending a new friend through network features, while the second plus stands for using node features. To verify our method, we conduct experiments on three social datasets crawled from the Sina microblog system (Weibo). Experimental results show that the proposed method significantly outperforms six baseline methods in terms of recall, precision, F1-measure, and MAP. As a final step, we describe a case study that demonstrates the scalability and universality of our method. Through discussion, we reach a meaningful conclusion: although common interests are more important than user activities in making recommendations, user interactions may be the most important factor in finding the most appropriate potential friends.

Keywords multi-individual friend recommendation architecture, behavior context analysis, Intimacy degree, random walk framework, social networks

Citation Gong J B, Gao X X, Cheng H, et al. Integrating a weighted-average method into the random walk framework to generate individual friend recommendations. Sci China Inf Sci, 2017, 60(11): 110104, doi: 10.1007/s11432-017-9243-7

* Corresponding author (email: gaoxxysu@163.com)

1 Introduction

An increasing number of people interact through social networks (SN); however, it is time-consuming and difficult for them to find desired information (e.g., new friends) from among the vast amount of content on these networks. Individual friend recommendation is an important task in various social networking-based applications; these include searching product information and ratings, recommending advertisements and services, and public sentiment surveillance. Friend recommendation has been extensively studied in traditional social networks [1]. Numerous recommendation methods have been proposed, including collaborative filtering [2], content-based recommendation [3], random walk (RW) models [4], graph models [5], and so on. Most existing RW-based methods utilize social ties for modeling, but rarely consider the behavior contexts or personal information of users. Compared to social ties data, behavior data can not only include interaction information of @, but also provide user context information such as forwarded posts, reply posts, and comments. Moreover, the personal information of users can offer relatively more obvious preference labels for personalized recommendations. Hence, an RW-based friend recommendation system that considers both behavior context and personal information can be more useful for finding new friends in the age of personalized user behavior.

In this paper, we propose a novel friend recommendation architecture, Friend++; the first plus signifies recommending a new friend through network features, while the second plus stands for using node features. Our friend recommendation algorithm considers social ties, behavior context, personal information, and second-degree friends (friends of friends). Friend++ first extracts network and node features, and then integrates a weighted average method (WAM) into the random walk model to make multi-individual friend recommendations. Here, network features can be defined as Intimacy degree [6], Trust Degree [7], or Interaction Degree [8]. In addition, we define the node feature InterestDeg according to personal data, and InterestActivity based on behavior context data. Moreover, the type and number of node features depend on the practical applications using Friend++. We note that WAM could also be replaced with other ranking methods such as support vector machines (SVMs) [9] or logistic regression (LR) [10]. Hence, considering all the factors mentioned above, we can easily discover the two primary characteristics of Friend++: scalability and universality.

Furthermore, in this study an instance of Friend++ is built according to real requirements from Sina Weibo. In this case, we instantiate the network feature to Intimacy degree, which was introduced in [6]. It describes the total number of @ operations that two users conduct with each other in social networks. In this study, we utilize a random-walk-based algorithm that is similar to an algorithm described in [6,11]; the main difference between the algorithms lies in the RW model construction phase. Friend++ constructs an RW model using relatively more personalized social data, including social ties, personal information, and user behaviors. More specifically, we formally define one network feature and two node features. We construct an undirected unweighted graph by defining the relationships between users, we build the state transition probability matrix of the graph, and we design a random-walk-with-restart algorithm to implement the Friend++ instance using the following parameters: WAM, Intimacy degree, InterestDeg, and InterestActivity. Finally, we conduct experiments to verify our method on three real social datasets crawled from the Sina microblog system (Weibo).

To extend the design of [6], we implement several major updates: (1) We propose a hybrid architecture for multi-individual friend recommendation in social networks in Subsection 2.2. (2) We define a network feature (Intimacy degree) and two novel individual node features (InterestDeg and InterestActivity) in Subsections 2.3.1 and 2.3.2. (3) Based on the features above, we use a weighted-average method to rank nodes; this is described in Subsection 2.3.3. (4) We conduct performance studies to compare our proposed system against existing algorithms, and perform a factor contribution analysis; this is described in Subsections 3.3 and 3.4. (5) Using a case study, we provide an in-depth analysis of the proposed method in Subsection 3.5.

Most existing random-walk-based methods utilize social ties for modeling, but rarely consider the behavior contexts or personal information of users. Hence, it remains a major challenge to design a unified method that can integrate the above factors to make friend recommendations. More specifically,

Table 1 Definition of variables

Symbol	Description
U	Set of target users
C	Set of candidate users
E	Set of relationships among users
$P_{i,j}$	Probability of moving from point i to point j
P	State transition probability matrix
$\bar{P}^k_{i,j}$	Column vector of matrix P
$\bar{R}^*_{u,\_}$	Row vector of matrix P
$\tilde{R}$	Reflects the target user's rating of a user
$\tilde{W}$	Adjacency matrix
e_j	An initial vector
R_t	Probability distribution vector after t iterations
$w(i,j)$	User i's interactive value to user j
$c(i,j)$	Number of mutual friends between user i and user j
t_i	Number of days ago that an interaction took place
$m(t)$	Interaction time between i and j during the last five days
$\phi(t)$	The time attenuation function
k_j	Degree of recommended user j

we must determine: (1) how to measure and utilize these factors (i.e., interactions with amicable chatting between users) to make recommendations, (2) which factor has the greatest influence on the results, and (3) the motivation of the target user for accepting the recommended new friends. In this study, we (1) integrate WAM into the random walk framework as a unified friend recommendation architecture/model, (2) define hybrid features and parameters of RW, and (3) analyze factors that influence recommendation performance via real-world experiments, observations in practice, and a Weibo-based application case. From the perspective of our method's process, we first use InterestDeg and InterestActivity as WAM inputs to rank nodes of candidate friends. Then, by combining Intimacy degree with those ranked values, the state transition probability matrix of the model is produced using hybrid features that are classified into network features and node features. Afterward, the matrix is provided as an input to our random walk algorithm, which calculates the friend recommendation probabilities of users. Finally, a list of potential friends is provided to the user, based on output from the random walk algorithm.

The main contribution of this work is that social ties, behavior context, and personal information are employed seamlessly in friend recommendation. We build an individual friend recommendation model that integrates the weighted average method into the random walk framework. One network feature and two novel node features are extracted locally and employed in friend recommendation. Moreover, when new users and behaviors are introduced into databases, our recommendation method (unlike most random walk (RW) methods) does not need to update any existing model or structure (such as a state transition probability matrix). In some cases, the network and node features (or WAM) are fixed for our method, and in other cases, they can be replaced; this determination will be based on the application. The result is that our method is scalable and universal, and consequently it can be dynamically implemented and efficiently deployed on a new application.

The remainder of our paper is organized as follows: In Section 2, we define two new types of individual features for friend recommendation: a network feature (Intimacy degree) and two node features (InterestDeg and InterestActivity); we also present the hybrid architecture (random walk framework + weighted average method) for friend recommendation. Section 3 describes experimental details and validations of our results, and describes a case study. Section 4 provides a detailed discussion of our results. Finally, Section 5 introduces related work and Section 6 discusses our conclusion.

In this paper, notations are summarized in Table 1.

2 Our approach: Friend++

2.1 Problem definition

In this subsection, we provide several preliminary definitions and then present a formal definition of the problem.

A static mobile social network can be represented as $G = \langle V, E\rangle$, where V is the set of $|V| = N$ users and $E \subset V \times V$ is the set of undirected links between users. Each user has various activities in the mobile social network.

Definition 1 ($t-$Dynamic continuous network)**.** A dynamic continuous network (from time 0 to t) is denoted as $G^t = (V, E^t, X^t, A^t)$, where V is the set of users, $e^t_{ij} \in E^t$ is the edge between users v and u up to time t, X^t represents the continuous time-evolving attributes of all users in the network, and A^t represents the set of activities of all users in the social network.

Problem 1. We formulate the friend recommendation problem as follows: G is a graph that is an instance of the SN data model described above. Given the users' behaviors and corresponding graph G, our aim is to predict potential friends for a user u in U. The potential friends list is an ordered list of elements from U with size N, which is the desired number of recommendations. The goal is to generate a potential friend list with the highest accuracy; in other words, with minimal errors relative to the user's future friends.

The fundamental challenge of this problem is obtaining the unified state transition probability matrix P to run the RW algorithm. Specifically, we first define the network feature and obtain a preliminary P, and then we define the node features and integrate them into P after ranking the nodes using WAM. Finally, we run the final RW-based algorithm with updated matrix P to obtain a ranked candidate friends list for the target user.

