A Simple but Effective Method to Incorporate Trusted Neighbors in Recommender Systems

Guibing Guo *, Jie Zhang, and Daniel Thalmann

School of Computer Engineering
Nanyang Technological University, Singapore
{gguo1,zhangj,danielthalmann}@ntu.edu.sg

Abstract. Providing high quality recommendations is important for online systems to assist users who face a vast number of choices in making effective selection decisions. *Collaborative filtering* is a widely accepted technique to provide recommendations based on ratings of similar users. But it suffers from several issues like *data sparsity* and *cold start*. To address these issues, in this paper, we propose a simple but effective method, namely "Merge", to incorporate social trust information (i.e. trusted neighbors explicitly specified by users) in providing recommendations. More specifically, ratings of a user's trusted neighbors are merged to represent the preference of the user and to find similar other users for generating recommendations. Experimental results based on three real data sets demonstrate that our method is more effective than other approaches, both in accuracy and coverage of recommendations.

1 Introduction

Recommender systems are heavily used in e-commerce to provide users with high quality, personalized recommendations to help them find satisfactory items (e.g. books, movies, news, music, etc.) among a huge number of available choices. Collaborative filtering (CF) [7] is the most commonly used technique to generate recommendations. The heuristic is that the items appreciated by those who have similar taste will also be appreciated by the active user (the user who needs recommendations). However, CF suffers from several inherent drawbacks like *data sparsity* and *cold start*. Data sparsity arises due to the fact that users in general only rate a small portion of items. Cold start refers to the dilemma that accurate recommendations are expected for new users whereas they often rate only a few items that are difficult to reveal their preferences.

To mitigate the problems suffered by CF, trust-aware recommender systems (TARSs) have been proposed to incorporate social trust information (i.e. trusted neighbors of users) [2, 5]. For example, Massa et al. [5] suggest that trust information is more meaningful to bootstrap recommender systems than item-rating information. Both implicit trust (e.g. [6, 9]) and explicit trust (e.g. [2, 5, 1, 8])

* Would like to thank the Institute for Media Innovation (IMI) at Nanyang Technological University (NTU) for providing a PhD grant.

have been utilized in the literature whereas explicit trust is more accurate than the implicit one. Although the overall performance of recommendation can be improved to some extent by the trust-aware recommender systems [12], the mitigation for the cold-start problem is still limited [10].

In this paper, we propose a simple but effective method called "Merge" to incorporate trusted neighbors explicitly specified by users in recommender systems to improve the overall performance of recommendation and mitigate the cold-start problem. Specifically, we merge the ratings of an active user's directly trusted neighbors by averaging the neighbors' ratings for their commonly rated items according to how much the neighbors are trusted by the active user. The merged rating set is then used to represent the active user's preference and find similar other users for the active user. Finally, the ratings provided by both the similar users and the trusted neighbors are used to predict item ratings for the active user. Experiments on three real data sets are conducted to verify the effectiveness of our method. The results show that it can achieve promising accuracy and coverage for recommendation, and is especially useful for cold-start users, compared with other approaches. Our method thus shades light on incorporating trusted neighbors for building an effective trust-aware recommender system.

2 Related Work

Trust has been extensively studied in recommender systems, that is trust-aware recommender systems. The intuition is that trusted users may share similar taste. In fact, researchers have found that trust has a positive and strong correlation with preference [11].

O'Donovan et al. [6] indicate that trust is useful to decrease recommendation error. They define *profile-level* and *item-level* trust as the percentage of correct predictions from the view of general profile and specific items, respectively. In our work, we focus on explicit trust relations as they are directly specified by users and more accurate than implicit ones. Jamali and Ester [3] design the *TrustWalker* approach to randomly select neighbors in the trust network formed by users and their trusted neighbors. Trust information of the selected neighbors is combined with an item-based technique to predict item ratings. On the contrary, our work focuses on generating predictions by combining trust information with a user-based technique. Liu and Lee [4] report that more accurate prediction algorithms are possible by incorporating trust information into traditional collaborative filtering. They do not directly use trust to substitute similarity but rather amplify similarity measurement by taking into account the number of messages exchanged among users. Thus this approach is message specific.

