Expert Systems with Applications 41 (2014) 8075–8084

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Social network-based service recommendation with trust enhancement

Shuiguang Deng [a,b,*], Longtao Huang [a,b], Guandong Xu [c]

[a] College of Computer Science and Technology, Zhejiang University, China
[b] MIT Sloan School of Management, Massachusetts Institute of Technology, USA
[c] Advanced Analytics Institute, University of Technology Sydney, Australia

ARTICLE INFO

Article history:
Available online 17 July 2014

Keywords:
Social network
Service recommendation
Trust-enhanced
Random walk

ABSTRACT

Given the increasing applications of service computing and cloud computing, a large number of Web services are deployed on the Internet, triggering the research of Web service recommendation. Despite of service QoS, the use of user feedback is becoming the current trend in service recommendation. Likewise in traditional recommender systems, sparsity, cold-start and trustworthiness are major issues challenging service recommendation in adopting similarity-based approaches. Meanwhile, with the prevalence of social networks, nowadays people become active in interacting with various computers and users, resulting in a huge volume of data available, such as service information, user-service ratings, interaction logs, and user relationships. Therefore, how to incorporate the trust relationship in social networks with user feedback for service recommendation motivates this work. In this paper, we propose a social network-based service recommendation method with trust enhancement known as RelevantTrustWalker. First, a matrix factorization method is utilized to assess the degree of trust between users in social network. Next, an extended random walk algorithm is proposed to obtain recommendation results. To evaluate the accuracy of the algorithm, experiments on a real-world dataset are conducted and experimental results indicate that the quality of the recommendation and the speed of the method are improved compared with existing algorithms.

1. Introduction

Service computing, which provides flexible computing architectures to support the modern service industry, has emerged as a promising research area. (Deng, Huang, Li, & Yin, 2014a) With the prevalence of cloud computing, more and more modern services are being deployed in cloud infrastructures, providing rich functionalities. The numbers of services and service users are increasing dramatically (Deng, Huang, Tan, & Wu, 2014b; Deng, Wu, & Wu, 2013a). There has been an enormous explosion in data generation by these services, with a prevalence of user social networks, mobile devices, and large-scale service-oriented systems (Deng, Huang, Wu, & Xiong, 2013b). The overwhelming amount of service-generated data makes it more difficult for users to find appropriate services from a large number of candidate services to meet their functional and non-functional requirements. In most cases, users do not clearly know what they need or know about the existence of potential services on the Internet. As a result, service recommendation plays an increasingly more important role in helping users out of the service overload predicament and automatically recommending proper services for users (Deng, Huang, Wu, & Wu, 2014).

With the development of Web 2.0 in recent years, the Internet has transformed from a static platform for propagating and sharing information to a social interaction platform with greater participation and communication. Many famous online social networks, such as Myspace, Facebook, and Twitter, have sprung up. The dynamic and open Web service environment is quite similar to the social network environment: (1) service consumers can determine which objects to interact with by themselves, (2) the interaction information can be shared, (3) it is possible to collect and analyze the historic interaction records to make a decision about the interaction, (4) users can share information through trust delegation and recommendation, and (5) service participants have an obligation to provide recommendation information to others. Therefore, it becomes possible to adopt the research achievements in sociology in Web service research.

Driven by social networks, the participation of users has become more abundant and broader. Thus, descriptions of services not only include functional and non-functional attributes but are also enriched by users' feedback information, interaction histories, and users' social relationships. Such information can help to mine

* Corresponding author at: College of Computer Science and Technology, Zhejiang University, China. Tel.: +1 6177855155; fax: +1 6173241150.
E-mail addresses: dengsg@zju.edu.cn, dengsg@mit.edu (S. Deng).

http://dx.doi.org/10.1016/j.eswa.2014.07.012
0957-4174/

users' preferences on services, trust relationships between users, and users' influence on others to make personalized service recommendations more accurate and objective. Thus, utilizing social network information brings new opportunities to personalized service recommendations. Meanwhile, the emerging social networks also involve a large number of generated data with complex structures. The big data generated from these users and services are typically network based, which can reflect the relationship among users and services. Hence, understanding how to effectively utilize these big data to make service composition more accurate has become a significant challenge.

Currently, most studies on service recommendations mainly rely on the properties of individual Web services, i.e., Quality of Service (QoS) Shao, Zhang, & Wei, 2007; Chen, Feng, & Wu, 2011, but overlooking users' views on services. Thus, these methods may fail to capture users' personalized preferences accurately. With the help of social networks, users are more likely to share the feedbacks of services with their friends. Therefore, it is possible to make personalized recommendation by collecting users' feedback. Recommendation methods based on user feedback have been put into practice in many online commercial systems, such as Amazon, and Netflix. Collaborative filtering (Su & Khoshgoftaar, 2009), as a dominantly used approach in recommender systems, is being extended by researchers to Web service recommendations. However, these methods still suffer the same intrinsic shortcomings as in traditional recommender systems:

1.1. Sparsity problem

In addition to the extremely large volume of user-service rating data, only a certain amount of users usually rates a small fraction of the whole available services. As a result, the density of the available user feedback data is often less than 1% (Shao et al., 2007). Due to this data sparsity, collaborative filtering approaches suffer significant difficulties in identifying similar users or services via common similarity measures, e.g., cosine measure, in turn, deteriorating the recommendation performance.

1.2. Cold-start problem

Apart from sparsity, cold-start problem, e.g., users who have provided only little feedback or services have less been rated, or even new users or new services, is a more serious challenge in recommendation research. Because of the lack of user feedback, any similarity-based approaches cannot handle such cold-start problem.

1.3. Trustworthiness problem

With the increase of the number of services and users, some noisy information, or even the fake feedback data for malicious purposes may sneak into the inputs of recommendation systems. Apparently, such feedback will harm the prediction accuracy of the methods, thus should be differentiated and eliminated.

To solve the above problems, some research (Andersen, Borgs, & Chayes, 2008; O'Donovan & Smyth, 2005; Walter, Battiston, & Schweitzer, 2008) has proposed trust-enhanced recommendation methods. Instead of referring to the whole population's ratings on objects, such methods mainly consider the trusted friends' ratings to make predictions. The current trust-enhanced recommendation methods are based on 0/1 trust relationships – users who are trusted by the target user can be treated as neighbors equally in making recommendations. In reality, however, the contributions of users with different levels of trustworthy should be differentiated. Hence, we present a novel trust-enhanced service recommendation method based on social networks in this paper. We first introduce the concept of trust relevancy, which measures the trustworthiness weight of neighbors in social network. Then, we propose the algorithm RelevantTrustWalker, which uses random walk to make service recommendations in the weighted social network. Finally, we conduct experiments with a real-world dataset to evaluate the accuracy and efficiency of the proposed method. The experimental results validate the improvement of the proposed method.