2.2 Architecture of Friend++

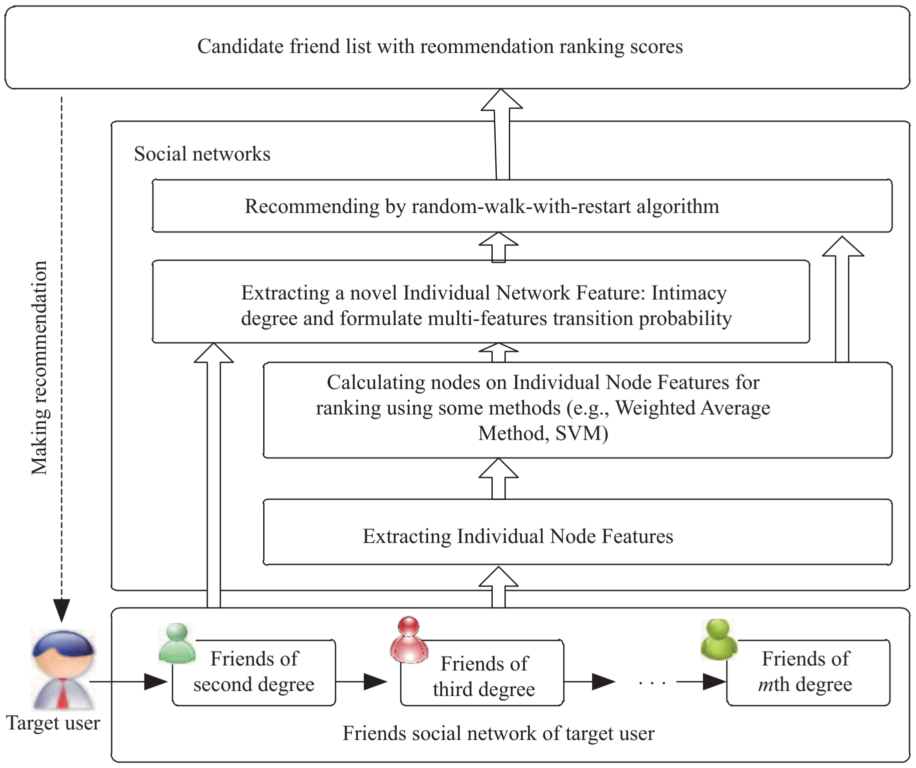
In this study, we design and implement Friend++, a hybrid and multi-individual friend recommendation architecture, as shown in Figure 1.

Figure 1 shows the construction of the target user's friend social network, which includes the target user's second-degree friends, third-degree friends, etc. until the target user's mth-degree friends. Moreover, the infrastructure of Friend++ has three parts: (1) the individual network feature (i.e., Intimacy degree) measuring the degree of interactions between users, (2) the random-walk-with-restart (RWR) framework for ranking the list of candidate friends, and (3) the state transition probability matrix being calculated according to Intimacy degree and running on RWR. Considering its universality and scalability, the variable parts of the architecture have (1) individual node features and (2) some type of ranking method. In practice, we can define some desired node features or utilize any type of algorithm or method to calculate the ranking values of nodes, depending on the practical applications used. We can consider user profile information and the interactive information between users in social networks, and finally improve recommendation accuracies.

Aiming at the characteristics of social networks, the proposed solution is based on the random walk algorithm, and introduces an individual friend recommendation method that uses interaction information to rate the friends of the target user. The random walk (RW) algorithm uses a first-order Markov model, and thus the status of the user's probability at time $t+1$ can only be influenced by the status at time t on the graph structure. This is expressed by

$$\Pr\left(X_{u,t+1} = i \mid X_{u,t}\right). \tag{1}$$

Let C represent the set of candidate friends in the system and m represents the size of the set of candidate friends. Thus, a directed graph $G = \langle V, E\rangle$ that has m points can be established. Weight $P_{i,j}$ represents the probability of moving from point i to j on the graph. According to the description of



Candidate friend list with reommendation ranking scores
Social networks
Recommending by random-walk-with-restart algorithm
Extracting a novel Individual Network Feature: Intimacy degree and formulate multi-features transition probability
Calculating nodes on Individual Node Features for ranking using some methods (e.g., Weighted Average Method, SVM)
Extracting Individual Node Features
Making recommendation
Target user
Friends of second degree
Friends of third degree
· · ·
Friends of mth degree
Friends social network of target user

Figure 1 (Color online) Architecture of Friend++.

individual features, we obtain

$$\begin{aligned} P_{i,j} &= \Pr\left(X_{u,t+1} = j \mid X_{u,t} = i\right) \\ &= \text{sigmoid}\left(\log\left(\#uv + 1.1\right) \times \log\left(\#vu + 1.1\right) + \gamma \text{RV}\left(u, v\right) + \mu \text{RV}\left(v, u\right)\right). \end{aligned} \tag{2}$$

We can obtain a state transition probability matrix P (and a multi-features transition probability matrix) according to formula (2). U denotes the set of all users in the recommendation system, n states the size of the user set, $\bar{P}^k_{i,j}$ stands for the column vector of matrix P, and $\bar{R}^*_{u,\_}$ denotes the row vector of matrix P and represents the rating value from user u to user m. According to the activity levels of users on the network, we can establish a dimensional information matrix between target users and candidate friends. We can then obtain the friendship probability between the target user and candidate friend j at time t.

$$\Pr\left(X_{u,t} = j\right) = \alpha \sum_{i=1}^{m} \Pr\left(X_{u,t-1} = i\right) P_{i,j} = \alpha^k \sum_{i=1}^{m} R^*_{u,i} P^k_{i,j} = \alpha^k \bar{R}^*_{u,\_} \bar{P}^k_{\_,j}. \tag{3}$$

According to formula (3), we can deduce the formula for calculating the total friendship probability between the target user and all candidate friends, as shown in

$$\Pr\left(X_u = j\right) = \frac{\sum_{k=1}^{x} \Pr\left(X_{u,k} = j\right)}{\sum_{k=1}^{x} \sum_{i=1}^{m} \Pr\left(X_{u,k} = i\right)} = c \sum_{k=1}^{x} \alpha^k \bar{R}^*_{u,\_} \bar{P}^k_{\_,j}. \tag{4}$$

According to formula (4), the overall sequencing matrix of the rating scores for all candidate friends is given in

$$\tilde{R} = \sum_{k=1}^{x} \alpha^k R P^k = R \alpha P \left(1 - \alpha P\right)^{-1}. \tag{5}$$

Because matrix $\tilde{R}$ reflects the target user's ratings of users, we can sequence the target user's user ratings according to this matrix; accordingly, we can receive the users' Top-N recommendations.

The random-walk-with-restart model is used for evaluating the similarity between points. Compared to other traditional collaborative filtering algorithms, it can more effectively solve problems caused by data sparseness, and is more efficient and accurate. In general, the algorithm starts from one point in a graph, and walks randomly along the edges of the graph. When it arrives at one point, it will choose the edge of the point using a certain probability, and continue to walk or arrive at the last point. For a non-periodical irreducible graph, after walking N times, each point's probability distribution will become stable. When they become stable, every point of the graph can be interpreted as the relation degree between the current point and the starting point.

The RWR algorithm calculates the similarity between the target point and point j according to

$$R_t = (1-c)\,\tilde{W}R_{t-1} + ce_j, \tag{6}$$

where $c \in (0,1)$ is the restart probability, $\tilde{W}$ is an adjacency matrix, e_j is an initial vector, and R_t is a probability distribution vector after t iterations.

The next key step involves (1) abstracting the relations between users, including fan relations, comment relations, retransmission relations, and @ relations between users on a target user-user bipartite graph, (2) establishing a grade mechanism among users, (3) introducing the random-walk-with-restart algorithm, and (4) individually recommending friends to a target user.

2.3 Instance of Friend++

To address Problem 1, we create a real instance of Friend++ by replacing two customizable parts with two novel node features and the weighted average method, respectively. More specifically, we (1) define and extract individual node features InterestDeg and InterestActivity, (2) use the weighted average method to calculate the ranking values of nodes, and (3) update the state transition probability matrix by combining individual features such as InterestDeg(u,v), InterestActivity, and Intimacy degree, which are set at different weights accordingly. Afterward, we integrate the weighted average method into the random walk framework for individual friend recommendation.

2.3.1 *Individual network feature*

In this subsection we describe Intimacy degree, an individual network feature introduced in [6], and utilize it to measure the degree of interaction between two users and to indicate how well two users communicate/chat with each other.

Definition 2 (Intimacy degree). This is an individual network feature that represents the total number of replies, forwards, comments made, "like" clicks, and follower-followee relationships with other users in the network. Formally, we have

$$\text{Intimacy degree}\,(u,v) = \log\left(\#uv + 1.1\right) \times \log\left(\#vu + 1.1\right), \tag{7}$$

where u or v is a user in G, $\#uv$ denotes the total number of replies, forwards, comments made, and "like" clicks that users u and v conduct with each other; thus, the value of $\#uv$ is a non-negative integer. In this formula, the symbol log represents a logarithm function based on 10 and is used for a feature scaling operation that ensures that the value of Intimacy degree is on a similar scale. In practice, we observed that the value of Intimacy degree varies drastically from dozens to tens of thousands, because $\#uv$ denotes the total number of @ operations that users u and v conduct with each other. This will cause the slow convergence of our algorithm if there is no feature scaling. Using the log function, we can place every value of Intimacy degree into a small approximate range that further promises the stable convergence of our RW-based algorithm.

The definition of Intimacy degree comes from our extensive practical experience. The value of 1.1 in the definition is an empirical one and determined using our prior knowledge. To ensure that the calculated result of the log function becomes a positive number, we must introduce a number greater than 1. Actually, using 1.2 or some other similar value has few effects on our recommendation results, because the value of $\#uv$ is often relatively large.

2.3.2 *Individual nodes features*

The individual node feature InterestDeg(u,v) describes the matching degree between user u's interests and user v's interests. It is measured from the target user's point of view, and computed using cosine similarity according

$$\text{InterestDeg}\,(u,v) = \frac{I_u \bigcap I_v}{I_u \bigcup I_v}. \tag{8}$$

I_u is an array of topic keywords describing user u's interests; $K = |I_u|$ represents the number of interest topics. Its details are given by

$$I_u = \text{LDA}(p(d_i|\alpha,\beta), D_{\text{new}}), \tag{9}$$

where $D\ (= \{d_i\}, i = 1,2,\ldots,M)$ denotes the training set consisting of all posts of the target user in the past N months, α represents the prior distribution of interest topics in the target user's posts, β denotes the prior distribution of words in interest topics, D_{new} represents a long post document generated by merging the text from all of user u's posts in the past N months, and $p(d_i|\alpha,\beta)$ displayed in formula (10) represents the learned parameters of the LDA model on training set D.

$$p(d_i|\alpha,\beta) = \int\int p(\phi|\beta)p(\theta_{d_i}|\alpha)\prod_{n=1}^{M} p(d_i|\phi^{z_{d_i,n}},\theta_{d_i})\mathrm{d}\phi\mathrm{d}\theta_{d_i}, \tag{10}$$

where θ represents probabilities of all interest topics in every post, ϕ stands for the distribution of interest topics represented by words in the vocabulary, θ_{d_i} denotes the distribution of interest topics in the posts of the target user, z_{d_i} denotes the interest topic of post d_i, and M indicates the size of set D.