The closest approaches to ours are as follows. Massa and Avesani [5] analyze the drawbacks of CF-based recommender systems and describe how and why trust can mitigate those problems. They propose *MoleTrust* [5], which performs depth-first search, to propagate and infer trust in the trust network. Empirical results show that the coverage is significantly enlarged but the accuracy remains comparable when propagating trust. Besides, Golbeck [2] proposes a breadth-first search method *TidalTrust* to infer and compute trust value, but the performance of them is close [12]. Hence, we only consider MoleTrust for comparison in this paper. Chowdhury et al. [1] propose to enhance CF by predicting the ratings of similar users who did not rate the concerned items according to the ratings of their trusted neighbors, so as to incorporate more users for recommendation. However, it performs badly for cold-start users, which is the main concern of this work. Another recent work using the trust network is proposed by Ray and Mahanti [8]. They improve the prediction accuracy by reconstructing the trust network. More specifically, they remove the trust links between two users if their correlation is lower than a threshold. Empirical results show that good performance is achieved at the cost of poor coverage.

In addition, although many trust-aware recommender systems have been proposed to exploit explicit trust for effective recommendations, most of them are evaluated on only one data set. These approaches often achieve improvements in either accuracy or coverage, but not both. More importantly, the cold-start problem has not been well addressed yet. Therefore, how to incorporate trust information for effective recommendations remains a big challenge [10]. The purpose of our work is to take a step further in addressing this challenge by proposing a simple but effective method to incorporate trusted neighbors in TARSs.

3 The Merge Method

Our Merge method incorporates trusted neighbors of an active user for recommendations by taking the following three steps: 1) merging the ratings of trusted neighbors to represent the preference of the active user; 2) finding similar users according to the merged rating set; and 3) predicting the ratings of items for the active user based on the ratings for the items provided by the similar users and trusted neighbors. The detailed and formal description as well as the insights of the Merge method are given in the subsequent sections.

3.1 Merging the Ratings of Trusted Neighbors

Let U and I denote the sets of all users and items in the system, respectively. Let $r_{v,i}$ be the rating of an item $i \in I$ provided by a user v. For an active user $u \in U$ who has not rated an item $j \in I$, the task is to predict a rating for the item j that the active user u will likely provide, denoted by $\hat{r}_{u,j}$.

In the system, the active user u has identified a set of trusted neighbors TN_u. For a trusted neighbor $v \in TN_u$, user u also specifies a trust value $t_{u,v}$ indicating the degree to which user u trusts user v. We assume that the active user u should fully trust herself because the ratings of items provided by herself should accurately represent her own preference on the rated items. Thus, user u herself is also included in the set TN_u of her trusted neighbors, and $t_{u,u} = 1$ if the highest possible degree of trust is 1.

For an item $i \in I$ that is rated by at least one trusted neighbor in TN_u, we merge the ratings of item i provided by the trusted neighbor(s). More specifically, we average the ratings according to the trust values of the trusted neighbors specified by the active user u, as follows:

$$\tilde{r}_{u,i} = \frac{\sum_{v \in TN_u} t_{u,v} r_{v,i}}{\sum_{v \in TN_u} t_{u,v}} \tag{1}$$

where $\tilde{r}_{u,i}$ is the merged rating for the active user u on item i, according to the ratings of her trusted neighborhood TN_u (including herself).

We perform the process of merging ratings for every item in I that is rated by at least one trusted neighbor in TN_u. We denote the set of such items as $\tilde{I}_u$. In the end, we have a set of merged ratings, each of which is for an item in $\tilde{I}_u$. This merged rating set is used to represent the preference of the active user u.