The rest of this paper is organized as follows. Some related work is discussed in Section 2. The problem definition is presented in Section 3. We introduce our proposed approach in Section 4. We describe the experiments in Section 5. Finally, we conclude the paper and present some directions for future work in Section 6.

2. Related work

In recent years, service recommendation has become one of the hottest research topics in the field of service computing. Numerous studies have been performed on Web service recommendations (Chen, Liu, & Huang, 2010; Jiang, Liu, & Tang, 2011; Mu-Hsing, Chen, & Liang, 2009; Wang & Zeng, 2010). The early research on Web service recommendations mainly focused on users' functional requirements (Blake & Nowlan, 2007; Thio & Karunasekera, 2007). With the number of services with similar functionalities increasing, it has become more challenging to recommend services with better quality. Therefore, a heavily researched topic in Web service recommendation concentrates on QoS-based Web service recommendations that support optimized Web service selection by considering the QoS attributes of Web services with similar functionalities as well as preferences from users.

QoS-aware service recommendation approaches aim to predict the values of the QoS attributes of Web services and then make a final recommendation according to the predicted QoS values. Among the existing approaches, collaborative filtering has been widely adopted. Shao et al. proposed a user-based CF algorithm using PCC (Pearson Correlation Coefficient) to compute levels of similarity between users (Shao et al., 2007). Users who have similar historical QoS experiences on Web services are deemed similar. For any active user, the missing QoS values of a Web service can be predicted by considering the corresponding QoS values of services used by the users similar to him. Finally, Web services with high predicted QoS values are recommended. Zheng et al. proposed a novel hybrid collaborative filtering algorithm for Web service QoS prediction by systematically combining both item-based PCC (IPCC) and user-based PCC (UPCC) Deng et al., 2014b. The approach in Jiang et al. (2011) is based on the premise that if two users have similar QoS experiences for the same services, the two users are similar. However, this approach fails to solve the cold-start problem. If a user never has QoS experiences for any service, the method cannot recommend any services to him.

To improve the accuracy of recommendations for Web services, several new enhanced methods are proposed. Chen Xi, Liu Xudong, et al. recognized the influence of user location in Web service QoS prediction and proposed a novel method (Chen et al., 2010). The method groups users into a hierarchy of regions according to users' locations and their QoS records so that the users in a region are similar. When identifying similar users for a target user, instead of searching the entire set of users, the method only searches the regions to which the target user belongs. However, it does not consider service location in recommending Web services. Jiang proposed that the influence of the personalization of Web service items should be taken into account when computing the degree of similarity between users (Jiang et al., 2011). That is, more popular services or services with a stabler QoS from user to user should contribute less to user similarity measurements. Zhang suggested that it was better to combine users' QoS experiences,

an environment factor and a user input factor to predict Web service QoS values (Shao et al., 2007). However, the environment factor and user-input factor hardly represent users' subjective opinions.

In general, most of the current Web service recommendation approaches aim to predict the values of the QoS attributes of Web services and use historic invocations to extract users' interests or preferences while hardly considering the trust issue. As a consequence, they cannot be directly employed in a real Web service recommendation system to actively recommend services without users' interference (Maamar, Hacid, & Huhns, 2011). Users' feedback on services can reflect users' real opinions and preferences. With the development of social networks, it has become easier than ever before to obtain users' feedback. In fact, feedback-based recommendation approaches have been widely used in e-commerce applications; however, these approaches remain uncommon in the field of Web service recommendations.

Integrating trust in recommendations has become an enhancement for collaborative filtering-based approaches. The most common trust-enhanced recommendation approaches make users explicitly issue trust statements for other users. Golbeck proposed an extended-breadth first-search method in the trust network for prediction called TidalTrust (Golbeck, 2005). TidalTrust finds all raters with the shortest path distance from the given user and then aggregates their ratings, with the trust values between the given user and these raters as weights. MoleTrust is similar to TidalTrust (Massa & Avesani, 2007) but only considers the raters within the limit of a given maximum-depth. The maximum-depth is independent of any specific user and item. Jamali et al. proposed a random walk method called TrustWalker, which combines trust-based and item-based recommendations (Jamali & Ester, 2009). The random walk model can efficiently avoid the impact of noisy data by considering an adequate number of ratings. TrustWalker considers not only the ratings of the target item but also those of similar items. The probability of using the rating of a similar item instead of a rating for the target item increases with the length of the walk increases. Their framework contained both trust-based and item-based collaborative filtering recommendations as special cases. Their experiments showed that their method outperformed other existing memory-based approaches.

However, one of the limitations of the above work is that the recommendations from trusted friends are not absolutely suitable for the target user because they may have different tastes, preferences, or selection habits. For example, given user u_1 wants to see the movie *Transformers*, but he knows nothing about the movie. He may take into account his trusted friends' recommendations. Among his trusted friends, u_2 and u_3 have rated this movie as 4 and 2, respectively, and u_1 trusts u_2 and u_3 the same amount. However, u_1 and u_2 both like science fiction action movies, whereas u_3 prefers literary movies. Based on this information, there is a very high probability that u_1 will think that *Transformers* is a very good movie and worthy of watching. Thus, we think that the degree of trust of the given user's directed trusted friends should also be differentiated based on their preferences. Compared to the existing work, the main contributions of this paper are summarized as follows:

(1) We propose the concept of trust relevancy, which is used to weight the existing 0/1 social networks. Although the degree of trust in some trust-enhanced approaches is not 0 or 1, they should be initialized by users. However, in practical social websites, people can hardly define degrees of trust for their friends. Thus, these approaches are difficult to put into practice.
(2) We develop a random walk algorithm for recommendations based on the proposed trust relevancy. The target of each walk is selected according to the trust relevancy among users instead of being selected randomly. Thus, this algorithm can accelerate the convergence speed and improve the recommendation accuracy.
(3) We conduct a series of experiments with large-scale datasets to evaluate the accuracy and efficiency of the proposed method. The datasets are from the real-world website Epinions. The evaluation results show that the proposed method can be applied directly in existing social networks and provide high recommendation accuracy.