The individual node feature InterestActivity is the activity index of the latest interest-specific posts posted by a candidate friend v of target user u. It is computed by

$$\text{InterestActivity}_u\,(v) = \sum_{\text{latest } N \text{ months},\ v\in C} \text{SoM}\,(M_n\,(v)) \times w\,(n)\,, \tag{11}$$

where C denotes the set of candidate friends of target user u, $M_n(v)$ denotes the set of v' s interest-specific posts in the Nth month, SoM($M_n(v)$) represents the total number of reviews, forwards, comments made, "like" clicks, and post replies in $M_n(v)$, $w(n)$ is the weight value of the nth month, and N indicates the number of months under consideration. It is important to note that this formula measures the properties of v rather than the relationship between u and v.

2.3.3 *Ranking nodes using weighted average method*

We calculate the node rankings using the weighted average method, which combines the user's InterestDeg and InterestActivity; the computations are performed using

$$\text{RV}\,(v,u) = d\_\text{weight} \times \text{InterestDeg}\,(v,u) + a\_\text{weight} \times \text{InterestActivity}_u\,(u)\,, \tag{12}$$

$$\text{RV}\,(u,v) = d\_\text{weight} \times \text{InterestDeg}\,(u,v) + a\_\text{weight} \times \text{InterestActivity}_u\,(v)\,, \tag{13}$$

where d_weight and a_weight are parameters describing the InterestDeg and InterestActivity between users.

2.4 Implementation of Friend++

2.4.1 *Building random walk graph*

In social networks, we believe the correlation between two users depends on the following three factors: (1) the followee relationships among users, (2) the friend relationships among users, and (3) the interaction relationships among users (including making comments, forwarding, and so on). To analyze these relationships, we propose a normalized relationship rating formula, as shown in

$$W\,(i,j) = \alpha \times [w\,(i,j) + w\,(j,i)] + \beta \times c\,(i,j)\,, \tag{14}$$

where α and β are coefficients, $\alpha + \beta = 1$, $\alpha \in (0, 1)$, $\beta \in (0, 1)$, $w(i, j)$ is user i's interactive value to user j, $w(j, i)$ is user j's interactive value to user i, and $c(i, j)$ is the number of mutual friends between i and j.

Owing to the dynamic nature of social networks, the interactive information of the user has time sensitivity. To measure the importance of a user's activities in developing his/her social network, we believe that recent activities should be weighted more heavily than less recent activities. Thus, we introduce an attenuation function to calculate the interactions of two users in social networks, as shown in

$$\phi\left(t_i\right) = \frac{k}{k + t_i}, \tag{15}$$

where k is a constant factor, and t_i is the number of days from the day of the interaction to the current day. An interactive index can be calculated by

$$w\left(i.j\right) = \sum_{t=1}^{N} m\left(t\right) \phi\left(t\right), \tag{16}$$

where $m(t)$ is the interaction time between i and j over the past five days, and $\phi(t)$ represents the time attenuation function. Using (14)–(16), we can obtain adjacency matrix $\tilde{W}$. Each element of the expression in adjacency matrix $\tilde{W}$ is expressed by

$$W_{i,j} = \frac{i}{k_j} \sum_{i=1}^{m} \frac{w\left(i, j\right)}{k_l}, \tag{17}$$

where k_j denotes the number of users who have interactions with the recommended user j, and k_l denotes the number of users who have interactions with the recommended user l.

2.4.2 *Random walk process*

There are four input parameters in a random walk process: (1) adjacency matrix $\tilde{W}$, (2) the initial probability distribution vector e_0, (3) the probability of restart c, and (4) the probability distribution vector p. In addition, r is the probability distribution vector after randomly walking each step. According to formula (2), the probability distribution vector r_0 can be initialized. Then r is calculated by

$$r = (1 - c)\, \tilde{W}^{\mathrm{T}} r_0 + cp. \tag{18}$$

According to formula (18), the calculated r is put into r_0 iteratively, and iterations are performed until the algorithm converges (normally, the algorithm will reach convergence within 50 iterations). We can then obtain the stable distribution probability vector r, which is the similarity vector of recommendation results.

This section describes the random-walk-with-restart algorithm. The main step in this algorithm is shown in Algorithm 1.

3 Experimental results

3.1 Data collection and observations

3.1.1 *Datasets*

To evaluate our method, we developed a Hadoop-based system to crawl Weibo social data, and distributively deployed it on the Ali cloud platform. The data we crawled and extracted consists of post forward time, forward content, user information, the numbers of times that users forwarded, commented on, or "liked" content, and so on. User information includes user ID, gender, location, number of fans, number of posts, etc. According to the different characteristics of Weibo data, we followed different crawling strategies to obtain three original datasets. In order to ensure the effectiveness of our evaluations, we

Algorithm 1 Random-walk-with-restart algorithm

```
1: $\tilde{W} \Leftarrow$ The normalized matrix;
2: $r_0 \Leftarrow$ the initial vector;
3: iterator $= 0.000001$;
4: $M \Leftarrow$ the state transition matrix;
5: Initialize $r_{(k)}^{(0)} = \left(r_{(1)}^{(0)}, r_{(2)}^{(0)}, \ldots, r_{(n)}^{(0)}\right)$ , $m$=1;
6: for $i = 1, 2, \ldots, N$ do
7:     $\phi_i^{(m-1)} = \frac{r_i^{(m-1)}}{\|r_i^{(m-1)}\|_1}$ ;
8:     $A^{(m-1)} \Leftarrow$ the aggregation matrix;
9:     $(A^{(m-1)})_{ij} = \phi_i^{m-1} M_{ij} e_j$;
10:    $\zeta^{(m-1))} \Leftarrow$ characteristic vector;
11:    $\zeta^{(m-1))} = \zeta^{(m-1)} A^{(m-1)} + c_{1\times N}$ (constant vector);
12:    if $r_{(k)}^{(m)} - r_{(k)}^{(m-1)}) \leqslant$ iterator then
13:        $r_{(k)}^{(m)} = (\zeta_1^{(m-1)} \phi_1^{(m)}, \zeta_2^{(m-1)} \phi_2^{(m)}, \times, \zeta_N^{(m-1)} \phi_N^{(m)})$;
14:    else
15:        $m \Leftarrow m + 1$;
16:    end if
17: end for
18: $r$ $\Leftarrow$Stationary distribution probability;
```

Table 2 Experimental setup

Dataset No.	Real size	Random selection	Alias	Characteristic
Dataset1	300000	160000	DS1	Dense network
Dataset2	165001	160000	DS2	High activities
Dataset3	212064	160000	DS3	High common interests

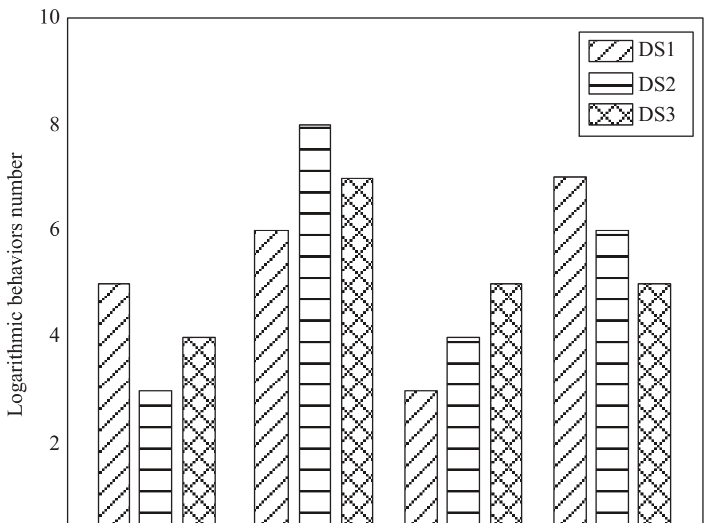


Figure 2 Results for user behavior statistics.

normalized these data and randomly selected 160000 users from every dataset as experimental users. We briefly describe these experimental datasets, referred to as DS1, DS2, and DS3, in Table 2. We compiled statistics on user behaviors in these datasets and the results are shown in Figure 2. The x-coordinate describes user behaviors that involve Weibo posts, clicking "like" forwarding, and making comments. The y-coordinate describes the logarithm that determines the number of user behaviors.

To gain insight into our experimental datasets, we abbreviated Intimacy degree as IN, InterestActivity as IA, and InterestDeg as ID. We then created the following statistical analyses on them in terms of IN, IA, and ID.

IN analysis. In Figure 3(a), the horizontal coordinate indicates the number of Weibo users; the unit is 10000, and the scale range is from 2 to 16 with an interval of 2. The y-coordinate describes the denary logarithm of SumIN, where SumIN is the total number of forwards, comments made, and "like" clicks. The maximum scale is $Y1$=8. To obtain $Y1$=8, we randomly selected $M = 10000$ users from DS2, and respectively counted the number of forwards ($fn = 657782$), the number of "like" clicks ($cn = 403081$), and the number of comments made ($mn = 290749$). Thus, the SumIN of each user is $(fn + cn + mn)$ $\div M \approx 135.2$, and the total sum of all users' SumIN in DS2 is SumIN$= 135.2 \times 160000 = 21632000$. Thus, lg(SumIN) ≈ 7.33, and $Y1$ equals $8 > 7.33$. From this figure, we can see that the level of IN is highest in DS1, followed by DS3 and DS2.