3.2 Incorporating with Collaborative Filtering

Given the merged rating set on the items in $\tilde{I}_u$, which represents the preference of the active user u, we then apply the collaborative filtering technique to predict the rating of the item j that is not rated by u. More specifically, we first find a set of similar users (i.e. a set of nearest neighbors denoted as NN_u) for the active user u based on the merged rating set. The rating of item j is then predicted by aggregating the ratings for the item j provided by the nearest neighbors in NN_u and the trusted neighbors in TN_u.

For finding a set of similar users for the active user u, we adopt the popular Pearson Correlation Coefficient (PCC) to compute the similarity between user u and another user v who is not in TN_u, as follows:

$$s_{u,v} = \frac{\sum_{i \in I_{u,v}} (\tilde{r}_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum_{i \in I_{u,v}} (\tilde{r}_{u,i} - \bar{r}_u)^2} \sqrt{\sum_{i \in I_{u,v}} (r_{v,i} - \bar{r}_v)^2}} \tag{2}$$

where $I_{u,v} \subseteq \tilde{I}_u$ is the set of the items in $\tilde{I}_u$ that are also rated by user v, $\tilde{r}_{u,i}$ is the merged rating for the active user u on item i calculated using Equation 1, $\bar{r}_u$ is the average of the merged ratings for the active user u on the items in $\tilde{I}_u$, and $\bar{r}_v$ is the average of the ratings of all the items rated by user v.

A group of similar users, or *nearest neighbors*, is then selected as follows:

$$NN_u = \{v | s_{u,v} > \theta, v \in U\} \tag{3}$$

where θ is a predefined similarity threshold, and NN_u denotes the nearest neighborhood of the active user u.

Finally, the predicted rating $\hat{r}_{u,j}$ of item j for the active user u is generated by aggregating the ratings of item j provided by the nearest neighbors in NN_u and the trusted neighbors in TN_u weighted by their similarity values and trust values respectively, as follows:

$$\hat{r}_{u,j} = \frac{\sum_{v \in NN_u} s_{u,v} r_{v,j} + \sum_{v \in TN_u} t_{u,v} r_{v,j}}{\sum_{v \in NN_u} s_{u,v} + \sum_{v \in TN_u} t_{u,v}} \tag{4}$$

The neighbors who have larger similarity with the active user u or are trusted more by user u will have higher impact on the predicted rating.

3.3 The Insights of the Merge Method

One common characteristic of the *data sparsity* and *cold-start* problems is that the small number of commonly rated items between users makes it difficult to accurately compute user similarity and hence difficult to find effective nearest neighbors for the active users. In many cases, there is even no commonly rated items between two users because of data sparsity, causing their similarity not computable. In our method, we merge the ratings of the active user u's trusted neighbors to represent the preference of user u. Since the merged rating set usually covers a larger number of items than the active user u's own rating set (i.e. $|\tilde{I}_u| > |I_u|$), the number of the items in $\tilde{I}_u$ that are also rated by another user v, which is $|I_{u,v}|$, is also likely to be larger. This is especially true for cold-start users who have not rated many items yet. As a result, the similarity between a larger number of users can be computed accurately. In this way, our method mitigates the data sparsity and cold-start problems.

Many trust-based approaches (for example, the MoleTrust algorithm in [5] and the approach proposed in [8]) predict ratings for items based only on the ratings provided by the trusted neighbors. In contrast, our Merge method not only makes use of the ratings provided by the trusted neighbors, but also considers the ratings of similar users (NN_u) found based on the merged rating set of trusted neighbors (see Equation 4). Thus, the number of neighbors used for rating prediction is certainly larger in our method, resulting in the improvement in both accuracy and coverage of rating prediction that will be confirmed by the experimental results in Section 4.3.