3. Problem definition and preliminary

Fig. 1 gives an example of the trust-enhanced service recommendation problem. The trust relations among users form a social network. Each user invokes several web services and rates them according to the interaction experiences. When a user needs recommendations, it predicts the ratings that the user might provide

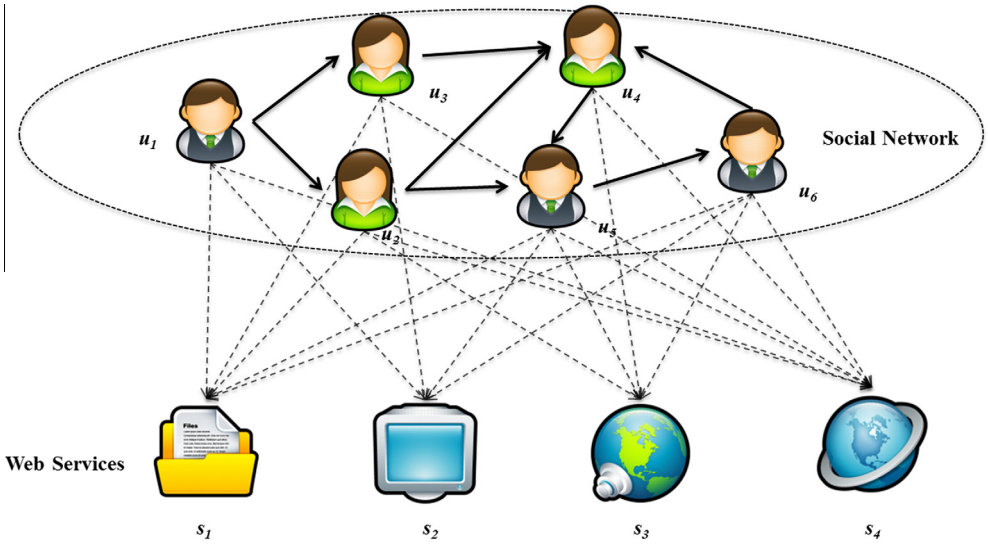


Fig. 1. An example of trust-enhanced service recommendations.

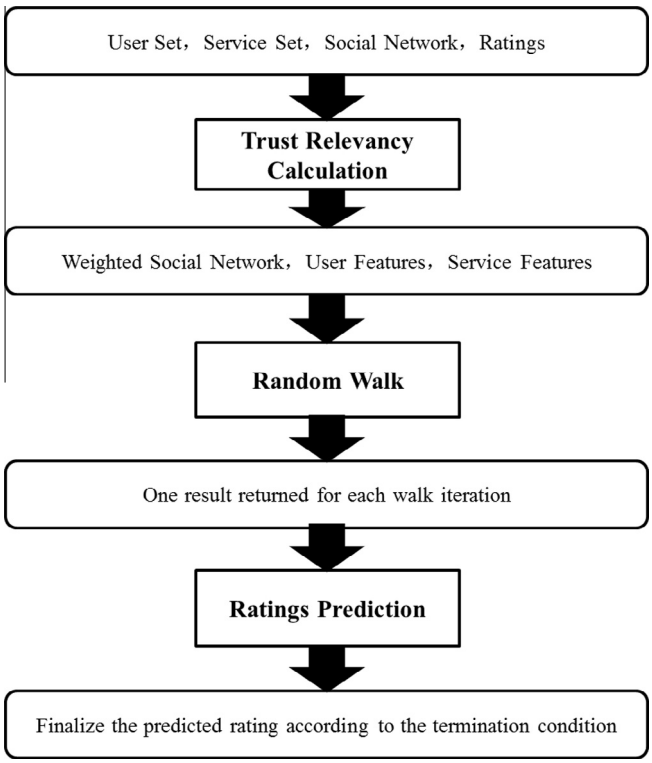


Fig. 2. Overview of the proposed method.

and then recommends services with high predicted ratings. Hence, the target of the recommender system predicts users' ratings on services by analyzing the social network and user-service rating records.

As stated in Jamali and Ester (2009), there is a set of users $U = \{u_1, u_2, ..., u_m\}$ and a set of services $S = \{s_1, s_2, ..., s_n\}$ in a trust-enhanced service recommender system. The ratings expressed by users on services are given in a rating matrix $R = [R_{u,s}]_{m\times n}$. In this matrix, $R_{u,s}$ denotes the rating of user u on service s. $R_{u,s}$ can be any real number, but often ratings are integers in the range of Deng et al. (2014a), Deng et al. (2014). In this paper, without loss of generality, we map the ratings $1, ..., 5$ to the interval $[0, 1]$ by normalizing the ratings. In a social rating network, each user u has a set S_u of direct neighbors, and $t_{u,v}$ denotes the value of social trust u has on v as a real number in $[0, 1]$. Zero means no trust, and one means full trust. Binary trust networks are the most common trust networks (Amazon, eBay, etc.). The trust values are given in a matrix $T = [T_{u,v}]_{m\times m}$. Non-zero elements $T_{u,v}$ in T denote the existence of a social relation from u to v. Note that T is asymmetric in general.

Thus, the task of a trust-enhanced service recommender system is as follows: Given a user $u_0 \in U$ and a service $s \in S$ for which $R_{u_0,s}$ is unknown, predict the rating for u_0 on service s using R and T.

4. Trust-enhanced recommendation approach

This section presents our approach in detail for trust-enhanced service recommendations. First, we define the concept of trust relevancy, on which our recommendation algorithm is based. Then, we introduce the algorithm RelevantTrustWalker by extending the random walk algorithm in Jamali and Ester (2009). Lastly, the predicted ratings are returned. The main process of our approach is shown in Fig. 2.

4.1. Trust relevancy

As stated earlier, the direct application of a trust relationship in a social network does not always increase the prediction accuracy. Services recommended from trusted users can only be considered reliable; such recommendations do not absolutely affect the target user's ratings because the target user and trusted users might differ in interests, preferences and perception. Therefore, we propose the concept of trust relevancy, which considers both the trust relations between users together with the similarities between users.

Definition 1 (*Trust Relevancy*). Given user u and v, the trust relevancy between u and v is as follows:

$$tr(u, v) = simU(u, v) * t(u, v),$$

where $simU(u, v)$ is the similarity of users u and v, and $t(u, v)$ is the degree of trust of u towards v. $t(u, v)$ can be explicitly provided by users or calculated based on some algorithms. Because most practical social networks are 0/1 trust networks, the degree of trust between two users usually equals 1. In this paper, we obtain the degree of trust directly from the Epinions dataset. If the degree of trust has not been explicitly provided by the dataset, we can calculate the values based on the historic interactions using some existing algorithms such as (Jøsang, Roslan, & Colin, 2007). By computing the trust relevancy between all connected users in a social network, we can obtain a weighted trust network (SN^+), where the weight of each edge is the value of trust relevancy.

