Soc. Netw. Anal. Min. (2014) 4:234
DOI 10.1007/s13278-014-0234-0

ORIGINAL ARTICLE

Recommender systems based on collaborative filtering and resource allocation

Amin Javari · Joobin Gharibshah · Mahdi Jalili

Received: 6 April 2014 / Revised: 23 September 2014 / Accepted: 13 October 2014 / Published online: 24 October 2014

Abstract Recommendation systems are important part of electronic commerce, where appropriate items are recommended to potential users. The most common algorithms used for constructing recommender systems in commercial applications are collaborative filtering methods and their variants, which is mainly due to their simple implementation. In these methods, structural features of bipartite network of users and items are used and potential items are recommended to the users based on a similarity measure that shows how similar the behavior of the users is. Indeed, the performance of the memory-based CF algorithms heavily depends on the quality of similarities obtained among users/items. As the obtained similarities are more reliable, better performance for the recommender systems is expected. In this paper, we propose three models to extract reliability of similarities estimated in classic recommenders. We incorporate the obtained reliabilities to improve performance of the recommender systems. In the proposed algorithms for reliability extraction, a number of elements are taken into account including the structure of the user-item bipartite network, the individual profile of the users, i.e., how many items they have rated, and that of the items, i.e., how many users have rated them. Among the proposed methods, the method based on resource allocation provides the highest performance as compared to others. Our numerical results on two benchmark datasets (Movielens and Netflix) shows that employing resource allocation in classical recommenders significantly improves their performance. These results are of great importance since including resource allocation in the systems does not increase their computational complexity.

A. Javari · J. Gharibshah · M. Jalili
Department of Computer Engineering, Sharif University of Technology, Tehran, Iran

M. Jalili (✉)
School of Electrical and Computer Engineering, RMIT University, Melbourne, Australia
e-mail: mjalili@sharif.edu

Keywords Social networks · Recommender systems · Similarity · Collaborative filtering · Resource allocation · Information theory

1 Introduction

Recommender systems get ever-increasing importance in computer science (Resnick and Varian 1997), which is mainly due to the popularity of online social networks (Garton et al. 1997). Modern societies have networked structures and complex networks can be found everywhere in daily life. The amount of information increases exponentially in the world and much of this information can be modeled as networks. User-item interaction is one of the frameworks which could be modeled by networks (Fields et al. 2011). An interesting question in these networks is to find a proper set of items to recommend to each user, which has found many applications in recent years (Sarwar et al. 2001; Adomavicius and Tuzhilin 2005; Deshpande and Karypis 2004; Symeonidis and Ntempos 2014). When a customer buys a product in an online store, the store may recommend additional titles based on the information on the users who have bought that particular product. A good example is Netflix which offers the users a list of movies he/she might be interested to watch, which is done based on the users' previous watching habits and ratings (James Bennett et al. 2007). The aim of recommender systems is to maximize the revenue by encouraging users to buy the items through recommending appropriate items such that if the users had used them, they would give high rates to them. A user-item network can be modeled as a bipartite network where the connections (e.g., user's opinions on the items) are from one part of the network, i.e., users' group, to the other part, i.e., items' group (Maslov and Zhang 2001; Zhou et al. 2007; Cattuto et al. 2007; Shang et al. 2010; Zhou et al. 2010). This bipartite network is the main ingredient for constructing recommender systems.

Collaborative filtering (CF) methods are the most popular recommenders in both industry and academia (Ahn 2008; Ben Schafer et al. 2007). These methods provide a recommendation list for any user based on the previous interactions between users and items. Another common approach for constructing reliable recommender systems is content-based filtering algorithms (Basu et al. 1998), which take into account the content of the items. Hybrid recommender systems—combining CF and content-based filtering methods—have been shown to be effective in some cases (Gunawardana et al. 2009; Burke 2002; Popescul et al. 2001). Since in many applications content of the items are difficult to access, CF algorithms are more prevalent methods employed for recommendation tasks. A prominent approach for building a recommendation system is to use memory-based CF (Sarwar et al. 2001; Konstan et al. 1997), which can be implemented in either user-based (John et al. 1998; Herlocker et al. 1999; Jin 2004) or item-based fashions (Deshpande and Karypis 2004; Badrul Sarwar et al. 2001). There are many other algorithms proposed for recommendation such as matrix factorization techniques (Koren et al. 2009), Bayesian networks (Yang et al. 2011) and latent semantic models (Hofmann 2004). Although these model-based techniques result in better personalization than CF, they have often high computational complexity for building the models that limit their applicability to large-scale systems.

Memory-based CF works by finding a similarity between users and recommending the items to a particular user based on those who have received positive feedback from its most similar users. Indeed, a one-mode projection is applied to the bipartite user–item network resulting in two networks; one among users and another among items (Zhou et al. 2007; Javari and Jalili 2013). Weighted connections among the users indicate their similarities in their preferences in item selection and valuation, i.e., the more the similar scores given by two users to the same items, the more the weight of their connecting link in the one-mode projected network. One of the main advantages of CF is that it does not need to have information on the content of the items and, therefore, it is capable of accurately recommending complex items to users. Although CF suffers from issues such as cold start, scalability and sparsity, it is still the most widely used recommender system in both academia and industry. The manner in which similarities between users and items are estimated in CF greatly influences its performance (Fischer 2001). Similarity extraction among a set of users is not only important in recommendation systems, but also has applications in other frameworks. In general, to predict behavior of a target user using CF, first similarity between the users is extracted, and then behavior of the target user is predicted based on behavior of its most similar users (Javari and Jalili 2014). However, because of the sparsity problem, the obtained similarity values are not reliable in many cases, and using unreliable similarities in the prediction process may decrease the performance of the recommender.

In this paper, we approach the problem (i.e., unreliability of extracted similarities) by introducing three models. The reliability values estimated by these methods are then used in the prediction process. We assume that a user who has reliable similarity with the target user is a better source for predictions. Indeed, we show that not only similarity scores are important in providing accurate recommendations, but also their reliability is important. We consider two principles in defining reliability of similarities. One may argue that as the number of samples used to extract similarities between the users increases, the obtained quantities become more reliable. Also, there are some items which are more informative for similarity extraction. Consider an item which is very popular among a large group of users and another item which is popular only among a small group of users. One may argue that based on Shannon entropy, common votes of two users on less popular items better describe their tastes. Our introduced models for reliability extraction (resource allocation-based, rank-based and diffusion-based models) have basis in these principles. We show that considering the reliability of similarities improves the performance of classical CF recommendation algorithm. The effectiveness of the proposed algorithm is revealed in Movielens and Netflix datasets that have been frequently used as benchmarks for assessing performance of recommender systems.