Due to relying only on the ratings provided by the trusted neighbors for rating prediction, the trust-based approaches may also suffer from the similar cold-start problem where some users may only specify a small number of other users as their trusted neighbors. This issue could be a common case for many social systems, especially when users are lack of incentives to be pro-active. Thus, the performance is limited since only a few neighbors can be incorporated for recommendation. Our Merge method addresses this problem by also making use of the ratings of the active user u herself if any. In particular, the active user u is considered as a fully trustworthy neighbor to herself when merging the ratings of trusted neighbors. When user u has no trusted neighbors but rated a certain number of items, the merged rating set will be the same as her own rating set because the only trusted neighbor is herself. The whole procedure will be exactly the same as the traditional collaborative filtering technique. In this way, our method is competent to mitigate the cold-start problem.

To cope with the cold-start problem for trusted neighbors, some work (e.g. [6]) also proposes approaches to infer implicit trust from users' rating profiles. However, implicit trust is not as accurate as explicit trust that is directly specified by users. Trust propagation [5] has also been widely used to cope with the cold-start problem by inferring the trust between two users based on the trust network formed by any available trusted neighborhood relationships. However, it has several shortcomings: 1) the best propagation length is difficult to be determined for different networks; 2) trust propagation makes it possible to incorporate less valuable users, especially when the propagation length is long, and hence may decrease the prediction accuracy; 3) it is often costly and time-consuming to propagate trust, especially when the trust network is dense. Our method makes use of only direct trusted neighbors. We will also show in Section 4.3 that trust propagation does not bring any benefit to our method.

4 Experimental Validation

In order to verify the effectiveness of the Merge method, we conduct experiments on three real data sets. We aim to find out: 1) how the performance of our Merge method is in comparison with other approaches; 2) whether it is effective to propagate trust for our method; and 3) how the performance changes when tuning the similarity threshold θ in Equation 3.

4.1 Data Acquisition

Three real data sets are used in our experiments, including FilmTrust[1], Flixster[2] and Epinions[3]. FilmTrust is a trust-based social site in which users can rate and review movies. Since there is no publicly downloadable data set, we crawled one in June 2011, collecting 1,986 users, 2,071 movies and 35,497 ratings (scaled from 0.5 to 4.0 with step 0.5). Besides, it also contains 1,853 trust ratings that are issued by 609 users. The trust ratings in FilmTrust are binary where 1 means "trust" and 0 otherwise. Flixster is also a social movie site in which users can make friends and share their movie ratings. The original data set[4] is very large. For the simplicity, we sample a subset by randomly choosing 53K users who issued 410K item ratings (scaled from 0.5 to 4.0 with step 0.5) and 655K trust ratings. The trust ratings in Flixster are scaled from 1 to 10 but not available in the data set. We assign the trust value 1 to a user who is identified as a trusted neighbor, and 0 otherwise. Epinions is a website in which consumers can express their opinions by assigning numerical ratings to items. The data set[5] is generated by Massa and Avesani [5], consisting of 49K users who issued 664K ratings (scaled from 1 to 5 with step 1) over 139K different items and 478K trust ratings. The trust ratings in Epinions are also binary (either 1 or 0).

4.2 Experimental Settings

In our experiments, we compare Merge with the following approaches:

- **TrustAll** simply trusts every user and predicts a rating for an item by averaging all ratings of those who have rated the item.

[1] http://trust.mindswap.org/FilmTrust/
[2] http://www.flixster.com/
[3] http://www.epinions.com/
[4] http://www.cs.sfu.ca/~sja25/personal/datasets/
[5] http://www.trustlet.org/datasets/downloaded_epinions