A number of methods for calculating user similarities have been proposed (Benesty, Chen, & Huang, 2009; Danielsson, 1980; Krause, 1987). These methods basically calculate the distance between users' characteristic vectors. In service recommendation systems, the user-service rating matrix is usually very large. As a result, the dimensionality of the user/service vectors is extremely high. However, most of the score data are missing. Therefore, matrix factorization (MF) has been widely utilized in recommendation research to improve efficiency by dimension reduction (Koren, Bell, & Volinsky, 2009). MF can provide high extensibility as well as good prediction accuracy. For an m∗n users-service rating matrix R, the purpose of matrix factorization is to decompose R into two latent feature matrices of users and services with a lower dimensionality d. The calculation of the similarity of users in this paper first utilizes MF to obtain the latent feature matrix of users and then calculates the similarity distance between user vectors.

In this paper, we mainly aim to decompose the user-service rating matrix R. Based on the matrix factorization technique, matrix R can be approximately decomposed into two matrices P and Q:

$$R \approx PQ^T \tag{1}$$

where $P \in \mathbf{R}^{m\times d}$ and $Q \in \mathbf{R}^{n\times d}$ represent the latent feature matrices of users and services, respectively. Each line of the respective matrix represents a user or service latent feature vector. After decomposing the matrix, we use the cosine similarity measure (Ye, 2011) to calculate the similarity between two users. Given the latent feature vectors of two users u and v, their similarity calculation is as follows:

$$simU(u, v) = \cos(\mathbf{u}, \mathbf{v}) = \frac{\mathbf{u} \cdot \mathbf{v}}{|\mathbf{u}| \cdot |\mathbf{v}|} \tag{2}$$

where $\mathbf{u}$ and $\mathbf{v}$ are the latent feature vectors of the users u and v. The following example is used to clearly illustrate the process of calculating similarity.

Example 1. Given 6 users and 4 services, the user-service rating matrix is as follows:

$$R = \begin{bmatrix} 5 & 5 & 0 & 5 \\ 5 & 0 & 3 & 4 \\ 3 & 4 & 0 & 3 \\ 0 & 0 & 5 & 3 \\ 5 & 4 & 4 & 5 \\ 5 & 4 & 5 & 5 \end{bmatrix}$$

Through the MF technique, the rating matrix R is decomposed into two latent feature matrices, P and Q:

$$p = \begin{bmatrix} -0.4472 & 0.5373 \\ -0.3586 & -0.2461 \\ -0.2925 & 0.4033 \\ -0.2078 & -0.6700 \\ -0.5099 & -0.0597 \\ -0.5316 & -0.1887 \end{bmatrix} Q = \begin{bmatrix} -0.5710 & 0.228 \\ -0.4275 & 0.5172 \\ -0.3846 & -0.8246 \\ -0.5859 & -0.0532 \end{bmatrix}$$

We select the users $\mathbf{u_5}$ (−0.5099, −0.0597) and $\mathbf{u_6}$ (−0.5316, −0.1887). The similarity between the two users can be calculated as 0.9749. From the rating matrix, we can observe that user $\mathbf{u_5}$ and $\mathbf{u_6}$ only rate the third service differently. Therefore, their similarity should approach 1. It can be observed that the method accurately measures the similarity of two users.

4.2. Recommendation algorithm

This section introduces the proposed trust-enhanced random walk algorithm, RelevantTrustWalker, in detail. Table 1 gives the variables definitions used in the algorithm.

The RelevantTrustWalker algorithm attains a final result through multiple iterations. For each iteration, the random walk starts from the target user u_0 in the weighted trust network SN^+. In the kth step of the random walk in the trust network, the process will reach a certain node u. If user u has rated the to-be-recommended service s, then the rating of s from user u is directly used as the result for the iteration. Otherwise, the process has the following options:

(a) The random walk will stop at the current node u with a certain probability $\varphi_{u,s,k}$. Then, the service s_i is selected from RS_u based on the probability $F_u(s_i)$. The rating of s_i from u is the result for the iteration.

The probability that the random walk stops at user u in the k-th step is affected by the similarity of the services that u has rated and the to-be-recommended service s. The more similar the rated services and s, the greater the probability is to stop. Furthermore, a larger distance between the user u and the target user u_0 can introduce more noise into the prediction. Therefore, $\varphi_{u,s,k}$ should increase when k increases. Thus, the calculation for $\varphi_{u,s,k}$ is as follows:

$$\varphi_{u,s,k} = \max_{s_i \in RS_u} simS(s_i, s) \times \frac{1}{1 + e^{-\frac{k}{2}}}, \tag{3}$$

where $simS(s_i, s)$ is the similarity between the services s_i and s. The sigmoid function of k can provides value 1 for big values of k, and a small value for small values of k. Existing methods for calculating the similarity between two services are mostly based on collaborative filtering techniques. However, when two services do not have ratings from mutual users, such approaches cannot calculate their similarity values. Therefore, we propose calculating similarities between services based on the matrix factorization method. In Section 4.1, the matrix factorization of the rating matrix can be used to obtain not only the user latent features but the service latent features. Then, each service can be represented as a latent feature vector. Similar to formula (2), for two services s_i and s_j, the similarity calculation is as follows:

$$simS(s_i, s_j) = \cos(\mathbf{s_i}, \mathbf{s_j}) = \frac{\mathbf{s_i} \cdot \mathbf{s_j}}{|\mathbf{s_i}| \cdot |\mathbf{s_j}|} \tag{4}$$

When it is determined that user u is the terminating point of the walk, the method will need to select one service from RS_u. The rating of s_i from u is the outcome for the iteration. The probability of the chosen service $F_u(s_i)$ is calculated according to the following formula:

$$F_u(s_i) = \frac{simS(s, s_i)}{\sum_{s_j \in RS_u} simS(s, s_j)} \tag{5}$$

The way to select services based on $F_u(s_i)$ is through a roulette-wheel selection (Lipowski & Dorota, 2012), that is, services with higher values of $F_u(s_i)$ are more possible to be selected. Once the service s_i is selected, the rating of s_i from user u is returned as the result of the iteration. The random walk is likely to run and never stop. Therefore, we adopt the concept of "six degrees of separation" (Milgram, 1967). The maximum step for a random walk is set to 6.

(b) By contrast, the walk can continue with a probability of $1 - \varphi_{u,s,k}$. A user is selected from the set of users whom the current user u directly trusts as the target node for the next step.

At present, the existing methods usually choose the target node randomly, which means each user whom u trusts has an equal chance of being selected. However, as mentioned above, these users have different reference significance for user u. To distinguish different users' contribution to the recommendation prediction, we propose that the target node v for the next step from the current user u is selected according to the following probability:

$$E_u(v) = \frac{tr(u, v)}{\sum_{x \in TU_u} tr(u, x)}, \tag{6}$$

where $tr(u, v)$ is the trust relevancy introduced in Section 4.1. The trust relevancy guarantees that each step of the walk will choose the user that is more similar to the current user, making the recommendation more accurate.