2 Related works

Many variants of CF-based recommendation methods have been introduced and a common topic in all of them is similarity extraction among a group of users (or items). Traditionally, measures such as Pearson correlation, Spearman rank correlation or cosine similarity are used in contracting CF algorithms Adomavicius and Tuzhilin (2005). These methods use the common votes of the users to extract similarities between them. It is also possible to consider contextual information on items and users to obtain more reliable similarity scores. It has been shown that the performance of memory-based CF can be improved using similarity metrics which are specifically designed to deal with sparsity or cold-start problems (Ahn 2008; Anand and Bharadwaj 2011; Bobadilla et al. 2012). Random walks can also be employed in extracting similarity scores, e.g., average first-passage time has been shown to be effective in similarity extraction (Fouss et al. 2007). Many classical models consider only local features of the bipartite users–items network for similarity extraction. A better way would be to consider global features (Bobadilla et al. 2012); (Anand and Bharadwaj 2011); however, this makes the computations more complex. Although different similarity measures have been introduced in the literature, most of practical implementations rely on classical vector similarity extraction methods such as Pearson correlation and cosine similarity, which is mainly due to their simple computation.

In this work, rather than proposing a new similarity extraction technique, we take into account the reliability of similarity extraction in the recommendation process. This can be combined with any similarity extraction method without increasing its computational complexity. Indeed, we assume that as a similarity score is more reliable, it would be more informative in the recommendation. Based on this assumption, we expect better performance for the conjunction of the reliability models with classic CF model than the pure CF model. From another viewpoint, it could be said that the introduced models for reliability extraction incorporate more information in the recommendation process than the classic functions for similarity extraction. By the help of the reliability values, we include information about the structure of the user–item bipartite network, whereas classical CF algorithm considers only users' rating vectors as the data source. It has been shown that the structure of the user–item network may be informative in different ways. The network could describe the diversity of users' interests, which may shed light to deal with the sparsity problem (Ni et al. 2014). Analyzing the structure of the network would be useful in detecting users with special behavioral patterns such as gray-sheep users (Ghazanfar and Prügel-Bennett 2014). The community structure can be used for designing algorithms as a remedy for scalability problem (Altingovde et al. 2013).

The proposed model for reliability extraction can also be applicable in neighborhood selection problem (Bellogín et al. 2013; Adamopoulos and Tuzhilin 2013). Some recommenders consider every user in the user space as neighbors for the target user; however, to decrease the computational complexity, some others select a subset of the user space as neighbors. Obviously, the strategy to select neighbors significantly influences the performance. Based on the proposed models to extract reliabilities, one can select neighbors with the highest reliability of similarity with the target. However, in this paper, we do not investigate this application of the proposed methods and particularly focus on increasing accuracy of recommendations.

3 Methods

A recommendation system consists of users and items where there are a number of ratings from users to items. Recommendation systems are often modeled as bipartite networks with users in one side and items in the other side. Let us denote items set as $I = [i_1, i_2, \ldots, i_n]$ and users set as $U = [u_1, u_2, \ldots, u_m]$; the network can be fully described by a bipartite $n \times m$ adjacency matrix $A = [a_{pq}]$, where $a_{pq} = 1$, if item i_q is rated (or collected) by user u_p, and $a_{pq} = 0$, otherwise. The network may also be weighted where the link weights are the valuation of users on items. Given this framework for recommendation problem, memory-based CF is one of the most popular methods, which has wide-spread use in many commercial applications. Since CF methods work based on the similarities between the users or items, the methodology to estimate the similarities and the reliability of estimations greatly influences the recommendation quality.

3.1 Collaborative filtering

Recently, collaborative filtering (CF) algorithms have been widely used for recommendation systems and they are indeed the most successful algorithms in this field (Ben Schafer DF et al. 2007). They often use proper similarity measures to define relevant interests. CF can be performed in either user-based (using users' similarity scores) or item-based (using items' similarity scores) fashions (Manos Papagelis 2005). An assumption in user-based CF is that the users who had common idea about an item in the past will agree on similar items in the future. User-based CF systems recommend items to a user according to the opinions of other users. To this end, proper statistical methods are often used to find a set of users who have a history of agreeing with the target user by rating or selecting similar set of items. To this end, the similarity between two users can be measured by computing Pearson correlation coefficient, cosine similarity (COS), constrained Pearson correlation (CPC) or Spearman rank correlation (SRC) that are the most frequently used effective technique (Ahn 2008; Gediminas Adomavicius 2005). The Pearson similarity between users u_i and u_j is obtained as

$$\mathrm{Sim}_{\mathrm{Pearson}}(u_i, u_j) = \frac{\sum_{h=1}^{n} (R_{u_i,h} - \overline{R_{u_i}})(R_{u_j,h} - \overline{R_{u_j}})}{\sqrt{\sum_{h=1}^{n} (R_{u_i,h} - \overline{R_{u_i}})^2} \sqrt{\sum_{h=1}^{n} (R_{u_j,h} - \overline{R_{u_j}})^2}}, \quad (1)$$

where $R_{u_i,h}$ denotes the rating of user u_i on item i_h and $\overline{R_{u_i}}$ is the average rating of user u_i. n is the number of items purchased (or rated) by users u_i and u_j.

Similarly, cosine similarity ($\mathrm{Sim_{COS}}$), Constrained Pearson correlation ($\mathrm{Sim_{CPC}}$) and Spearman rank correlation ($\mathrm{Sim_{SRC}}$) are defined as

$$\mathrm{Sim_{COS}}(u_i,u_j)=\frac{\sum_{h=1}^{n}R_{u_i,h}R_{u_j,h}}{\sqrt{\sum_{h=1}^{n}R_{u_i,h}^2}\sqrt{\sum_{h=1}^{n}R_{u_j,h}^2}}, \tag{2}$$

$$\mathrm{Sim_{CPC}}(u_i,u_j)=\frac{\sum_{h=1}^{n}(R_{u_i,h}-R_{\mathrm{med}})(R_{u_j,h}-R_{\mathrm{med}})}{\sqrt{\sum_{h=1}^{n}(R_{u_i,h}-R_{\mathrm{med}})^2}\sqrt{\sum_{h=1}^{n}(R_{u_j,h}-R_{\mathrm{med}})^2}}, \tag{3}$$

$$\mathrm{Sim_{SRC}}(u_i,u_j)=1-\frac{\sum_{h=1}^{n}d_h^2}{n(n^2-1)}, \tag{4}$$

where $\mathrm{R_{med}}$ in Eq. (3) is median of the rating scale (e.g., $\mathrm{R_{med}}=3$ for the rating scale of [0, 5]) and d_h^2 in Eq. (4) is the difference between the rank for item h by two users u_i and u_j.