- **CF** computes user similarity using the PCC measure, selects the users whose similarity is above the threshold θ, and uses their ratings for prediction.
- **MT**x ($x = 1, 2, 3$) are the implementations of the MoleTrust algorithm [5] in which trust is propagated in the trust network with the length x. Only trusted neighbors are used to predict ratings for items.
- **RN** denotes the approach proposed in [8] that predicts item ratings by reconstructing trust network. We adopt their best performance settings where the correlation threshold is 0.5, propagation length is 1, and the top 5 users with highest correlations are selected for rating prediction.
- **TCF2** denotes the approach proposed in [1] that enhances CF by predicting the ratings of the similar users who did not rate the items according to the ratings of the similar users' trusted neighbors, so as to incorporate more users for recommendation. In [1], the best performance is achieved when trust propagation length is 2. We adopt the same setting in our experiments.
- **Merge2** is a variation of the Merge method where the trust propagation length is 2, to also incorporate the trusted neighbors of the trusted neighbors. The purpose is to investigate the impact of trust propagation.

In addition, we split each data set into different views in the light of user-related or item-related properties as defined in [5]:

- **All** represents the whole data set.
- **Cold Users** are those who rated no more than 5 items.
- **Heavy Users** are those who rated more than 10 items.
- **Opinionated Users** are those who rated more than 4 ratings, and the standard deviation of the ratings is greater than 1.5.
- **Black Sheep** rated more than 4 ratings, and the average difference between their average rating and the mean rating of each item is greater than 1.
- **Controversial Items** are those which received ratings with standard deviation greater than 1.5.
- **Niche Items** are those which received less than 5 ratings.

We focus on the performance in the views of **All** and **Cold Users**, which indicate the effectiveness to mitigate the data sparsity and cold-start problems.

The evaluation is proceeded by applying the *leave-one-out* technique [5] on every user rating. The results are analyzed according to the performance in terms of accuracy and coverage. In particular, the predictive accuracy is evaluated using *Mean Absolute Error* (MAE), the degree to which a predicted rating is close to the ground truth. *Rating coverage* (RC) is measured as the percentage of all items that are predictable.

4.3 The Performance of the Merge Method

In this set of experiments, we evaluate the performance of our Merge method, in comparison with the other approaches presented in the previous section. We fix the similarity threshold θ to be 0. Tables 1, 2 and 3 summarize the results on the FilmTrust, Flixster and Epinions data sets, respectively.

We obtain very close results on the Epinions data set in Table 3 as those in [5] and [1]. The similar trends of results are also obtained on the other two data sets, as shown in Tables 1 and 2. From all these results, we can see that our Merge method achieves consistent and better performance both in accuracy and coverage whereas other approaches expose their limitations in either accuracy or coverage. More specifically, CF results in benchmark performance and large diversity across three data sets, which can be explained by [7] that its effectiveness is heavily associated with the distributions of ratings of similar users. The trust-based approaches (MTx) are able to increase rating coverage to a large extent, but the accuracy is quite low. The RN method accomplishes good accuracy but covers the smallest portion of items since only the ratings of the users who have a large number of trusted neighbors and high rating correlation with others are possible to be predicted. Although TCF2 achieves relatively good results and improves both accuracy and coverage over CF, RN and MTx, its performance varies on different data sets. Comparing with TCF2, the accuracy of our Merge method is similar on Epinions but much better on FilmTrust and Flixster, and the coverage of our method is much better on Flixster but worse on Epinions. Therefore, we can conclude that in general our Merge method outperforms the other approaches. It consistently achieves high accuracy and large coverage, demonstrating its effectiveness in mitigating the data sparsity problem.