IA analysis. In Figure 3(b), the x-axis is the number of Weibo users, the unit is 10000, and the scale range is the same as Figure 3(a). The y-coordinate indicates the denary logarithm of SumIA, which is the total sum of the number of Weibo posts. The maximum scale is $Y2$=6. To compute it, we determined that the number of new daily Weibo posts can reach DNP $\approx$ 200 million and that the number of daily active users reaches DUA $\approx 49,800$ thousand. Therefore, the average number of Weibo posts per day of DNP users is ANW $\approx$ DNP $\div$ DUA ≈ 4.016, and the total number posts of 160000 users is TUP $\approx 64,2570.3$. Consequently, lg(SumIA) ≈ 5.8079, and $Y2$ equals $6 > 5.81$. From this figure, we can see that the level of IA is highest in DS2, followed by DS3 and DS1.

ID analysis. In Figure 3(c), the x-coordinate is the same as in both Figure 3(a) and (b). The y-coordinate describes the denary logarithm of SumID, which is the number of pairwise interest matches among N users ($2 \leqslant N \leqslant 160000$). The maximum scale is $Y3$=16. To obtain it, we first calculated SumID=$1+C_2^2 + C_3^2 + C_4^2 + L + C_N^2$, N=160,000. Then, SumID=$\sum_{N=2}^{160000} \frac{N(N-1))}{2} + 1$=7399023460993. Finally, lg(SumID) ≈ 12.869, and $Y3$ equals $16 > 12.869$. From this figure, we can see that the level of ID is highest in DS3, followed by DS2 and DS1.

Figure 3 highlights the following notable results:

• DS1 has the highest level of IN because it represents a dense social network (briefly, dense network) and owns the largest number of relationships/ties; meanwhile, its levels of IA and ID are relatively average because most users are linked to many other users.

• DS2 has the largest number of user activities (namely, a high level of activities), and thus it has the highest level of IA among the three datasets; in contrast, its levels of IN and ID respectively represent the lowest and middle values among the datasets. We surmise that if the IA factor is removed, the recommendation results may be somewhat influenced.

• DS3 has the highest level of common interests among Weibo users and therefore its ID is the highest. Moreover, both the IN and IA of DS3 are relatively average compared to DS1, DS2, and DS3. We believe that a higher ID indicates a higher probability of mutual communication between users.

3.1.2 *Observations*

Before proposing Friend++, we first investigate whether amicable chatting has some influence on friend recommendation, because a major motivation of our work is to discover the underlying factors that affect friend recommendation.

We define the term Interaction index to indicate how well two Weibo friends chat with each other. In general, the Interaction index refers to the total number of @ used by two friends to reply or to make comments to each other. To compute this index in detail, we first randomly select user u_1 as the target user from DS1, then select u_2 from DS2 and u_3 from DS3. We then compute Interaction indexes between the target user and all his/her friends according to

$$Y_i(u) = \omega_1 \times \mathrm{Som}(u, F_i) + \omega_2 \times (\mathrm{Som}(P(u, F_i)) + \mathrm{Som}(Q(u, F_i))), \tag{19}$$

where $u \in \{u_1, u_2, u_3\}$ are the target users, F is the set of the target user's friends, F_i is the ith friend of u, $Y_i(u)$ is the interaction index between u and the ith friend in F. Som(u, F_i) is the total number of interactive days between u and F_i in the past six months (i.e., for any day on which they had an interaction, the value is incremented by 1), Som$(P(u, F_i))$ is the total number of times that u and F_i

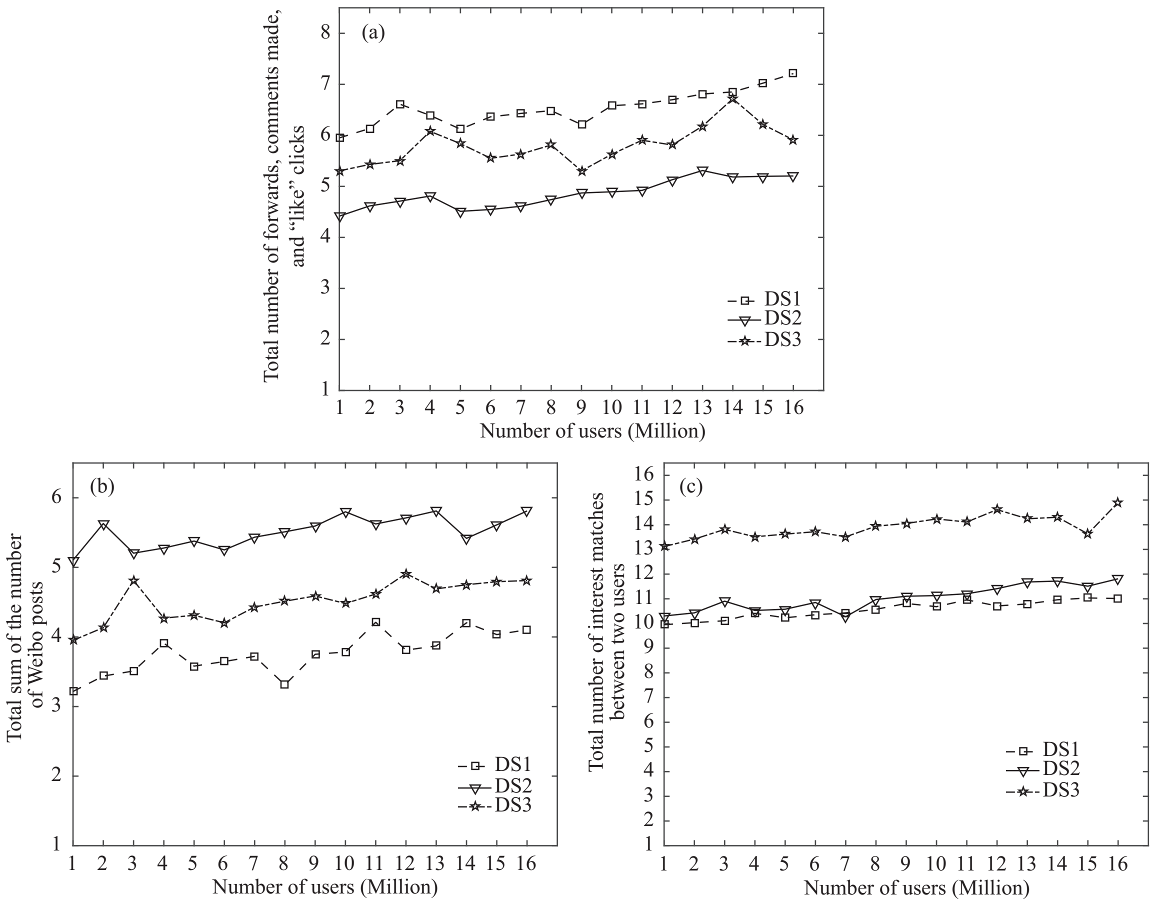


Figure 3 Analysis results for DS1, DS2, and DS3. (a) Results of IN analysis; (b) results of IA analysis; (c) results of ID analysis.

interacted on three consecutive days in the past six months, $\mathrm{Som}(Q(u, F_i))$ is the total number of days in the past six months on which more than 20 messages were exchanged between u and F_i (i.e., for each such day, the value is incremented by 1). Here, ω_1 takes 0.001, and ω_2 takes 0.1. We compute the $Y_i(t)$ for all users in F, then we obtain the set $C = \{C_i\}$ $(i = 1, 2, 3)$ in terms of $Y_i(u)$'s values ranked from large to small. According to Mark Zuckerberg, the founder of US social network Facebook, a person only has 50 people to contact; thus, we select the top-50 users in C_i, and further normalize the Interaction index using

$$\alpha = \frac{|F \bigcap C_{ij}|}{|F|}, \tag{20}$$

where $|F|$ is the number of friends the target user has, C_{ij} is the set of the jth user's friends in C_i, $|F \bigcap C_{ij}|$ is the number of common friends between the target user and the jth user in C_i, and α is the ratio of $|F \bigcap C_{ij}|$ to $|F|$. According to formula (20), we obtain values of α when the target user is u_1, u_2, or u_3; observation results are shown in Figure 4.

From Figure 4, we can observe that with a decrease in interaction, the value of α also decreases, which indicates that the number of common friends declines. Moreover, all analyses on the three random target users show similar changing trends. Actually, in real life, amicable chatting between two people can evidently show that they have similar interests or conversational preferences, and further motivate them to be good friends. In addition, an amicable chatting relationship is transitive in mathematics and can be passed from one person to another via an intermediary person. Consequently, we can strongly conclude that the amicable chatting factor in social networks has a significant influence on friend recommendation results.

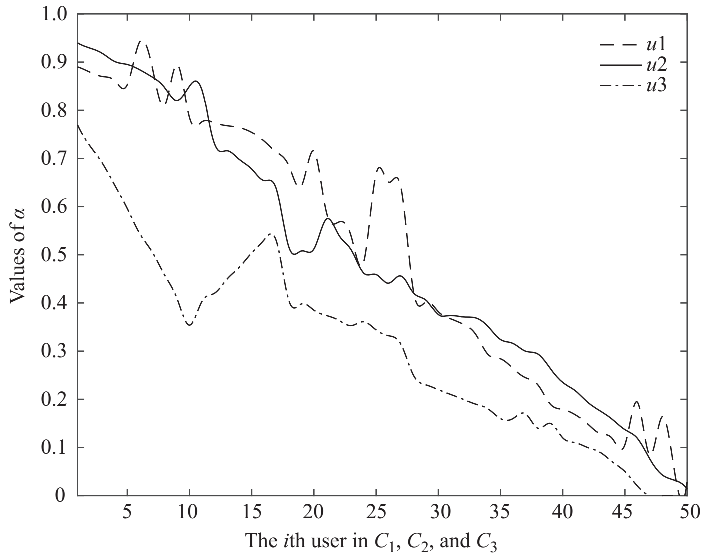


Figure 4 Trend of α for u_1, u_2, and u_3.