As the similarity between users is obtained, the weighted sum of user-based similarities is computed, which indeed adjusts the users' ratings. Then, the prediction value of the rating of user u_x to item y, $P(u_x,y)$, is obtained as

$$P(u_x,y)=\overline{R_{u_x}}+\frac{\sum_{i=1}^{m}(R_{u_i,y}-\overline{R_{u_i}})\mathrm{Sim}(u_x,u_i)}{\sum_{i=1}^{m}|\mathrm{Sim}(u_x,u_i)|}, \tag{5}$$

where $\overline{R_{u_x}}$ is average rating of user x and $R_{u_i,h}$ is the rating item y receives from user u_i. $\mathrm{Sim}(u_x,u_i)$ is the similarity between u_x and u_i that is obtained from one of Eqs. (1)–(4).

3.2 Reliability of similarity scores

The methodology used to estimate similarities between users or items has a significant influence on the performance of memory-based CF. Considering similarity extraction methods expressed by Eqs. (1)–(4), in this section, we introduce the concept of reliability in estimated similarity between two users, i.e., the confidence level for estimated similarities. We modify the prediction method as

$$P(x,y)=\overline{R_x}+\frac{\sum_{i=1}^{m}(R_{u_i,y}-\overline{R_{u_i}})\mathrm{Sim}(x,u_i)R(x,u_i)}{\sum_{h=1}^{m}|\mathrm{Sim}(x,u_i)R(x,u_i)|}, \tag{6}$$

where $R(x,u_i)$ incorporates the reliability of the estimated similarity between users x and u_i. In other words, we strengthen (weaken) the weights of the ratings for users who have more (less) reliable similarities with the target user x. Our proposed model for incorporating reliability of similarities in the user-based CF for top-N recommendation task can be summarized in the following pseudo-code:

```
– for each item $y$ that user $x$ has no vote on
    • for each user $u_i$ that has voted on item $y$
        • compute similarity $\mathrm{Sim}(x,u_i)$ between $x$ and $u_i$
        • compute reliability $R(x,u_i)$ of $\mathrm{Sim}(x,u_i)$ between $x$ and $u_i$
        • calculate average of user $x$'s vote for item $y$, weighted by $\mathrm{Sim}(x,u_i)$ and $R(x,u_i)$
– return top-$N$ items (based on weighted average).
```

Now, the question is how to obtain the reliability of similarity values. In this paper, we propose three models for reliability extraction. The proposed methods have basis in two main principles. In the next subsections, we argue these principles and propose three models for reliability extraction based on them.

3.2.1 *Reliability of similarities dependent on number of common ratings*

In statistics, estimation is considered as a process by which parameters about a population can be inferred based on the information extracted from samples. It can be simply shown that as the number of samples increases, the estimated parameter of the population is less likely to miss the real value (Hoel 1954). For instance, let us consider the estimation of population mean based on n random samples. Expected value in which the estimation misses the population mean, called standard error of the mean (SE), can be obtained as

$$\mathrm{SE}=\frac{\sigma}{\sqrt{n}}, \tag{7}$$

where σ is the population standard deviation. Thus, the error for mean estimation decreases by the square root of the number of samples. Similarly, one may argue that the higher the number of items rated by two users, the more reliable the estimated similarity between them.

3.2.2 *Importance of unpopular items in similarity extraction*

Let us look at the problem from information theory point of view. According to information theory and Shannon entropy, the lower the probability of a probabilistic event, the more information received when the event occurs (Cover and Thomas 2012). Given a set of outcomes [e_1, e_2, …, e_n] for a random variable V, self-information (SI) for each outcome is defined as

$$\mathrm{SI}(e_i) = -\log P(e_i), \tag{8}$$

where $P(e_i)$ is the probability for the outcome e_i. Indeed, self-information for the outcome e_i quantifies the information contained in occurrence of e_i. Thus, an outcome with lower probability has higher self-information.

We may consider each item as an outcome of a random variable. In this way, popularity of the items represents their probability of occurrence. Thus, relative to self-information, items with lower popularity are more informative than those with higher popularity. In other words, similarity scores extracted based on items with lower popularity are more valuable and reliable than those based on items with higher popularity. Intuitively, two users who have purchased the same popular item are less likely to have similar tastes compared to the users who have purchased a rarely attended item (an item with low popularity). Since popular items attract attentions from users with different tastes, they are not valuable resources to extract similarities. For example, in movie recommendation framework, consider a movie which has won a prestigious award. Obviously, this movie attracts a lot of users whilst most of them necessarily do not have similar tastes. Consider another movie that is popular among only a set of users with special favors. The users visiting such a movie will have similar tastes. Next, we propose some methods to consider such issues in the design of CF.

3.3 Reliability extraction methods

As mentioned, the similarities obtained from classical similarity extraction functions do not have the same reliability values. As we give more weights for reliable similarities in the recommendation process, better performance is expected. However, the problem is how to define reliability values for similarity scores. In the next section, we articulate three models based on the principles introduced above.