The pseudo-code of the RelevantTrustWalker algorithm is as follows:

Table 1
Variables for RelevantTrustWalker.

Variables	Description
$r_{u,s}$	The real rating of service s from user u
$p_{u,s}$	The predicted rating of service s from user u
$\varphi_{u,s,k}$	The probability that the random walk stops at user u in the kth step
TU_u	The set of users that user u trusts
$E_u(v)$	The probability that user v is selected from TU_u as the target of the next step
$simS(s_i, s_j)$	The similarity of services s_i and s_j
RS_u	The set of services that user u has rated
$F_u(s_i)$	The probability that s_i is selected from RS_u to obtain u's rating for s_i

Algorithm 1. RelevantTrustWalker

Input: U(user set), S(service set), R(rating matrix), SN^+(weighted social network), u_0(the target user), s(to-be-recommended service)
Output: r (predicted rating)

```
1   set k = 1; //the step of the walk
2   set u = u0; //set the start point of the walk as u0
3   set max-depth = 6; //the max step of the walk
4   set r = 0;
5   while (k <= max-depth){
6     u = selectUser(u); // select v from TU_u as the target of
    the next step according to the probability E_u(v)
7     if (u has rated s){
8       r = r_{u,s};
9       return r;
10      }
11      else{
12        if (random(0,1) < φ_{u,s,k} || k == max-depth){
        //stop at the current node
13          s_i = selectService(u); // service s_i is selected from
    RS_u according to the probability F_u(s_i)
14          r = r_{u,si};
15          return r;
16        }
17        else
18          k++;
19      }
20    }
21    return r;
```

Example 2. Fig. 3 shows an example to illustrate the Algorithm 1 clearly. The weight of each edge represents the probability $E_u(v)$ which is calculated according to the Eq. (6). Suppose the service s_3 is to be recommended for the user u_1. For the first step of the walk (shown in Fig. 3 (a)), u_2 is more likely to be selected as the target node since the value of $E_u(u_2)$ is larger. If u_2 has rated s_3 with the rating r, r will be returned as the result of this walk (Line.7–9). Otherwise, if the termination condition (Line.12) is not reached, the walk would continue. For the second step, u_5 is selected (shown in Fig. 3 (b)). It should also check whether u_5 has rated s_3. If u_5 has not rated s_3 but the termination condition is reached, it will select the most similar service to s_3 from the services u_5 has rated (Line.13). The selection probability of each service is according to the Eq. (5). Then, the rating of the selected service by u_5 is returned as the result of this walk.

4.3. Ratings prediction

The RelevantTrustWalker algorithm attains a final result through multiple iterations. The final predicted rating is obtained by polymerizing the results returned from every iteration:

$$p_{u_0,s} = \frac{1}{n}\sum_{i=1}^{n} r_i \tag{7}$$

where r_i is the result of each iteration, n is the number of iterations. To obtain a stable predict result, the algorithm needs to perform an adequate number of random walks. We can decide the termination condition of the algorithm through the calculation of the variance of the prediction values. The variance of the prediction results after a random walk is denoted as σ^2. The calculation of σ^2 is as follows:

$$\sigma_i^2 = \frac{1}{i}\sum_{j=1}^{i}(r_j - \bar{r})^2 \tag{8}$$

where r_j is the result of every iteration, i is the total number of iterations until the current walk, and σ_i^2 is the variance obtained from the last i iterations, which will eventually tend to a stable value. In this paper, when $|\sigma_{i+1}^2 - \sigma_i^2| \leqslant \varepsilon$, the algorithm terminates ($\varepsilon = 0.0001$).

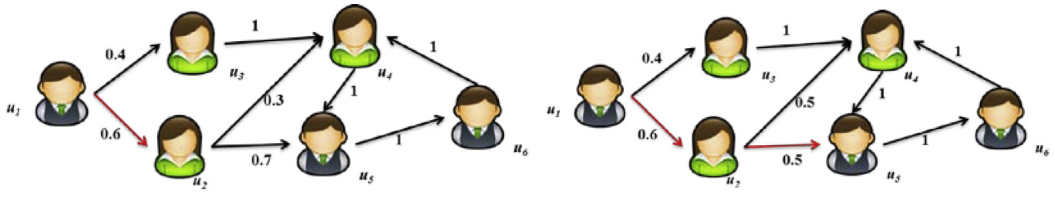
5. Evaluation and analysis

In this section, we conduct several experiments to compare the recommendation quality of the RelevantTrustWalker approach with other state-of-the-art trust-enhanced service recommendation methods.

5.1. Experiment setup

In the trust-enhanced recommender research domain, there is not much publicly available and suitable test data (Milgram, 1967). The Epinions dataset is considered the only publicly available social rating network dataset. Hence, in this paper, we choose Epinions as the data source for our experiments on trust-enhanced recommendations. Epinions is a well-known product review website that was established in 1999 (www.epinion.com). Users can rate products from 1 to 5 and submit their personal reviews. These ratings and reviews influence other customers when they make decisions on whether to buy a product. In addition, users are also allowed to specify whom to trust and build a social trust network. In the social trust network, reviews and ratings from a user are consistently found to be valuable by his trustees. Furthermore, products online can also be regarded as services, and several researches on services recommendation have utilized the datasets from Epinions (Alnemr, Bross, & Meinel, 2009; Noor & Sheng, 2014). Thus, Epinions is an ideal source for our experiments.

We use the dataset of Epinions published by the authors of Massa and Avesani (2006). Table 2 shows the statistics of the data source. It consists of 49,290 users, who rated a total of 139,738 different items at least once. The total number of ratings is 664,824. The density of the user-item rating matrix is less than 0.01%. Each user has rated 13.4 items on average. As to the social trust network, the total number of issued trust statements is 487,181. Each user has 9.9 direct trusters on average. We can observe that the user-item rating matrix of Epinions is incredibly large in size and extremely sparse, much sparser than another commonly used



(a) 1st step (b) 2nd step

Fig. 3. Example of RelevantTrustWalker.

Table 2
Statistics of the Epinions dataset.

Users	49,290
Items	139,738
Ratings	664,824
Trust Relations	487,181
Avg. ratings per user	13.4
Max ratings per user	1845
Avg. ratings per item	4.8
Max ratings per item	6843
Avg. trusters per user	9.9
Max trusters per user	1587
Avg. trustees per user	9.9
Max trustees per user	2,365

collaborative filtering dataset, Movielens, with a density of 4.25%. Thus, the dataset can evaluate the performance of our approach with big data and high sparsity at the same time. (Massa & Avesani, 2007) found that those with a service number of less than five comments can be identified as cold-start users. Cold-start users and relevant data accounted for more than 50%. Therefore, it is very important to consider cold-start users to enhance the accuracy of recommendations and coverage.