3.2 Evaluation metrics

In order to evaluate the performance of our method, we adopt the following metrics: P@k, MAP, Recall, and F1-Measure.

P@k. Indicates precision rates of the first k results that the system generates in response to inputted query keywords; it is defined as

$$\text{P@}k = \frac{\text{number of recommendation cases in the first } k \text{ results}}{k}. \tag{21}$$

MAP. Denotes the average accuracy value that corresponds to the query keywords of each user. Specifically, given an existing query keyword, the average value of precision rates will be computed according to the precision rates of the first k results. Namely, MAP is the average value of MAP in the entire testing dataset, where it is described as

$$\text{MAP} = \frac{\sum_{k \text{ is relavent}} \text{P@}k}{\text{date of recommendation cases}}. \tag{22}$$

Recall. Indicates recall rates of the first k results that the system generates in response to inputted query keywords; it is defined as

$$\text{Recall} = \frac{\text{number of recommendation cases in the first } k \text{ results}}{\text{the number of newly added friends}}. \tag{23}$$

F1-Measure. The weighted average of P@k and Recall; it is defined as

$$\text{F1-Measure} = \frac{2 \times \text{P@}k \times \text{Recall}}{\text{P@}k + \text{Recall}}. \tag{24}$$

3.3 Analysis of recommendation performance

3.3.1 *Recommendation performance*

As shown in formula (12), we utilize WAM as an instance to implement our proposed Friend++ architecture. Therefore, in this subsection, we will study how the weight a_weight (equivalently, d_weight) influences the performance of our method, because a_weight + d_weight = 1. We calculate the trends of MAP, Recall, and F1-Measure that occur with changes in weight a_weight from 0 to 1, on experimental datasets DS1, DS2, and DS3. The results are shown in Figure 5.

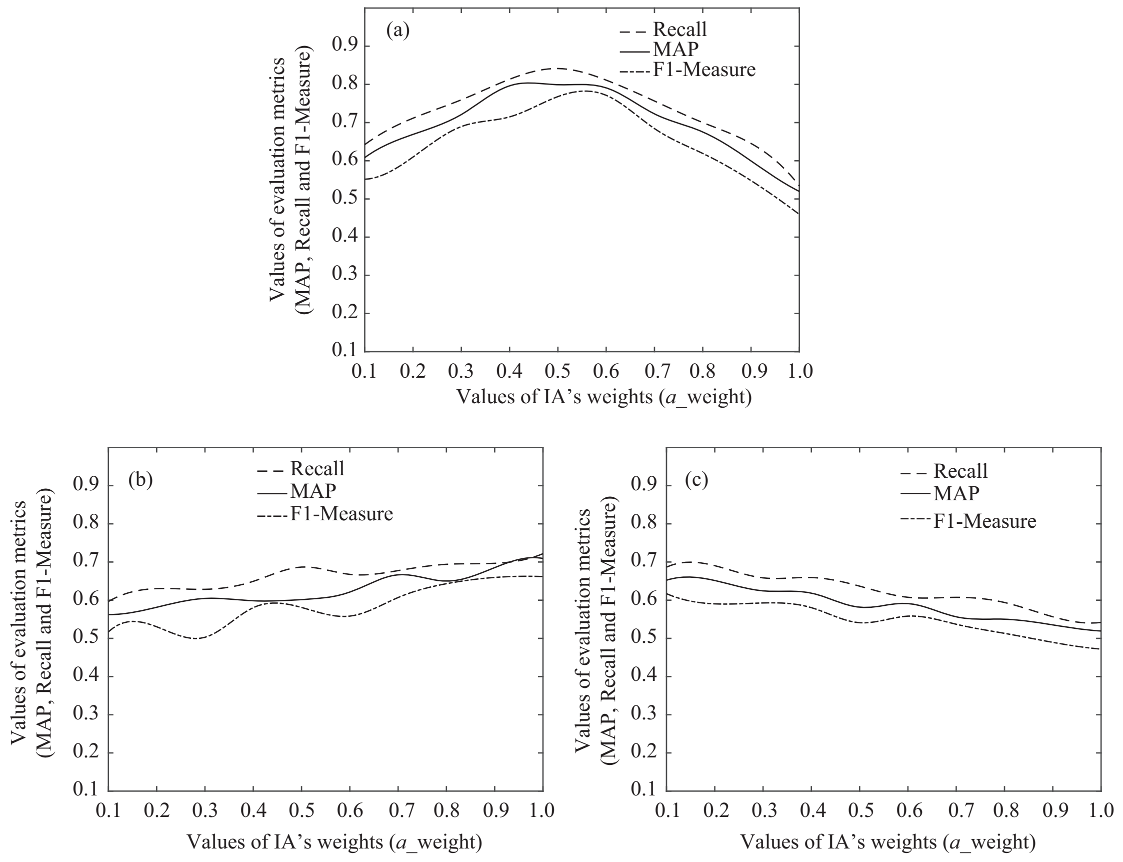


Figure 5 Trends of MAP, Recall, and F1-Measure with changes in weight a_weight in (a) DS1, (b) DS2, and (c) DS3.

From Figure 5(a), as the value of a_weight increases, the performance first increases and then decreases. Further, the performance is best when the a_weight ≈ 0.5. Formula (12) tells us that when a_weight (i.e., InterestActivity) increases, d_weight will decrease and cause the common interests factor to decrease to zero. Simultaneously, considering the DS1 characteristics mentioned above, these facts indicate that user activities do not have a significant influence on recommendation results, and that two other factors, interactions (i.e., Intimacy degree) and common interests (i.e., InterestDeg), may have far greater influence. Naturally, we set a_weight $= 0.5$ for the following experiments, because the performance is best in this case.

From Figure 5(b), as the value of a_weight increases, the performance always increases. Among the three datasets, DS2 has the largest number of activities, the intermediate number of common interests, and the smallest number of interactions. These facts tell us that the performance is not sensitive to weight a_weight even though there are enough user activities to be introduced into friend recommendation. On the other hand, because there is little data concerning common interests in DS2, weakening the InterestDeg factor has almost no effect on performance. This is further proof that user activities have limited influence on recommendation results.

From Figure 5(c), as the value of a_weight increases, the performance curve almost keeps falling. As described above, DS3 has the largest number of common interests, the intermediate number of activities, and the smallest number of interactions. Therefore, increasing a_weight from 0 to 1 will lead to a dramatic decrease in common interests. This can be explained by the fact that there is too much data concerning common interests in DS3. Another reason is that Weibo users with common/similar interests may be more likely to be friends. This means that common interests (i.e., InterestDeg) have a significant influence on performance. These are our preliminary conclusion regarding influence factors on recommendation results; we will analyze factor contribution in detail later.

According to the above results, we computed the values of MAP, Recall, and F1-Measure, and display

Table 3 Performance of Friend++

Dataset	MAP	Recall	F1-Measure
DS1	87.4	0.8327	0.8747
DS2	71.5	0.8186	0.5087
DS3	80.6	0.8121	0.7263

the results in Table 3. The table shows that our method performs best on DS1, followed by DS3 and DS2. However, in each case our method performed well, and thus it is reasonable to set a_weight=0.5.

3.3.2 *Comparison with baseline algorithms*

Here, we explain how our architecture Friend++ outperforms existing friend recommendation methods. To conduct our comparison, we selected random walk (RW) [6], LDA-based similarity (LDAS) [12], location-based recommendation (RWCFR) [13], common neighbors (CN) [14], the Jaccard index (Jaccard) [14], and the Adamic-Adar index (AA) [14] as baseline algorithms.

Random walk (RW) [6]**.** Introduces an individual network feature, Intimacy degree. In this approach, we first extract friend relationship data and related interaction activity data from social networks, and then compute Intimacy degrees between the target user and candidate friends. Next, we build an intimacy transition matrix and run the random walk algorithm (RW) based on a bipartite graph. Finally, after ranking the results, we obtain a Top-N recommendation list for the target user.

LDA-based similarity (LDAS) [12]**.** Makes recommendations by calculating the semantic similarity between two users. Specifically, we first extract nearly three months of Weibo post data, and then use this data to generate a high-dimensional topic vector via the latent Dirichlet allocation (LDA) method; we can then calculate the similarity of any two vectors. Finally, we rank the results and obtain a Top-N recommendation list for the target user.

Location-based recommendation (RWCFR) [13]**.** Constructs an undirected unweighted graph model that represents users, locations (check-in positions), and their relationships, in order to provide personalized recommendations. Instead of check-in data, we extract login position information from previously crawled user profile data. Check-in data is not involved because our social networks are online but not mobile. Subsequently, we build an undirected graph according to current user contexts such as local social relations, personal preferences, and login positions. Finally, we implement the algorithm, which is similar to RWCFR [13], in order to obtain a ranked recommendation list for the target user.

Common neighbors (CN) [14]**.** Finds and measures common friends between the target user and his or her second-degree friends. We first construct an undirected graph and then identify common neighbors within two hops. Finally, we obtain a recommendation list by measuring all sets of common friends.