3.3.1 Resource allocation-based model (RA)

Resource allocation is a well-known method in link prediction problems where the likelihood of existence of a link between non-connected nodes is estimated. There are a number of methods for link prediction in complex networks (Lü and Zhou 2011). Some of these methods use structural information of the network to predict the pair of nodes likely to be connected in future. Among such methods, the ones based on common neighbors, Adamic-Adar index (David Liben-Nowell 2007; Adamic EA 2003) and resource allocation (RA) index (Tao Zhou and Zhang 2009) have shown better estimation of the missing links in bipartite networks of users and items. RA is a degree-based local similarity measure, in which its value for two users depends on the degree of the items selected by both users; i.e., the more the popular items are selected by two users, the less the value of their RA index. In this way, the contribution of similarity between users who have selected popular items is weakened. According to RA algorithm, a pair of nodes—which are not connected directly—can share their resources through their common neighbors. If each node distributes its resources equally to all of its neighbors, the RA-reliability—indicating the degree of reliability for estimated similarity between users u_i and u_j—is obtained as

$$R_{\mathrm{RA}}(u_i, u_j) = \sum_{z \in \Gamma(u_i) \cap \Gamma(u_j)} \frac{1}{k_z}, \tag{9}$$

where k_z is degree of item z, i.e., the number of users who have rated the item, and $\Gamma(u_i)$ is set of the neighbors of user u_i that is indeed the number of items that user u_i has rated. Equation (9) indicates that the RA values depend on both the number of common neighbors and their degree. Obviously, RA index, as a reliability parameter, addresses the two principles we discussed in the previous section. As $R_{\mathrm{RA}}(i,j)$ is computed, it is used in the prediction method (Eq. 6) as a parameter indicating the reliability of similarity estimation. Let us denote the combination of RA value with classical CF as CF-RA.

3.3.2 Ranking-based model (Rank)

One can take into account the items' ranking in the network as their popularity and use this information for measuring the reliability of similarities between users. To rank the nodes in the network, we used Hyperlink-Induced Topic Search (HITS)—a well-known ranking algorithm (Kleinberg 1999). In HITS ranking algorithm, there are two vectors for the nodes, i.e., their hub and authority vectors. The initial value for hub and authority is taken as unity, and then, the following recursive update rules are applied for computing them

$$\begin{cases} \mathrm{hub}_p = \sum_{q=1}^{n} \mathrm{aut}_q \\ \mathrm{aut}_p = \sum_{q=1}^{n} \mathrm{hub}_q \end{cases} ; \begin{cases} \mathrm{hub}_q = \sum_{p=1}^{m} \mathrm{aut}_p \\ \mathrm{aut}_q = \sum_{p=1}^{m} \mathrm{hub}_p \end{cases}, \tag{10}$$

where $\mathrm{hub_p}$ and $\mathrm{aut_p}$ are hub and authority values of user u_p, respectively ($\mathrm{hub_q}$ and $\mathrm{aut_q}$ are those for item i_q). m and n represent the number of users and items, respectively.

HITS has guaranteed convergence and is one of the most frequently used ranking methods in the literature (Kleinberg 1999). As the ranking algorithm converges, the authority values for the items indicate their rankings in the network. These authority values somehow represent the popularity of the items and can be used for calculating the reliability measure, i.e., the higher the authority value of an item, the less informative the item in measuring similarity between the users. Having the authority values for the items, the reliability index for the similarity between the users is obtained as

$$R_{\mathrm{Rank}}(i,j) = \sum_{z \in \Gamma(u_i) \cap \Gamma(u_j)} \frac{1}{\mathrm{aut}_z}, \tag{11}$$

where aut_z is the authority value for item z which has been selected by both users u_i and u_j. Similar to R_{RA}, R_{Rank} also finds items with high popularity; however, R_{rank} considers items' authority in HITS ranking algorithm as their popularity, whereas R_{RA} takes into account their degree. $R_{\mathrm{rank}}(i,j)$ is then used in Eq. (6) to make the prediction. Let us denote the combination of Rank value with classic CF as CF-Rank.

3.3.3 *Diffusion-based model (Diff)*

An alternative approach for computing reliability of similarities between the users is to consider not only the items they rate similarly, but also their individual characteristics. Indeed, selecting a non-popular item by a less active user is more informative as compared to the case when the same item is selected by an active user. For instance, a user who watches many movies is likely to watch many others in the future. However, when a less active user selects a movie to watch, this indicates that there should be something special in the movie that has encouraged the user to watch it. The individual characteristics of the users can greatly enhance the outcome of the recommender systems through providing better estimation for the reliability of similarities between the users.

Here, we used two-phase diffusion process to provide such information to use for the recommendation system (Liu Jian-Guo WB-HAGQ 2009; Zan Huang and Zeng 2004). The method has two diffusion steps, which is based on resource allocation procedure. In the first step, each item absorbs its resources among all users who have rated the item and the resource allocated to item i_q reads as (Liu Jian-Guo WB-HAGQ 2009; Zan Huang and Zeng 2004)

$$f_q = \sum_{j=1}^{m} \frac{a_{jq} f_j}{k_j}, \tag{12}$$

where f_q is the resource the item i_q receives from the users and k_i is degree of user u_i on bipartite user–item graph. f_i is an initial resource allocated to user u_i, which is corrected in the next step.

As the algorithm proceeds, each item sends back what has received in the previous step to all users who have selected the item. The resource allocated to user u_i is calculated as

$$f_i = \sum_{q=1}^{n} \frac{a_{iq} f_q}{k_q} = \sum_{q=1}^{n} \frac{a_{iq}}{k_q} \sum_{j=1}^{m} \frac{a_{jq} f_j}{k_j} = \sum_{j=1}^{m} \frac{1}{k_j} \sum_{q=1}^{n} \frac{a_{iq} a_{jq}}{k_q} f_q, \tag{13}$$

where a_{jq} is the entries of the adjacency matrix of the bipartite users–items graph, which has only a non-zero value when there is a rating from user u_j to item i_q.

Inspired by above formulation, the diffusion-based similarity between nodes u_i and u_j can be obtained as

$$R_{\mathrm{Diff}}(i,j) = \frac{1}{k_j} \sum_{q=1}^{n} \frac{a_{iq} a_{jq}}{k_q}. \tag{14}$$

It is worth mentioning that CF-RA algorithm has lower computational complexity compared to other two methods, and its computational complexity is the same as standard CF algorithm. We use the term CF-Diff to refer to the combination of Diff value with classical CF.

4 Experimental results

The performance of the above recommender algorithms was evaluated in two real datasets: Movielens (Konstan et al. 1997; Resnick NI et al. 1994) and Netflix (James Bennett et al. 2007). These datasets have been frequently used as benchmarks in recommender systems (Fidel Cacheda Vic et al. 2011). Movielens and Netflix consist of ratings to the movies on a scale of 1–5 with 5 being excellent and 1 being terrible. In our experiments, we used Movielens dataset of 943 users and 1,682 items with 100,000 ratings (i.e., weighted links) from the users to the items. Moreover, a subset of Netflix dataset with 902 users and about 8,000 items was used in experiments. The statistics of the datasets is summarized in Table 1.