To evaluate the performance improvement of RelevantTrustWalker, we compare our method with the following state-of-the-art methods:

- *UserCF*: This method is the classic user-based collaborative filtering method, which makes prediction merely based on user similarity (Su & Khoshgoftaar, 2009).
- *ItemCF*: This method is the classic item-based collaborative filtering method, which captures similar item characteristics to make predictions (Nilashi, Othman, & Norafida, 2014).
- *TidalTrust*: This method is proposed by Golbeck (Golbeck, 2005) and generates ratings based on a trust inference algorithm.
- *MoleTrust*: This method is proposed by Massa and Avesani (Massa & Avesani, 2007) and predicts the trust score of the source user for the target user by walking through the social network starting from the source user and propagating trust along trust edges.
- *TrustWalker*: This method is a random walk method based on trust and item similarity (Jamali & Ester, 2009).
- *CoTrustWalker*: This method is an extension of *TrustWalker* and utilizes a cloud model to compute item similarity (Zhu, Xu, & Liu, 2013).
- *CliquesWalker*: This method is a random walk method that considers social cliques (Yang & Sun, 2012).

5.2. Evaluation metrics

Our experiments are developed using Java, and the experimental setting is an Intel Core2 P7370 2.0GHZ with 4 GB RAM machine with Windows 7 and jdk 6.0.

Leave-One-Out (LOO) is widely used for the evaluation of recommendation methods (Golbeck, 2005; Jamali & Ester, 2009). In this paper, we adopt LOO to evaluate the recommendation algorithms by hiding an actual rating value and predicting its value through the compared recommendation algorithms. Then, we compare the accuracy of the algorithms by analyzing the root mean square error.

We adopt the Root Mean Squared Error (RMSE), which is widely used in recommendation research (Blake & Nowlan, 2007; Thio & Karunasekera, 2007), to measure the error in recommendations:

$$RMSE = \sqrt{\frac{\sum_{u,s}(R_{u,s} - \widehat{R}_{u,s})^2}{N}}, \tag{9}$$

where $R_{u,s}$ is the actual rating the user u gave to the service s and $\widehat{R}_{u,s}$is the predicted rating the user u gave to the service s. N denotes the number of tested ratings. The smaller the value of RMSE is, the more precisely the recommendation algorithm performs.

Some recommendation mechanisms may not be able to predict all the ratings in test data given the high sparsity of the dataset. Involving trust can enhance coverage without sacrificing the precision. Therefore, we use the metric *coverage* to measure the percentage of pairs of <user, service>, for which a predicted value can be generated. For a pair of <user, service>, if the recommender cannot find a prediction on the rating for the reason of lack of enough information to calculate similariy, then the recommender cannot cover this pair of <user, service>.

$$coverage = \frac{S}{N}, \tag{10}$$

where S denotes the number of predicted ratings and N denotes the number of tested ratings.

To combine RMSE and *coverage* into a single evaluation metric, we compute the F-Measure. We have to convert RMSE into a precision metric in the range of [0,1]. The precision is denoted as follows:

$$precision = 1 - \frac{RMSE}{4}. \tag{11}$$

In this Eq. (4) is the maximum possible error because the values of the ratings are in the range of [1,5].

$$F - Measure = \frac{2 \times precision \times coverage}{precision + coverage}. \tag{12}$$

5.3. Experimental results

This section presents the results of a different experimental comparison including the recommendation results of all users and cold-start users and a performance comparison of the different algorithms.

From the experimental results in Table 3 and Fig. 4, we can observe that the accuracy of all the algorithms improves when recommending for all users. This finding is mainly because the user-service rating information is more adequate. From Table 3, we can see that the trust-enhanced recommendation algorithms have no absolute advantages in terms of accuracy compared to the User-based CF and the Item-based CF recommendation algorithms because the traditional collaborative filtering methods can calculate user/service similarity with adequate rating information. Several trust-enhanced recommendation algorithms only consider the trust between users, without considering the interests and preferences between users, which would result in the improvement of coverage without sacrificing accuracy. By contrast, the proposed RelevantTrustWalker considers trust relevancy, which combines the trust degree and user similarity, to improve the accuracy of the recommendation. Thus, the recommendation performance of

Table 3
Comparison results for all users.

Algorithms	RMSE	Coverage (%)	F-measure
User-based CF	1.141	70.43	0.7095
Item-based CF	1.345	67.58	0.6697
TidalTrust	1.127	84.15	0.7750
MoleTrust	1.164	86.47	0.7791
TrustWalker	1.089	95.13	0.8246
CoTrustWalker	1.074	94.23	0.8236
CliquesWalker	1.045	96.78	0.8379
RelevantTrustWalker	1.012	98.21	0.8486

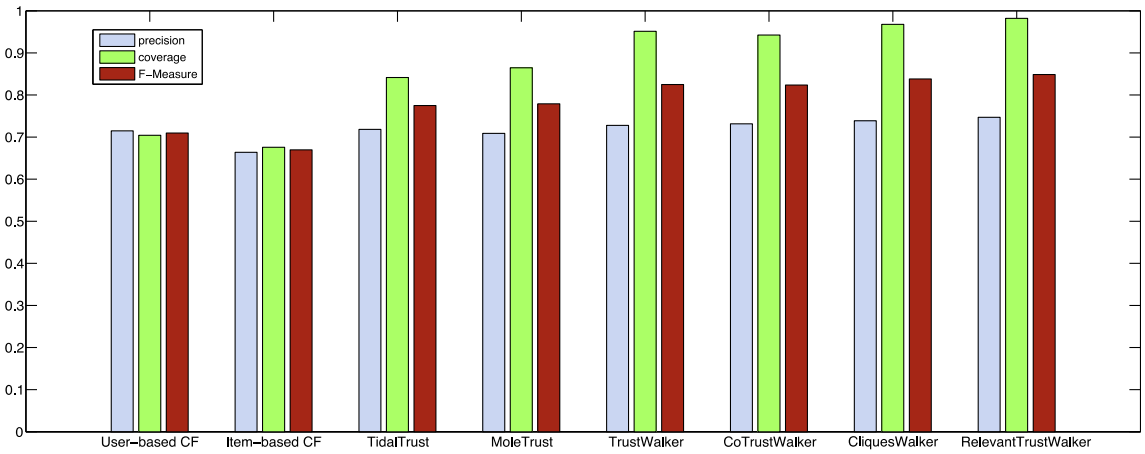


Fig. 4. Comparison results for all users.

RelevantTrustWalker is also superior to other algorithms when recommending for all users.