Jaccard index (Jaccard) [14]**.** Can measure multiple types of similarities between target user x and candidate user y. According to [14], we consider the similarity between x and y from the perspective of common friends. We first obtain the set of candidate users, and then we obtain the number of friends of every candidate user. We finally count the number of common friends between the target user and the candidate user and further calculate the Jaccard index between x and y.

Adamic-Adar index (AA) [14]**.** Defines simple counting of common neighbors by assigning more weight to less-connected neighbors. We first obtain a set U of common friends between target user x and candidate friend user y. Then, we compute the number of friends $k(u_i)$ of user $u_i \in U$. Finally, the similarity AA between x and y is obtained by calculating the sum of the inverse of $k(u_i)$ $(i = 1, 2, \ldots, |U|)$.

We adopt six metrics (P@5, P@10, P@15, Recall, MAP, and F1-Measure) and utilize the three previously described datasets to compare the performance of our proposed Friend++ architecture against the following baseline algorithms: RW, LDAS, RWCFR, CN, Jaccard and AA. Table 4 lists the evaluation metric results of the different methods on these datasets. From this table, it is clear that Friend++ outperforms the six baseline algorithms. More specifically, our approach provides better recommendation precision than RW, LDAS, RWCFR, CN, Jaccard, and AA in terms of P@k. In terms of Recall, Friend++ achieves an average 24% improvement over RW, an average 27% improvement over LDAS, an average

Table 4 Comparative evaluation results

Dataset	Method	P@5	P@10	P@15	Recall	MAP	F1-Measure
DS1	Friend++	80.0	51.4	27.1	0.9327	87.4	0.8747
	RW	77.1	47.1	26.4	0.5938	85.5	0.5759
	LDAS	65.7	44.3	25.0	0.5129	73.4	0.2597
	RWCFR	39.1	32.6	25.4	0.2156	51.3	0.2146
	CN	79.6	51.2	27.8	0.9315	87.2	0.8663
	Jaccard	74.1	53.5	21.8	0.7624	79.2	0.7412
	AA	76.1	48.7	19.6	0.7845	74.6	0.6687
DS2	Friend++	71.4	44.3	24.3	0.9186	71.5	0.5087
	RW	65.0	40.0	22.5	0.8014	65.5	0.6736
	LDAS	51.4	38.6	22.9	0.6324	66.0	0.5222
	RWCFR	41.3	38.7	26.2	0.1835	60.8	0.1864
	CN	57.2	40.6	23.1	0.8124	68.1	0.5123
	Jaccard	62.3	43.5	19.8	0.7528	71.2	0.4658
	AA	54.8	37.9	15.4	0.7233	65.4	0.5315
DS3	Friend++	81.4	50.0	26.4	0.8721	80.6	0.7263
	RW	51.4	42.9	21.4	0.6036	63.1	0.6993
	LDAS	47.5	36.3	21.3	0.7623	54.1	0.3696
	RWCFR	28.3	20.4	15.8	0.2016	41.9	0.2423
	CN	54.6	46.2	22.7	0.7884	67.6	0.6543
	Jaccard	49.8	47.9	19.6	0.7249	69.5	0.6915
	AA	58.4	46.5	15.7	0.7461	64.5	0.5748

70% improvement over RWCFR, an average 6% improvement over CN, an average 16% improvement over Jaccard, and an average 15.6% improvement over AA on datasets DS1, DS2, and DS3. In term of MAP, Friend++ achieves an average 12.67% improvement over the six baseline algorithms. Meanwhile, the six compared methods underperform Friend++ by an average of 17.82% in terms of F1-Measure. All of these results prove that the performance improvements provided by Friend++ are statistically significant.

Also in Table 4, we can easily find two particular cases involving location-based RWCFR and common-neighbors-based CN, Jaccard, and AA baseline methods. We aim to highlight their performance in order to verify some of our hypotheses regarding social networks. After Friend++, CN exhibits the next-best performance on DS1 in that its accuracies are always close to the best attained accuracies. This result is consistent with our common sense, which says that the more common neighbors there are, the more interactions there will be among users. In addition, CN provides average performance on DS2 and DS3. The results clearly show that high levels of interest matching or user activities (i.e., posting a post or clicking "like") is not influenced by the number of common neighbors. In other words, more interest matching or user activities among users will not contribute to acquiring more common neighbors. Another baseline method, RWCFR, provides far worse overall performance than Friend++, while its scores are lower than those reported in [13]. Such undesired results occur because of the following: (1) check-in data in [13] allows better performance than the login data in our example social network, and (2) RWCFR is better suited to mobile social networks than to online social networks, because it is focused on mobile communications of mobile users. The last two baseline methods, Jaccard and AA, are based on common neighbors. As mentioned above, DS1 represents a dense online social network with relatively more edges among user nodes, which may naturally lead to a greater number of common neighbors. Hence, as with CN, both Jaccard and AA perform better on DS1 than on DS2 or DS3. Jaccard mainly depends on the number of common friends and the total number of friends a user has; thus, it performs worse than CN. AA performs the worst because it relies entirely on the number of friends for each user in common friends. Although active users may have more friends than inactive users, their interests may have some influence on increasing the number of friends; hence, in the friend recommendation process, interest matching degree is a less important factor than user interactions.

As shown in Table 5, both mean absolute error (MAE) and root mean square error (RMSE) metrics are

Table 5 Performance comparison based on MAE and RMSE using real datasets

Method	MAE	RMSE
Friend++	0.21768± (0.00248)	0.20743± (0.00267)
RW	0.25381	0.26865
LDAS	0.24543± (0.00372)	0.31542± (0.00198)
RWCFR	0.26452± (0.00312)	0.35315± (0.00287)
CN	0.23152± (0.00265)	0.25523± (0.00325)
Jaccard	0.24316± (0.00242)	0.26847± (0.00147)
AA	0.25982± (0.00226)	0.28648± (0.00184)

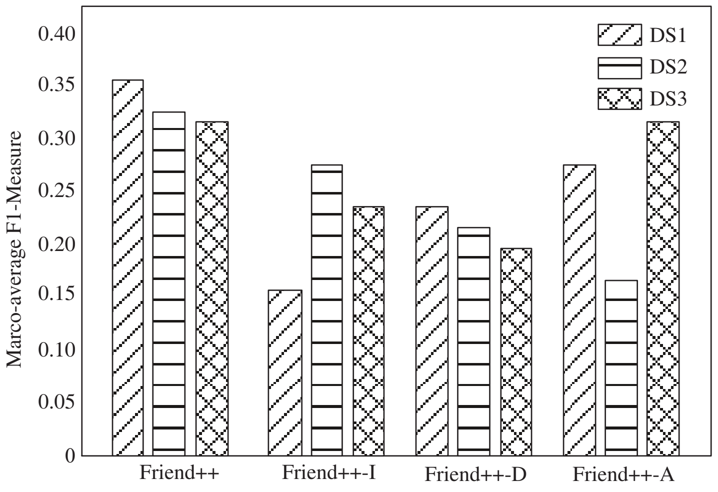


Figure 6 Values of Macro-average F1-Measure on DS1, DS2, and DS3.

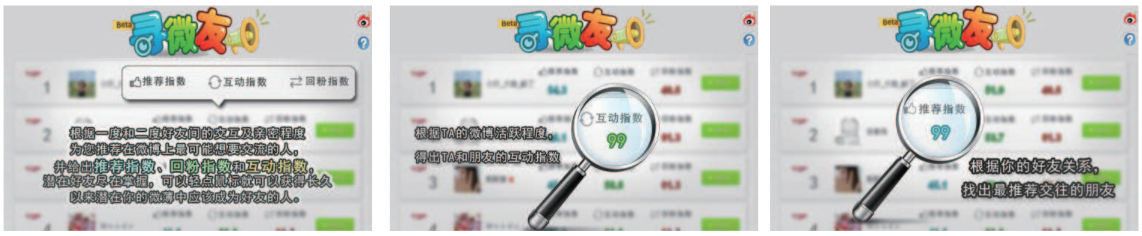
used to measure the recommendation quality of Friend++. Among the three methods based on common neighbors, we find that both the MAE and RMSE of CN are lower than those of Jaccard and AA, because the latter two are variants of the CN method and have lower overall performance than CN. However, the MAEs and RMSEs of these three methods are higher than those of Friend++. Further, it is clear that the proposed method steadily and consistently outperforms all six of the baseline methods. These results are consistent with our real experiences and common sense.

3.4 Factor contribution analysis

In Friend++, we consider three types of factor functions: Intimacy degree (I), interest matching degree (i.e., InterestDeg (D)), and the user activity index (i.e., InterestActivity (A)). We remove these factors one-by-one to show their impacts on recommendation performance. In particular, we first remove the InterestDeg factor function, denoted by Friend++-D, then we remove the InterestActivity factor function, denoted by Friend++-A, and finally we remove the Intimacy degree factor function, denoted by Friend++-I.

Figure 6 shows the Macro-average F1-Measure of different factor functions in DS1, DS2, and DS3, where Friend++-I excludes the Intimacy degree factor, Friend++-D excludes the InterestDeg factor, and Friend++-A excludes the InterestActivity factor. As discussed above, DS1 is a dataset with the highest IN, meaning that it has the most interactions. DS2 has the highest IA, which indicates the highest number of posting activities on Weibo. DS3 has the highest ID, which indicates the most matches of common interests between any two users. We observe a clear decrease on the Macro-average F1-Measure when ignoring some of the factors, which indicates that our method works well by combining the different factor functions. From this figure, we can also see that the value of the Macro-average F1-Measure of Friend++ is largest on DS1.