Table 1. The Performance on FilmTrust

MAE/RC									
Views	Approaches								
	CF	MT1	MT2	MT3	TrustAll	RN	TCF2	Merge	Merge2
All	0.703 93.83%	0.852 21.20%	0.795 27.95%	0.771 30.38%	0.726 98.17%	0.571 0.74%	0.683 96.85%	0.612 95.36%	0.624 95.52%
Cold Users	0.744 39.64%	0.853 17.11%	0.880 23.19%	0.819 23.85%	0.753 98.19%	NaN 0.00%	0.740 41.12%	0.604 68.91%	0.634 69.90%
Heavy Users	0.705 94.95%	0.854 21.53%	0.797 28.25%	0.772 30.75%	0.728 98.13%	0.571 0.80%	0.684 98.06%	0.617 95.82%	0.628 95.97%
Opin. Users	1.469 87.63%	1.268 14.43%	1.156 15.46%	1.194 15.46%	1.105 94.85%	NaN 0.00%	1.405 91.75%	1.210 93.81%	1.213 93.81%
Black Sheep	1.237 90.63%	1.228 19.94%	1.243 24.82%	1.269 26.13%	1.255 99.86%	NaN 0.00%	1.244 92.22%	1.130 90.94%	1.140 90.98%
Contr. Items	2.106 62.58%	2.358 16.04%	2.418 21.38%	2.265 27.36%	2.380 100.0%	0.500 0.31%	1.482 89.31%	1.947 66.35%	2.056 71.38%
Niche Items	0.986 53.92%	1.031 14.04%	1.011 19.35%	0.962 25.36%	1.009 79.51%	0.485 0.66%	0.574 85.17%	0.915 61.67%	0.940 63.44%

More importantly, none of previous approaches works well in the view of *Cold Users*. CF covers very limited percentage of items (around 3% in Flixster and Epinions) with very poor accuracy. MTx methods can alleviate this problem relative to CF in these two data sets. However, it performs worse than CF in FilmTrust because the performance of MTx depends on the number of trusted neighbors and this value is very small in FilmTrust (around 3 trusted neighbors

Table 2. The Performance on Flixster

MAE/RC									
Views	Approaches								
	CF	MT1	MT2	MT3	TrustAll	RN	TCF2	Merge	Merge2
All	0.928 68.56%	1.060 12.36%	0.932 71.37%	0.862 90.71%	0.855 98.11%	0.858 0.38%	0.811 86.82%	0.664 94.19%	0.776 95.86%
Cold Users	1.153 3.27%	1.127 8.11%	1.005 52.69%	0.934 79.55%	0.918 99.03%	NaN 0.00%	0.930 21.42%	0.723 82.73%	0.784 88.75%
Heavy Users	0.913 85.59%	1.046 13.29%	0.917 75.55%	0.846 93.29%	0.839 97.70%	0.858 0.52%	0.797 98.74%	0.654 95.92%	0.776 96.83%
Opin. Users	1.494 74.80%	1.574 12.65%	1.487 72.37%	1.457 92.50%	1.447 99.23%	1.095 0.55%	1.419 98.61%	1.098 98.03%	1.272 98.65%
Black Sheep	1.320 76.21%	1.300 13.59%	1.288 75.53%	1.273 93.46%	1.279 99.42%	1.258 0.23%	1.248 94.64%	0.977 97.92%	1.145 98.47%
Contr. Items	1.830 30.64%	1.847 2.33%	1.833 27.63%	1.873 76.94%	1.951 100.0%	1.167 0.10%	1.373 85.00%	1.549 68.68%	1.709 82.15%
Niche Items	1.068 11.77%	1.195 0.66%	1.021 11.23%	1.057 43.73%	1.073 61.60%	1.400 0.02%	0.409 81.42%	1.016 35.01%	1.029 46.10%