RelevantTrustWalker and other trust-enhanced recommendation algorithms offer the advantage of being able to solve the cold-start problem. It is necessary to assess their performance on recommendations for cold-start users. Table 4 and Fig. 5 show the comparison results for cold-start users.

From the above experimental results, we can observe that RelevantTrustWalker has a lower error than other recommendation algorithms for cold starts. Two traditional collaborative filtering algorithms, User-based CF and Item-based CF, perform the worst, mainly because the cold-start users rated few mutual services, which mean most user/service similarities cannot be calculated accurately. As a result, these two methods perform with the highest RMSE and the lowest coverage compared to other methods. Due to the introduction of the trust relationship between users, TidalTrust and MoleTrust can make service recommendations for users without mutual-rated services by utilizing trust relations. Therefore, the coverage improved greatly compared to the previous two algorithms, but the accuracy is not improved significantly. TrustWalker CoTrustWalker, and CliquesWalker have obvious improvements on both accuracy and coverage. Compared to these three random walk methods, RelevantTrustWalker has a lower RMSE because, at each step of the walk, RelevantTrustWalker chooses to trust more relevant users, rather than random selections, which justifies that not all recommendations from trusted users have equivalent reference values. In addition, the coverage of RelevantTrustWalker is also the highest because, when computing the similarity, we adopt the method of matrix factorization to obtain a service feature vector, which avoids the typical failures of similarity calculations in the absence of mutual-rated services. All in all, from the point of F-Measure, RelevantTrustWalker outperforms the compared algorithms for the cold-start users.

Table 4
Comparison results for cold-start users.

Algorithms	RMSE	Coverage (%)	F-measure
User-based CF	1.485	18.93	0.2910
Item-based CF	1.537	23.14	0.3364
TidalTrust	1.238	60.75	0.6463
MoleTrust	1.397	58.29	0.6150
TrustWalker	1.212	74.36	0.7195
CoTrustWalker	1.204	74.85	0.7229
CliquesWalker	1.187	76.78	0.7341
RelevantTrustWalker	1.143	79.64	0.7531

Because the size of the rating data is extremely large, the time cost of service recommendation is also an important evaluation indicator. Fig. 6 shows the average time cost of different algorithms. We can observe that the User-based CF and Item-based CF cost the least in terms of time, primarily because they only consider the rating relationship between users and services as the algorithm input. By contrast, the other algorithms also add

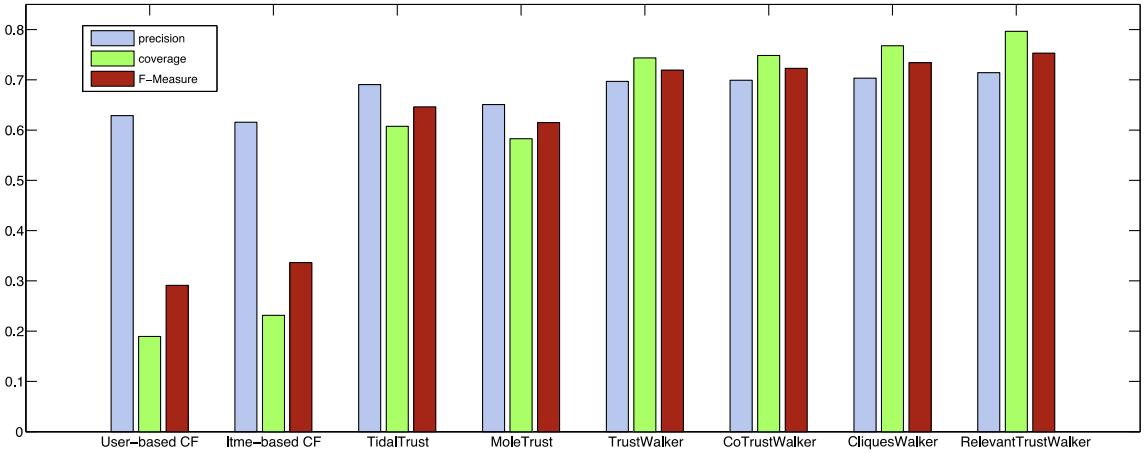


Fig. 5. Comparison results for cold-start users.

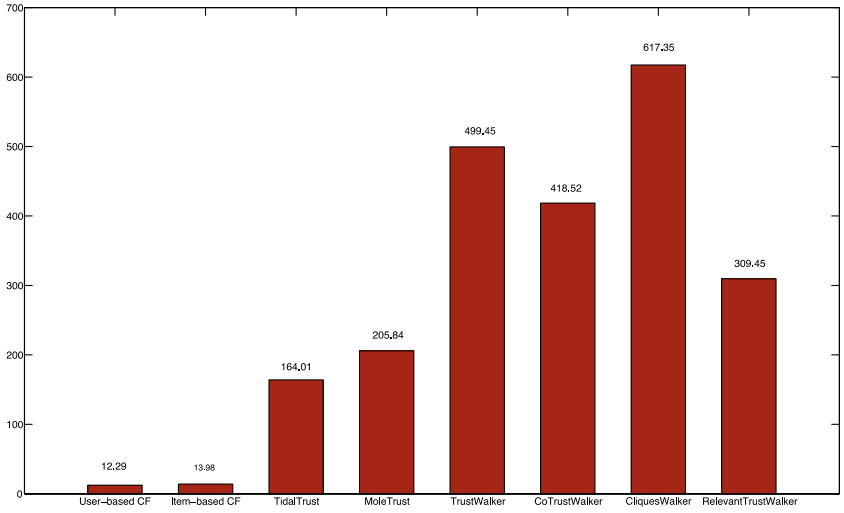


Fig. 6. Time costs of different algorithms.

the trust relationship between users as input, which increase the calculation of algorithms. Compared to the other random walk algorithm, RelevantTrustWalker costs much less in terms of time because, at each step of the walk, RelevantTrustWalker selects the target according to trust relevancy instead of selecting randomly, which makes the recommendation result tend to become stable more quickly to improve the calculation efficiency.

6. Conclusions

In solving the problem of personalized recommendations from large-scale service candidates, this paper has three main goals: (1) solving the problem of recommendations with cold-start users, (2) is solving the problem of recommendations with a large and spare user-service rating matrix, and (3) is solving the problem of recommendations with trust relations. To this end, this paper presents a trust-enhanced recommendation method, RelevantTrustWalker, the main contributions of which include (1) proposing the concept of trust relevancy, which is used to weight the existing social network. Furthermore, for this model, we (2) develop an improved random walk algorithm. Existing random walk methods usually randomly select the target node for each step when choosing to walk. By contrast, the proposed approach selects the target node based on trust relevance, which makes each walk step tend towards similar users. Thus, the recommendation contribution from trusted users is distinguished. We also (3) utilize large-scale real data sets to evaluate the accuracy of the algorithm. In this paper, the experimental data comes from the real-world Epinions website, and the data volume is extremely enormous. The experimental results show that the proposed approach can be directly applied in existing social networks and has high accuracy.