According to Figure 6, we further analyze the results on individual datasets. First, in DS1, there exists the highest number of user interactions, which results in the highest Intimacy degree (IN); thus, the



根据一度和二度好友间的交互及亲密程度为您推荐在微博上最可能想要交流的人，并给出推荐指数、回粉指数和互动指数，潜在好友尽在掌握，可以轻点鼠标就可以获得长久以来潜在你的微薄中应该成为好友的人。

根据TA的微博活跃程度，得出TA和朋友的互动指数

根据你的好友关系，找出最推荐交往的朋友

Figure 7 (Color online) Screenshots and descriptions of Xunweiyou, whose Chinese meaning is "Looking for Weibo friends".

performance is most negatively influenced and the F1-Measure value is reduced the most when the Intimacy degree factor function is removed, followed by removing InterestDeg and removing InterestActivity. Next, in DS2, there exists the most Weibo posting activities, which results in the highest InterestActivity (IA); thus, the value of the macro-average F1-Measure is most negatively affected and the y-coordinate value is reduced the most when the InterestActivity factor function is removed, followed by removing InterestDeg and removing InterestActivity. At last, in DS3, there exists the highest number of common interests among users, which leads to the highest InterestDeg (ID), and therefore the performance is most negatively influenced when the InterestDeg factor function is removed, followed by removing Intimacy degree and removing InterestActivity. It is clear that the value of Friend++-D is smaller than the other values, which indicates that users with common interests are more likely to be friends.

We then analyze the results across different datasets. When removing all three factors, on DS1 performance declined an average of 13% compared to Friend++; likewise, on DS2 performance declined an average of 11% and on DS3 performance declined an average of 7%. Simultaneously, as stated above, DS1 has the highest IN, DS2 has the highest IA, and DS3 has the highest ID. Consequently, we can intuit that among these three factors, IN has the most influence on recommendation results, ID is secondary, and IA has the least influence. On the other hand, these results indicate that, compared to user activities, common interests are a more important factor for friend recommendation, and that active Weibo users are more likely to be recommended compared to less active or inactive users. However, there is no doubt that amicable chatting is the most important factor. These analysis results prove that these three factors, Intimacy degree, InterestDeg, and InterestActivity, can effectively improve recommendation performance.

3.5 Case study

We developed a real Weibo application Xunweiyou as a case study to demonstrate how to recommend new Weibo friends to target users in our prediction task. Figure 7 shows that the application can provide Weibo users with a friend recommendation service that includes the following functions: (1) According to InterestDeg and InterestActivity, the Interaction index of every candidate friend is calculated and then provided to the target user. (2) According to Intimacy degree, the Recommendation index of every candidate friend is computed and then listed for the target user. (3) The target user will be provided with a list showing the candidate friends with whom they may want to chat, according to the target user's first- and second-degree friends.

Figure 8 shows that Xunweiyou can effectively solve practical problems and provide a target user with their desired recommendation service. Xunweiyou also offers some effectiveness metrics that users may want to review. From Figure 8, we can find that as the Interaction index increases, the Recommendation index also increases. This can be explained by the fact that interaction is the basis of the Intimacy degree factor; users who amicably chat with each other are more likely to be friends (and of course, no interactions, no friends).

According to tests conducted on Xunweiyou, in the best case, a candidate friend only uses 3 s to create a response after the friend request is sent by the target user. That is, the recommended friend accepts the request when the waiting time is only 3 s. In the worst case, the waiting time is 10 hours. This might occur in situations in which the recommended/candidate friend is not online. In addition, the follow-back ratio is approximately 97.73%, which indicates that our proposed architecture Friend++ is useful and

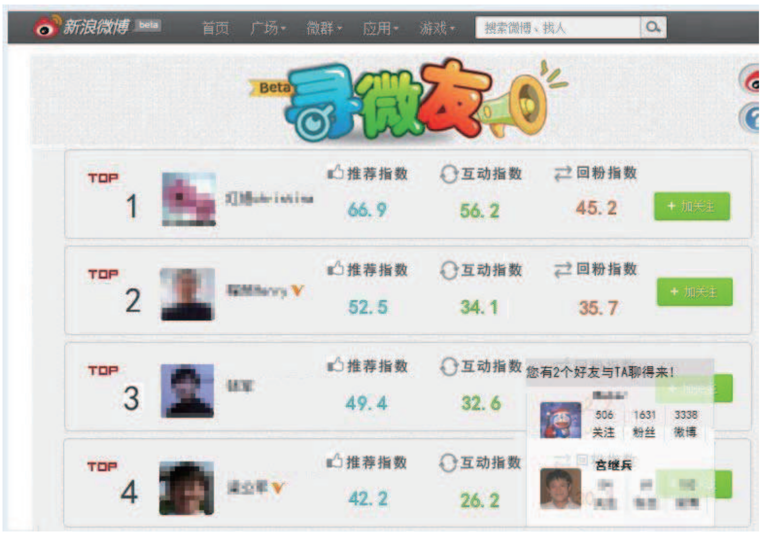


Figure 8 (Color online) Results of Xunweiyou application.

popular. We take a closer look at the results in Figure 8, and find that not only will the Top-N users in the recommendation list be recommended to the target user, but also their friends who are amicably chatting with the target user; this can provide a very fulfilling online social experience. Based on the results described above, the case study clearly demonstrates the usability and effectiveness of our method.

4 Discussion

In this section, we further discuss the architecture proposed in this paper, and provide an explanation of the previous analysis results.

(1) Our proposed method significantly outperforms the baseline methods. From Tables 4 and 5, we see that Friend++ achieves the best performance. This indicates that our proposed approach definitively improves the quality of friend recommendation.

(2) We see that by integrating the two individual node features (i.e., InterestDeg and InterestActivity) into random walk along with the network feature (i.e., Intimacy degree), we can significantly enhance the recommendation performance (+13%, +11%, and +7% respectively for Friend++-I, Friend++-D, and Friend++-A, in terms of Macro-average F1-Measure).

We have also observed an interesting result: the F1-Measure value of Friend++ is the highest on DS1, secondary on DS3, and the lowest on DS2. This is respectively consistent with the traits of the databases; i.e., DS1 has the highest IN, DS3 has the highest ID, and DS2 has the highest IA. Some reasonable explanations can help illustrate the results: for friend recommendation and other tasks in social networks, continuous active interactions are more important than common interests between any two users. If the network is not active, it will be very difficult for those users to become friends, even if they have the same interests. On the other hand, it is easier for users to become friends based on common interests rather than on posting activities. Similar to real life, two people may chat with each other, but if the content of their conversations does not overlap, they will be reluctant to communicate further, let alone be friends.

Further analysis shows that the Intimacy degree factor has the largest impact on recommendations, followed by the InterestDeg factor, while InterestActivity has the smallest impact on results; this is the case because Weibo posting activities are relatively less effective in attracting fans, which is consistent with our common sense and real experiences. In addition, although Friend++ outperforms the baseline algorithms, the method is not as effective in terms of metrics P@10 and P@15. We plan to continue our efforts to improve the performance of this method.

5 Related work

Numerous random-walk-based methods have been proposed for friend recommendation [6, 13, 15]. As a very important application of the PageRank algorithm [16], the random walk (RW) model is applicable to different domains [17]. For example, a four-step approach for topic-level random walk was proposed in [4]. The recommendation method employed in this study is similar to those in [4, 6, 15]. However, there are two main differences: (1) Our friend recommendation architecture takes into consideration that a replaceable ranking method (i.e., WAM) is integrated into the random walk framework to support universality. In contrast, other algorithms are based on a single consideration, and do not provide a data fusion model for personalized node features. (2) When considering what type of data to include for modeling, unlike most RW-based methods, our recommendation method makes extensive use of behavior context data to generate potential friend lists with maximal accuracy, specifically through defining two features: Intimacy degree and InterestActivity.

Among the works described above, Ref. [13] is not only a state-of-the-art friend recommendation method, but also so similar to our work that we must conduct some particular analyses to compare the two methods. Based on a previous work [15], Ref. [13] proposes RWCFR, a random-walk-based context-aware friend recommendation algorithm, which considers the current context (i.e., current social relations, personal preferences, and current location) of the user in order to provide personalized recommendations. Compared with RWCFR, Friend++ has three main differences: (1) In addition to social relations and personal preferences, Friend++ also includes behavior context data for modeling; in contrast, RWCFR uses check-in data to do so. (2) Friend++ is a hybrid recommendation architecture with two significant characteristics: "WAM plus random walk" and "network feature plus node features;" conversely, RWCFR is simply an algorithm based on a single consideration of the random walk model, and does not explicitly define network and node features for recommendation. (3) Friend++ constructs an undirected unweighted graph model that only includes users and their relationships, while RWCFR uses a better technique that includes users, locations, and their relationships.

Recently, an increasingly large number of researchers have been focusing on personalized friend recommendation (or individual friend recommendation) methods [18, 19] that employ some type of algorithm or model on social data related to locations (i.e., login and check-in positions) [13, 15], profiles [20], interests [6], or combinations of these [11], to obtain a ranked recommendation list for the target user. For example, Ref. [21] utilizes point of interest (POI) data to make personalized recommendations while referring to both friends' interests and registration information. Unlike most of the individual recommendation methods, our method provides a relatively more systematic recommendation service by considering that both network features and node features are extracted locally and employed in friend recommendation. Moreover, although these features have been investigated in many previous studies, we study them in relatively more depth: (1) We use more contextual behavior data to define the Intimacy degree network feature because behavior data can not only include interaction information of @, but can also provide user context information such as forwarded posts, reply posts, and comment content. (2) In terms of the node feature InterestDeg, most of the interest labels are generated from users' profile labels, post titles, frequent words in posts, and so on; however, our method applies the LDA model to produce semantic interest topics with higher accuracy. Moreover, we explain how to apply LDA to obtain them. (3) Although the node feature InterestActivity is slightly different from the others, it should be clearly noted that the user activities we choose are closely related to users' interest topics, and are generated in real-time; thus, they are more dynamic than others.