To evaluate the performance of the recommendation algorithms, we tested them on the datasets and used five-fold cross-validation algorithm (Kohavi 1995). In other words, the cross-validation procedure consists of 5

Table 1 Statistics of datasets

	Movielens	Netflix
Number of users	943	902
Number of items	1682	8068
Number of edges	100,000	100,292

iterations and in each of the iterations, 80 % of the links (i.e., ratings made by the users) is considered as training data, while others is taken into account as test data. This procedure is repeated 5 times in which at any time a new 20 % of the links are considered as test data.

There are a number of metrics for assessing the performance of a recommendation system. In this work, we used two metrics, namely, mean absolute error (MAE) and area under curve (AUC). MAE compares the numerical recommendation scores against the actual rating values given by the users, as

$$\mathrm{MAE} = \frac{\sum_{k=1}^{q} |P_k - R_k|}{q}, \tag{15}$$

where P_k and R_k are the predicted value and the real rating for k-th link, respectively, and q is the total number of predicted links in the network.

MAE shows to how much extent the prediction results are accurate in which the lower is the value of MAE, the more accurate the results. Although MAE works well for measuring the precision of individual algorithms, it might not be applicable for evaluating the prediction precision of individual classes (when the ratings are considered in various classifications, i.e., classes 1–5). To evaluate the ability of the recommendation algorithms in predicting the class—rather than the rating value—we used AUC. This metric gives a complete characterization of the performance across all classes (classes 1–5) of ratings. AUC is computed as

$$\mathrm{AUC} = \frac{\mathrm{TP}_{\mathrm{rate}} + \mathrm{TN}_{\mathrm{rate}}}{2}, \tag{16}$$

where $\mathrm{TP}_{\mathrm{rate}}$ indicates hit ratio and $\mathrm{TN}_{\mathrm{rate}}$ corresponds to correct rejection ratio of a target class. Hit ratio shows the number of items where their predicted class is the same as the actual class they belong to.

4.1 Comparing the performance of different algorithms for reliability extraction

The first experiment was designed to evaluate the performance of four recommender systems introduced in the previous section (CF, CF-RA, CF-Rank and CF-Diff). We also compared the proposed methods with a recent extension of CF (Bellogín et al. 2013), which is based on employing reliability in forming the algorithm. The models introduced in (Bellogín et al. 2013) are for neighborhood selection problem. As we discussed, in the neighborhood selection problem, the task is to select a subset of users who are more informative concerning the target user. We assume that reliable neighbors for the target user better represent his/her taste, and consequently, are better candidates to be selected as appropriate neighbors. This model suggests that a user with more common preferred items with the target user is likely to be an appropriate neighbor. (Bellogín et al. 2013) define neighbor appropriateness (we refer it as reliability) as a function of overlap degree. In this way, they introduced two overlap functions: Herlocker's weighting and McLaughlin's weighting. The Herlocker's weighting between the users u_i and u_j, HW(u_i,u_j), is defined as:

$$\mathrm{HW}(u_i, u_j) = \frac{\max(|\Gamma(u_i) \cap \Gamma(u_j)|, q)}{q}, \tag{17}$$

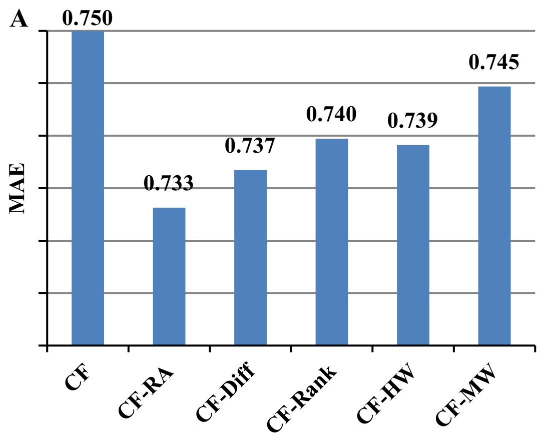
where q denotes the threshold of what is considered as small overlap between the users and $|\Gamma(u_i) \cap \Gamma(u_j)|$ represents the intersection size. McLaughlin's weighting between the users u_i and u_j, HW(u_i,u_j), is defined as:

$$\mathrm{MW}(u_i, u_j) = \frac{\min(|\Gamma(u_i) \cap \Gamma(u_j)|, q)}{q}. \tag{18}$$

We refer the combination of the values obtained from Herlocker's weighting and McLaughlin's weighting with classic CF as CF-HW and CF-MW, respectively. Note that, here, CF is based on Pearson correlation. In the experiments, the algorithms were applied to Movielens and Netflix datasets and the resulting MAEs were obtained. For both CF-MW and CF-HW, we set $q = 50$. As it can be seen, in general, combining the classical CF algorithm with reliability values could significantly improve its performance (Fig. 1). Computing the similarity between the users based on their rankings obtained through HITS algorithm reduced the MAE by 0.01 in Movielens and 0.004 in Netflix. The diffusion-based algorithm and algorithms based on overlap size (CF-HW and CF-MW) could further improve the MAE; however, the item-based resource allocation method in conjunction with CF (CF-RA) showed the best performance among the six. CF-RA could improve the MAE by 0.017 (Movielens) and 0.016 (Netflix) as compared to standard CF. These results are important achievement since the computational complexity of CF-RA is comparable to that of standard CF and is much lower than the other two algorithms. Next, we focus on CF and CF-RA algorithms and compare their performance from different perspectives.