Personalized service recommendations from large-scale service candidates have been a heavily researched topic in recent years. The proposed method in this paper effectively solves this problem. We believe that the method still has much room for improvement. For example, in this paper, we consider the trust relationships between users in the social trust network as invariant. In fact, the trust relationship between users can change with time. In addition, the ratings from users are also time sensitive; ratings that are significantly out-of-date may become noise information for recommendations. Therefore, we plan to include time sensitivity for trust-enhanced recommendations in our future work.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant No. 61170033 and the National High-Tech Research and Development Plan of China under Grant No. 2013BAD19B10.

References

Alnemr, R., Bross, J., & Meinel, C. (2009). Constructing a context-aware service-oriented reputation model using attention allocation points. In *IEEE international conference on service* (pp. 451–457).

Andersen, R., Borgs, C., & Chayes, J. et al. (2008). Trust-based recommendation systems: an axiomatic approach. In *International conference on world wide web* (pp. 199–208).

Benesty, J., Chen, J., & Huang, Y. (2009). Pearson correlation coefficient. Noise reduction in speech processing. Springer, Berlin Heidelberg, 2(2): pp. 1–4.

Blake, M.B., & Nowlan, M.F. (2007). A web service recommender system using enhanced syntactical matching. In *IEEE international conference on web service* (pp. 575–582).

Chen, X., Liu, X., & Huang, Z. (2010). Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation. In *IEEE international conference on web services* (pp. 9–16).

Chen, L., Feng, Y., & Wu, J. et al. (2011). An enhanced qos prediction approach for service selection. In *IEEE international conference on service computing* (pp. 727–728).

Danielsson, P. E. (1980). Euclidean distance mapping. *Computer Graphics and Image Processing, 14*(3), 227–248.

Deng, S., Huang, L., Li, Y., & Yin, J. (2014a). Deploying data-intensive service composition with a negative selection algorithm. *International Journal of Web Service Research, 11*(1), 75–92.

Deng, S., Huang, L., Tan, W., & Wu, Z. (2014b). Top-k automatic service composition: a parallel framework for large-scale service sets. *IEEE Transactions on Automation Science and Engineering, 11*(3), 891–905.

Deng, S., Huang, L., Wu, J., & Wu, Z. (2014). Trust-based personalized service recommendation: a network perspective. *Journal of Computer Science and Technology, V29*(1), 69–80.

Deng, S., Huang, L., Wu, B., & Xiong, L. (2013b). Parallel optimization for data-intensive service Composition. *Journal of Internet Technology, 14*(6), 817–824.

Deng, S., Wu, B., & Wu, Z. (2013a). Efficient planning for top-k web service composition. *Knowledge and Information Systems, 36*(3), 579–605.

Golbeck, J.A. (2005). Computing and applying trust in web-based social networks.

Jamali, M., & Ester, M. (2009). TrustWalker: a random walk model for combining trust-based and item-based recommendation. In *International conference on knowledge discovery and data mining* (pp. 397–406).

Jiang, Y., Liu, J., & Tang, M. (2011). An effective web service recommendation method based on personalized collaborative filtering. In *IEEE international conference on web services* (pp. 211–218).

Jøsang, A., Roslan, I., & Colin, B. (2007). A survey of trust and reputation systems for online service provision. *Decision Support Systems, 43*(2), 618–644.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer, 42*(8), 30–37.

Krause, E. F. (1987). *Taxicab geometry: An adventure in non-euclidean geometry*. Courier Dover Publications.

Lipowski, A., & Dorota, L. (2012). Roulette-wheel selection via stochastic acceptance. *Physica A: Statistical Mechanics and its Applications., 391*(6), 2193–2196.
Maamar, Z., Hacid, H., & Huhns, M. N. (2011). Why web services need social networks. *IEEE Internet Computing, 15*(2), 90–94.
Massa, P., & Avesani, P. (2006). Trust-aware bootstrapping of recommender systems. In *ECAI workshop on recommender systems* (pp. 29–33).
Massa, P., & Avesani, P. (2007). Trust-aware recommender systems. In *ACM conference on recommender systems* (pp. 17–24).
Massa, P., & Avesani, P. (2007). Trust metrics on controversial users: balancing between tyranny of the majority. *International Journal on Semantic Web and Information Systems, 3*(1), 39–64.
Milgram, S. (1967). The small world problem. *Psychology Today, 2*(1), 60–67.
Mu-Hsing, K., Chen, L., & Liang, C. (2009). Building and evaluating a location-based service recommendation system with a preference adjustment mechanism. *Expert Systems with Applications, 36*(2), 3543–3554.
Nilashi, M., Othman, I., & Norafida, I. (2014). Hybrid recommendation approaches for multi-criteria collaborative filtering. *Expert Systems with Applications, 41*(8), 3879–3900.
Noor, T. H., & Sheng, Q. Z. (2014). Web service-based trust management in cloud environments. In *Advanced Web Services* (pp. 101–120). New York: Springer.
O'Donovan, J., & Smyth, B. (2005). Trust in recommender systems. In *International conference on intelligent user interfaces* (pp. 167–174).
Shao, L., Zhang, J., & Wei, Y. et al. (2007). Personalized QoS prediction for web services via collaborative filtering. In *IEEE international conference on web service* (pp. 439–446).
Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence, 2009*, 19. http://dx.doi.org/10.1155/2009/421425. Article ID 421425.
Thio, N., & Karunasekera, S. (2007). Web service recommendation based on client-side performance estimation. In *Australian software engineering conference* (pp. 81–89).
Walter, F. E., Battiston, S., & Schweitzer, F. (2008). A model of a trust-based recommendation system on a social network. *Autonomous Agents and Multi-Agent Systems, 16*(1), 57–74.
Wang, Rui., & Zeng, Guangzhou. (2010). An efficient service recommendation using differential evolutionary contract net for migrating workflows. *Expert Systems with Applications, 37*(2), 1152–1157.
www.epinions.com/.
Yang, C., & Sun, J. (2012). New Social recommendation model of random walks based on users groups relation mining. *Journal of Chinese Computer Systems, 33*(3), 565–570.
Ye, J. (2011). Cosine similarity measures for intuitionistic fuzzy sets and their applications. *Mathematical and Computer Modelling, 53*(1), 91–97.
Zhu, L., Xu, X., & Liu, Y. (2013). Collaborative filtering recommendation algorithm based on item and trust. *Computer Engineering, 39*(1), 58–62.