Computing similarities between users is also an important component in friend recommendation. Many methods can calculate similarities [18,22,23]; examples include common neighbors [14,24], Google PageRank [25], graph-based recommendation [26,27], behavior-based recommendation [28], Pearson correlation coefficient of collaborative filtering [29], content-based recommendation [27, 29, 30], and so on. For instance, Ref. [14] presents a link prediction method based on common neighbors for dynamic social networks. It considers the nodes within two hops to achieve better recommendation. As we know, the LDA model is often used in friend recommendation to increase its accuracy. For example, an extended

LDA model is applied for modeling interest topics of social users; thus, according to the generated user-topic probability distribution matrix, user similarities are calculated to make Top-N friend recommendations [31, 32]. However, the method proposed in this paper simply applies the LDA model to generate semantic interest topics, rather than computing the similarity between users.

Still other methods for friend recommendation are conducted through link prediction [33–35]. For example, Ref. [20] first proves that the simplest measure, namely common neighbors, provides better overall performance than the other eight link prediction methods (i.e., Jaccard index, hub promoted index, Salton index, preferential attachment, Adamic-Adar index, and so on). It then proposes a new similarity measure, which performs noticeably better than common neighbors, while requiring no additional information or computational time. Furthermore, it is found that many links obtain the same scores under local similarity measures, similar to the degeneracy of energy levels. It therefore proposes a new measure using the information of the next nearest neighbors, which can break the “degeneracy of states” and thus remarkably enhance the algorithmic accuracy. Finally, it outlines some future interests in this direction. This reference is very important to our work because it demonstrates both overall and systematic investigations into friend recommendations using simple rules, and motivates us to synthetically choose appropriate baseline algorithms (i.e., common neighbors) to more effectively evaluate Friend++.

6 Conclusion

In this study, we first design the hybrid multi-individual Friend++ in order to generate more accurate friend recommendations in social networks. In Friend++, through integrating WAM into RW, we succeed in seamlessly employing social ties, behavior context, and personal information to obtain a list of potential friends. We then implement an instance of Friend++ through inputting the parameters of WAM, Intimacy degree, InterestDeg, and InterestActivity. Finally, we select six baseline methods to verify our method on three real social datasets. Experimental results indicate that Friend++ achieves the best performance in terms of all evaluation metrics, which illustrates that the method has both universality and scalability. In addition, a case study demonstrates the usability and effectiveness of our method.

Through conducting analyses and discussion on experimental results, we have some meaningful conclusion: (1) Friend++ is more suitable for online social networks than for mobile social networks, (2) users with the same interests are more likely to be friends, (3) active users who post frequently are more likely to be recommended than inactive users, and (4) amicable chatting between users is the most important factor for friend recommendation. In other words, although common interests are an apparent factor for making recommendation, user interactions may be the most important factor in helping to find the most appropriate potential friends.

Acknowledgements This work was supported by National High Technology Research and Development Program of China (863 Program) (Grant No. 2015AA124102), Hebei Natural Science Foundation of China (Grant No. F2015203280), Open Project Program of the State Key Laboratory of Mathematical Engineering and Advanced Computing (Grant No. 214125), National Natural Science Foundation of China (Grant No. 61303130), Graduate Innovation Funded Program of Yanshan University (Grant No. 2017XJSS028), and Innovation Zone Project Program for Science and Technology of China’s National Defense (Grand No. 2017-0001-863015-0009).

Conflict of interest The authors declare that they have no conflict of interest.

References

1 Doina A D, Simone S. Using context to get novel recommendation in internet message streams. In: Proceedings of the 24th International Conference on World Wide Web. New York: ACM, 2015. 783–786

2 Cai Y, Leung H, Li Q, et al. Typicality-based collaborative filtering recommendation. IEEE Trans Knowl Data Eng, 2014, 26: 766–779

3 Erheng Z, Nathan L, Yue S, et al. Building discriminative user profiles for large-scale content recommendation. In: Proceedings of the the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2015. 2277–2286

4 Yang Z, Tang J, Zhang J, et al. Topic-level random walk through probabilistic model. In: Proceedings of Asia-Pacific Web Conference and Web-Age Information Management. Berlin: Springer-Verlag, 2009. 162–173

5 Wang C, Tang J, Sun J, et al. Dynamic social influence analysis through time-dependent factor graphs. In: Proceedings of the 2011 International Conference on Social Networks Analysis and Mining. New York: ACM, 2011. 239–246

6 Gong J B, Gao X X, Song Y Q, et al. Individual friends recomme dation based on random walk with restart in social networks. In: Proceedings of Chinese National Conference on Social Media Processing. Berlin: Springer-Verlag, 2016. 123–133

7 Oh H, Kim J, Kim S, et al. A probability-based trust prediction model using trust-message passing. In: Proceedings of the 22nd International Conference on World Wide Web. New York: ACM, 2013. 161–162

8 Wang P, Xu B W, Wu Y R, et al. Link prediction in social networks: the state-of-the-art. Sci China Inf Sci, 2017, 58: 011101

9 Paul S, Boutsidis C, Magdon-Ismail M, et al. Random projections for linear support vector machines. ACM Trans Knowl Discov Data, 2014, 8: 1–25

10 Chen W L, Chen Y X, Mao Y, et al. Density-based logistic regression. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2013. 140–148

11 Gong J B, Wang L, Sun S T, et al. iBole: a hybrid multi-layer architecture for doctor recommendation in medical social networks. J Comput Sci Technol, 2015, 30: 1073–1081

12 Tran V, Michael G. A spatial LDA model for discovering regional communities. In: Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. New York: ACM, 2013. 162–168

13 Hakan B, Pinar K. Context-aware friend recommendation for location based social networks using random walk. In: Proceedings of the 25th International Conference on World Wide Web. New York: ACM, 2016. 531–536

14 Yao L, Wang L N, Pan L, et al. Link prediction based on common-neighbors for dynamic social network. Procedia Comput Sci, 2016, 83: 82–89

15 Hakan B, Pinar K. Context-aware location recommendation by using a random walk-based approach. Knowl Inf Syst, 2016, 47: 241–260

16 Page L, Brin S, Motwani R, et al. The Pagerank Citation Ranking: Bringing Order to the Web. Technical Report 66. 1999

17 Li R-H, Yu J X, Huang X, et al. Random-walk domination in large graphs. In: Proceedings of IEEE 30th International Conference on Data Engineering, Chicago, 2014. 736–747

18 Chen H H, Jin H, Cui X L. Hybrid followee recommendation in microblogging systems. Sci China Inf Sci, 2017, 60: 012102

19 Tang J, Zhang J. A discriminative approach to topic-based citation recommendation. In: Proceedings of Pacific-Asia Conference on Knowledge Discovery and Data Mining. Berlin: Springer-Verlag, 2009. 572–579

20 Zhou T, Lü L Y, Zhang Y-C. Predicting missing links via local information. Eur Phys J, 2009, 71: 623–630

21 Zhang C Y, Liang H W, Wang K. Trip recommendation meets real-world constraints: POI availability, diversity, and traveling time uncertainty. ACM Trans Inf Syst, 2016, 35: 1–28

22 Sheng Y L, Jin R M. Learning personal social latent factor model for social recommendation. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2012. 1303–1311

23 Bisio F, Meda C, Zunino R, et al. Real-time monitoring of Twitter traffic by using semantic networks. In: Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. New York: ACM, 2013. 966–969

24 Yang X W, Harald S, Liu Y. Circle-based recommendation in online social networks. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2012. 1267–1275

25 Liu Q, Li Z G, Lui J C, et al. PowerWalk: scalable personalized pagerank via random walks with vertex-centric decomposition. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. New York: ACM, 2016. 195–204

26 Cheng H, Zhou Y, Yu J X. Clustering large attributed graphs: a balance between structural and attribute similarities. ACM Trans Knowl Discov Data, 2011, 5: 12

27 Dong Y X, Tang J, Wu S, et al. Link prediction and recommendation across heterogeneous social networks. In:

Proceedings of IEEE 12th International Conference on Data Mining. Washington: IEEE Computer Society, 2012. 181–190

28 Wu S-H, Chien H-H, Lin K-H, et al. Learning the consistent behavior of common users for target node prediction across social networks. In: Proceedings of the 31st International Conference on Machine Learning, Beijing, 2014. 32: 298–306

29 Benedikt L, Katja H, Jürgen Z. Blended recommending: integrating interactive information filtering and algorithmic recommender techniques. In: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. New York: ACM, 2015. 975–984

30 Chen W, Hsu W, Mong L. Modeling user's receptiveness over time for recommendation. In: Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval. New York: ACM, 2013. 373–382

31 Wang Z Y, Zhou Y, Tang J, et al. The prediction of venture capital co-investment based on structural balance theory. IEEE Trans Knowl Data Eng, 2016, 28: 1568–1569

32 Tang J, Jin R M, Zhang J. A topic modeling approach and its integration into the random walk framework for academic search. In: Proceedings of IEEE International Conference on Data Mining. New York: ACM, 2008. 1055–1060

33 Ye J H, Cheng H, Zhu Z, et al. Predicting positive and negative links in signed social networks by transfer learning. In: Proceedings of the 22nd International Conference on World Wide Web. New York: ACM, 2013. 1477–1488

34 Huang J B, Huangfu X J, Sun H, et al. Backward path growth for efficient mobile sequential recommendation. IEEE Trans Knowl Data Eng, 2015, 27: 46–60

35 Song D J, Meyer D A, Tao D C. Efficient latent link recommendation in signed networks. In: Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2015. 1105–1114