4.2 CF-RA vs. CF

Since CF and CF-RA have lower computational complexity as compared to CF-Rank and CF-Diff, and thus can be applied in real systems, in this section we compared their performance in a detailed fashion. We investigated to how much extent employing resource allocation in the items can improve the AUC values. In addition to calculating AUCs jointly for all classes of rating, we also



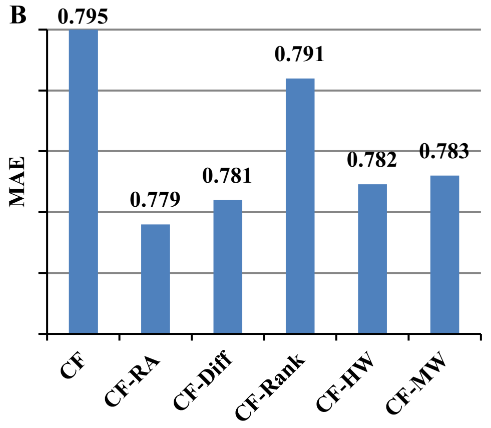


Fig. 1 Mean absolute error (MAE) of the recommendation systems based on four algorithms (collaborative filtering (CF), collaborative filtering with recourse allocation (CF-RA), diffusion-based collaborative filtering (CF-Diff), rank-based collaborative filtering (CF-Rank), Herlocker's weighting-based CF (CF-HW) and McLaughlin's weighting-based CF (CF-MW); see text for description of the methods) in **a** Movielens and **b** Netflix datasets

Table 2 AUC values in Movielens and Netflix datasets

	Class 1	Class 2	Class 3	Class 4	Class 5	Total
Number of link in each class	6,110	11,370	27,145	34,174	21,201	100,000
Movielens						
CF	0.540	0.546	0.604	0.591	0.547	0.5656
CF-RA	**0.544**	**0.552**	0.604	**0.598**	**0.569**	**0.5734**
Number of link in each class	4,959	9,586	26,285	33,812	25,657	100,299
Netflix						
CF	0.521	**0.521**	**0.586**	**0.560**	**0.580**	**0.5536**
CF-RA	0.521	0.524	0.594	0.569	0.595	0.5606

Bold digits show where the performance of CF-RA is better than that of CF

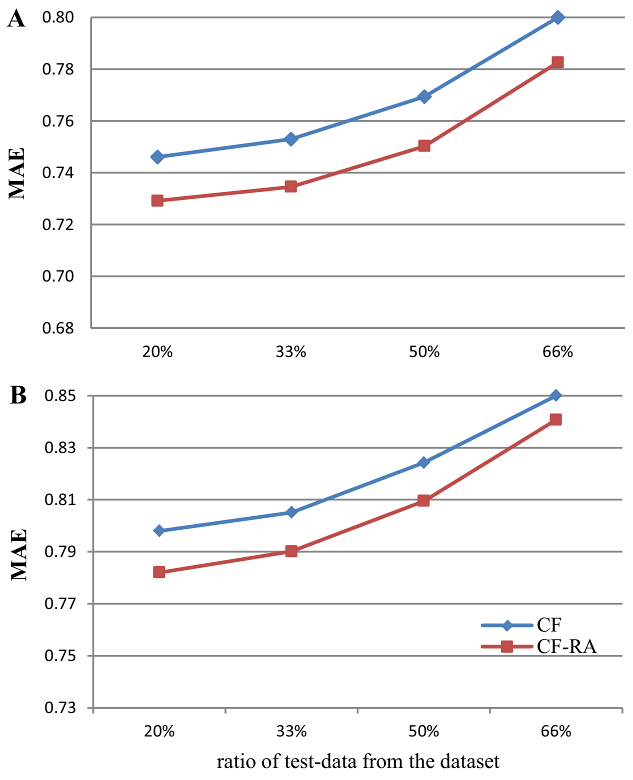


Fig. 2 MAE as a function of the size of test data (in cross-validation algorithm) in **a** Movielens and **b** Netflix datasets

calculated them separately for each class. Table 2 summarizes the AUC values for various classes. CF-RA has better (or equal) performance in all classes as compared to CF. The most improvement was achieved for class 5 where the performance of CF was improved over 4 and 2.6 % in Movielens and Netflix, respectively. Class 5 is the most important class showing the class where the users gave the highest scores to the items. Our results showed that employing item-based resource allocation is significantly important in improving the performance of CF algorithm in predicting the highly rated items to the users. CF-RA also showed improved AUCs as compared to standard CF when considering all classes together.

In cross-validation algorithms, the ratio of training-test data has significant influence on the final outcome of prediction (Abdessalem et al. 2009). Therefore, we considered different ratios for the test dataset as 20, 33, 50 and 66 %. Figure 2 shows the MAE values of CF and CF-RA algorithms as a function of the ratio of test data. Not surprisingly, as the ratio of test data increased, MAE increased, and thus, the prediction performance declined. This is due to the fact that by increasing the ratio of test data (i.e., decreasing the ratio of training data), the portion of data used for training reduces, and as expected, the

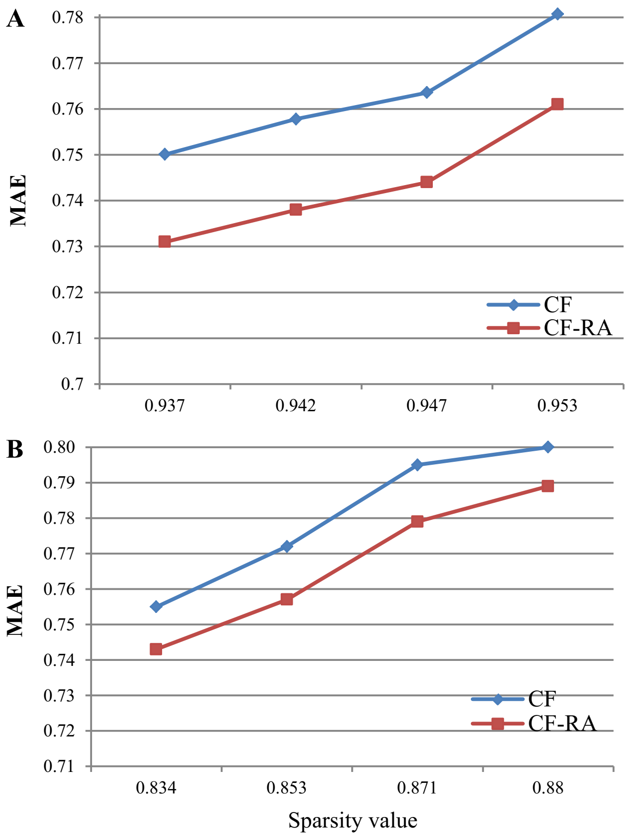


Fig. 3 MAE as a function of sparsity in **a** Movielens and **b** Netflix datasets

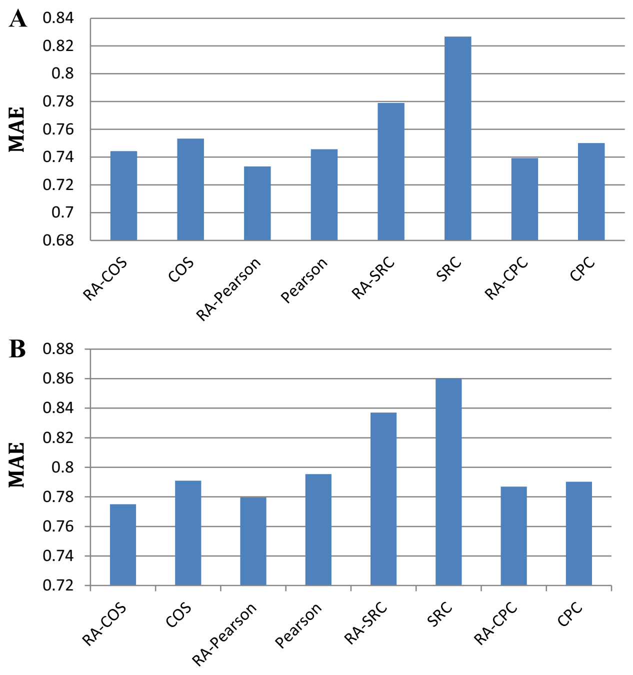


Fig. 4 MAE of user-based CF with different techniques used to extract similarity among users: Cosine similarity (COS), Pearson Correlation (Pearson), Spearman rank correlation (SRC), constrained Pearson correlation (CPC). RA indicates the method when resource allocation is considered as reliability measures. The results are for **a** Movielens and **b** Netflix datasets

prediction performance worsens. However, in all cases, CF-RA had lower MAE compared to standard CF, meaning that regardless of the size of test set, item-based resource allocation could improve the performance of CF.

It has been shown that recommendation systems are sensitive to sparsity level of the datasets (Manos Papagelis 2005). Therefore, we also changed sparsity of both datasets by eliminating users with their connecting links. The sparsity of the network is calculated by subtracting from unity the ratio of existing links in the network divided by the maximum number of possible links from users to items (Manos Papagelis 2005), more precisely,

$$\text{Sparsity} = 1 - \frac{\text{Number of links in the network}}{\text{Possible links between users and objects}}. \tag{19}$$

The sparsity value was changed by sequentially removing users with high degrees; the sparsity of Movielens was changed from 0.937 to 0.953 and that of Netflix from 0.834 to 0.88. As sparsity reduces (by removing active users), the link prediction is performed on less active users, and as expected, the MAE decreases. Figure 3 shows MAEs as a function of sparsity in the datasets; again CF-RA outperformed standard CF in all sparsity values for the datasets. CF-RA and CF showed almost the same sensitivity to the sparsity and the ratio of test dataset.

4.3 Applying resource allocation on different similarity extraction methods

The above experiments were carried out taking into account Pearson correlation as a similarity measure in CF. In this section, we investigated whether the proposed resource allocation strategy is helpful when other similarity metrics are used in CF. To this end, we considered constrained Pearson correlation, Spearman rank correlation and Cosine similarity. Since RA-reliability outperformed the other two reliability extraction models, in this section, we considered only RA. Figure 4 shows the MAE values for user-based CF with different similarity metrics and with/without resource allocations strategy. As it can be seen, in all cases, applying the RA-reliability improved the performance of the recommendations. For Movielens (Netflix) dataset, considering RA, the performance of CF with Spearman rank correlation as a similarity metric was improved more than 4 % (2 %). When combined with RA, the Cosine-based CF algorithm showed the best performance in Netflix dataset, while the Pearson-based CF was the best performed in Movielens dataset.

Considering RA in the recommendation process enables us to take into account further structural information in extracting the similarity scores (i.e., their reliability), which helps in providing more accurate recommendations. This is particularly interesting since computing RA does not increase the computational load of the algorithms. To extract the similarity between two users based on either of the metrics, first, common items voted by these users are obtained, and then, the similarity between the rating values is calculated. According to Eq. (9), to extract RA-based reliability measure, similar to similarity extraction methods, common votes of the target users should be extracted. Thus, the RA values can be calculated at the same time when the similarity values are calculated.

5 Conclusion and future work

Recommender systems are frequently used in many applications such as movie stores, music distributions, book stores and online social networks. The most common algorithm used for recommender systems is CF which is based on suggesting proper items to the users based on their similarities. However, estimation of similarity scores based on correlation methods may have different confidence values. Classical CF algorithms do not take into account these confidence values and reliability of similarities. To this end, we proposed three algorithms: resource allocation-based CF, diffusion-based CF and ranking-based CF. Intuition behind the proposed methods is that the insights obtained from the links to popular items are often less informative than those obtained from less popular items and also similarities estimated based on higher number of items are more reliable. Furthermore, the information obtained from silent users is more useful than those obtained from active ones. We performed experiments on two benchmark datasets: Movielens and Netflix, which have been frequently used for evaluating performance of recommender systems. The method based on resource allocation, among others, showed the best performance by resulting in the least mean absolute error. It significantly outperformed standard CF. Since adding resource allocation to CF does not increase the computational complexity of the algorithm, it has practical applications in real recommender systems. We also considered various metrics for similarity extraction in CF (including Pearson correlation, constrained Pearson correlation, Spearman rank correlation and Cosine similarity); applying resource allocation improved the accuracy in all of them.

As future works, more detailed structural information on the users and items can be considered. For example, one may take into account the community structure in the one-mode projected networks and use it in CF. Also, one of the important issues that has attracted attention in this field is diversity of recommendations. Applying CF for recommendation often results in poor diversity and it would be interesting to study how the balance between diversity and prediction accuracy can improve the performance of the system.

References

Abdessalem T, Cautis B, Souhli A (2009) Trust management in social networks. Télécom ParisTech, LTCI, UMR CNRS 514146

Adamic EA LA (2003) Friends and neighbors on the web. Soc Netw 25:211–230

Adamopoulos P, Tuzhilin A (2013) Probabilistic Neighborhood Selection in Collaborative Filtering Systems

Adomavicius G, Tuzhilin A (2005a) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Audio Electroacoust Knowl Data Eng 17:734–749

Adomavicius G, Tuzhilin A (2005b) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. Knowl Data Eng, IEEE Trans 17(6):734–749

Ahn HJ (2008) A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. Inf Sci 178:37–51

Altingovde IS, Subakan ÖN, Ulusoy Ö (2013) Cluster searching strategies for collaborative recommendation systems. Inf Process Manage 49:688–697

Anand D, Bharadwaj KK (2011) Utilizing various sparsity measures for enhancing accuracy of collaborative recommender systems based on local and global similarities. Expert Syst Appl 38:5101–5109

Badrul Sarwar GK, Joseph Konstan, and John Riedl (2001) ItemBased Collaborative Filtering Recommendation Algorithms. Hong Kong

Basu C, Hirsh H, Cohen W (1998) Recommendation as classification: Using social and content-based information in recommendation. National Conference on Artificial Intelligence

Bellogín A, Castells P, Cantador I (2013) Improving memory-based collaborative filtering by neighbour selection based on user preference overlap. LE CENTRE DE HAUTES ETUDES INTERNATIONALES D'INFORMATIQUE DOCUMENTAIRE pp 145–148

Ben Schafer DF J, Herlocker Jon, Sen Shilad (2007) Collaborative filtenng recommender systems. LNCS 4321:291–324

Bobadilla J, Ortega F, Hernando A, Bernal J (2012) A collaborative filtering approach to mitigate the new user cold start problem. Knowl-Based Syst 26:225–238

John S. Breese DH, and Carl Kadie (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering pp 43–52

Burke R (2002) Hybrid recommender systems: survey and experiments. User Model User-Adapt Interact 12:331–370

Cattuto C, Loreto V, Pietronero L (2007) Semiotic dynamics and collaborative tagging. Proc Natl Acad Sci USA 104:1461–1464

Cover TM, Thomas JA (2012) Elements of information theory, Wiley

David Liben-Nowell JK (2007) The link-prediction problem for social networks. Am Soc Info Sci Technol 58:1019–1031

Deshpande M, Karypis G (2004) Item-based top-N recommendation algorithms. ACM Trans Inf Syst 22(1):143–177

Cacheda F, Carneiro V, Fernandez D, Formoso V (2011) Comparison of collaborative filtering algorithms: limitations of current techniques and proposals for scalable, high-performance recommender systems. ACM Trans Web 5:2

Fields B, Jacobson K, Rhodes C, d'Inverno M, Sandler M et al (2011) Analysis and exploitation of musician social networks for recommendation and discovery. IEEE Trans Multimed 13:674–786

Fischer G (2001) User modeling in human–computer interaction. User Model User-Adap Inter 11:65–86

Fouss F, Pirotte A, Renders J-M, Saerens M (2007) Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. Knowl Data Eng, IEEE Trans 19:355–369

Garton L, Haythornthwaite C, Wellman B (1997) Studying online social networks. J Comp Mediat Commun 3:1–32

Gediminas Adomavicius AT (2005) Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. IEEE Trans Knowl Data Eng 17:734–749

Ghazanfar MA, Prügel-Bennett A (2014) Leveraging clustering approaches to solve the gray-sheep users problem in recommender systems. Expert Syst Appl 41:3261–3275

Gunawardana A, Meek C (2009) A unified approach to building hybrid recommender systems. ACM pp 117–124

Herlocker JL, Konstan JA, Borchers A, Riedl J (1999) An algorithmic framework for performing collaborative filtering. Berkeley, California, pp 230–237

Hoel PG (1954) Introduction to mathematical statistics. Introduction to mathematical statistics

Hofmann T (2004) Latent semantic models for collaborative filtering. ACM Trans Inf Syst (TOIS) 22:89–115

James Bennett CE, Bing Liu, Padhraic Smyth and Domonkos Tikk (2007) KDD Cup and workshop 2007. ACM SIGKDD Explorations Newsletter 9

Javari A, Jalili M (2013) Accurate and novel recommendations: An algorithm based on popularity forecasting. ACM Trans Intell Syst Technol (to appear)

Javari A, Jalili M (2014) Cluster-based collaborative filtering for sign prediction in social networks with positive and negative links. ACM Trans Intell Syst Technol (TIST) 5:24

Jin R, Chai JY,Si L (2004) An automatic weighting scheme for collaborative filtering Sheffield, UK pp 337–344

Kleinberg JM (1999) Authoritative sources in a hyperlinked environment. J ACM 46:604–632

Kohavi R(1995) A study of cross-validation and bootstrap for accuracy estimation and model selection. San Francisco, CA pp 1137–1143

Konstan J, Miller B, Maltz D, Herlocker J, Gordon L et al (1997) GroupLens: applying collaborative filtering in usenet news. Commun ACM 40:77–87

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

Liu Jian-Guo WB-HAGQ (2009) Improved Collaborative Filtering Algorithm via Information Transformation. Int J Mod Phy C 20:285–293

Lü L, Zhou T (2011) Link prediction in complex networks: A survey. Phys A 390:1150–1170

Manos Papagelis DP (2005) Qualitative analysis of user-based and item-based prediction algorithms for recommendation agents. Eng Appl Artif Intell 18:781–789

Maslov S, Zhang Y-C (2001) Extracting hidden information from knowledge networks. Phys Rev Lett 87:248701

Ni J, Zhang Y-L, Hu Z-L, Song W-J, Hou L et al (2014) Ceiling effect of online user interests for the movies. Physica A 402:134–140

Popescul A, Ungar LH, Pennock DM, Lawrence S (2001) Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. Seventeenth Conference on Uncertainty in Artificial Intelligence

Resnick P, Iacovou N, Suchak M, Bergstorm P, Riedl J (1994) GroupLens: an open architecture for collaborative filtering of netnews. ACM Press, Chapel Hill, pp 175–186

Resnick P, Varian HR (1997) Recommender systems. Commun ACM 40:56–58

Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms. 10th international conference on World Wide Web: ACM

Shang M-S, Lu L, Zhang Y-C, Zhou T (2010) Empirical analysis of web-based user-object bipartite networks. Europhys Lett 92:48006

Symeonidis P, Ntempos D, Manolopoulos Y (2014) Recommender systems for location-based social networks: Springer

Tao Zhou LL, Zhang Yi-Cheng (2009) Predicting missing links via local information. Eur Phy 71:623–630

Yang X, Guo Y, Liu Y (2011) Bayesian-inference based recommendation in online social networks. IEEE, Infocom

Zan Huang HC, Zeng Daniel (2004) Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. ACM Trans Inf Syst 22:116–142

Zhou T, Ren J, Medo M, Zhang Y-C (2007) Bipartite network projection and personal recommendation. Phys Rev E 76:046115

Zhou T, Kuscsik Z, Liu J-G, Medo M, Wakeling JR et al (2010) Solving the apparent diversity-accuracy dilemma of recommender systems. Proc Natl Acad Sci USA 107:4511–4515