Table 3. The Performance on Epinions

MAE/RC									
Views	Approaches								
	CF	MT1	MT2	MT3	TrustAll	RN	TCF2	Merge	Merge2
All	0.876 51.24%	0.845 26.34%	0.852 57.64%	0.832 71.68%	0.821 88.20%	0.673 9.87%	0.691 87.46%	0.708 77.94%	0.775 81.87%
Cold Users	1.032 3.22%	0.756 6.57%	0.916 22.06%	0.890 41.73%	0.857 92.92%	NaN 0.00%	0.936 10.52%	0.670 47.22%	0.738 57.56%
Heavy Users	0.873 57.41%	0.847 29.28%	0.848 62.40%	0.827 75.36%	0.818 87.50%	0.673 11.48%	0.677 95.24%	0.713 80.95%	0.780 84.07%
Opin. Users	1.120 49.99%	1.060 19.99%	1.124 52.02%	1.110 68.79%	1.105 92.80%	0.774 5.34%	1.022 86.79%	0.879 80.77%	0.990 85.09%
Black Sheep	1.246 55.72%	1.199 20.06%	1.259 53.73%	1.252 70.98%	1.255 97.03%	0.852 4.50%	1.205 89.85%	0.989 85.67%	1.123 89.57%
Contr. Items	1.598 45.40%	1.481 22.87%	1.646 57.81%	1.707 78.19%	1.741 100.0%	0.953 7.47%	1.389 86.15%	1.326 81.19%	1.553 88.91%
Niche Items	0.835 12.16%	0.743 7.84%	0.811 23.65%	0.829 39.37%	0.829 55.39%	0.598 2.14%	0.282 79.81%	0.775 37.42%	0.802 46.17%

per user on average). The MAE value of the RN method is NaN (not-a-number), meaning that it is unable to predict any rating. This is because only the users who have at least 4 commonly rated items with others will be kept in the trust network [8]. This is conflicting with the setting for cold users. TCF2 also covers only a limited range of items because it depends on the number of similar users, which is very small for cold users. This limitation also causes low accuracy. These results confirm that the cold-start problem remains a big challenge for recommender systems. Both CF and previous trust-aware methods cannot achieve good accuracy or coverage and even perform worse than TrustAll. On the contrary, our Merge method is especially effective for the cold users. The improvement in accuracy reaches up to 18.82%, 37.29% and 35.08% relative to CF, and 18.38%, 22.26% and 28.42% relative to TCF2 according to the results in Tables 1, 2 and 3, respectively. The amount of increment in coverage is even larger. This is because after merging, even the rating set of the cold users could cover a large number of items, hence they can find many similar users.

We also compare the Merge method with its variation Merge2 where the trust propagation length is 2. We find that the increment in coverage is very limited but the accuracy is much worse. The reason is that although propagating trust is able to incorporate more neighbors to represent user's preference, it does not guarantee that the merged rating set will cover more items than the one without propagation, especially when the active users initially have many trusted neighbors. In addition, propagation will increase the possibility to incorporate less valuable users which may decrease the accuracy. Furthermore, the Merge method has already covered a good range of items hence it is not necessary to propagate trust. Overall, trust propagation is not necessary for our method.

4.4 The Effect of the Similarity Threshold θ

The similarity threshold θ plays an important role in CF-based methods. It is used to select a group of similar users as recommenders for rating prediction (see Equation 3). Intuitively, when the similarity threshold is set high, a smaller number of less similar users (unreliable recommenders) will be selected. The prediction accuracy should be better, but the coverage may decrease. Therefore, to explore the effect of the similarity threshold, we tune the θ value from 0 to 0.9 with step 0.1. The results are illustrated in Figures 1, 2 and 3.

Surprisingly, the results show that the accuracy of CF does not increase as expected, rather, it becomes worse with the increment of θ. We attribute this counter-intuitive phenomenon to the overestimation problem of the PCC similarity measure. That is, when the number of commonly rated items between users is small, the computed PCC value tends to be high, which makes it difficult to distinguish reliable users from unreliable ones via the similarity threshold. Although RN achieves good accuracy when $\theta \geq 0.6$ on FilmTrust and Epinions, the results are not representative because it covers too few items (less than 10%). The performance of TCF2 is not much affected by the similarity threshold.

Our method works in the way as expected. With the increment of θ, the accuracy first goes up and then drops down. More specifically, the best accuracy for our method can be achieved when the similarity threshold is set to be 0.4, 0.6 and 0.8 for FilmTrust, Flixster and Epinions, respectively. Besides, the amount of increment in accuracy is around 4.81%, 2.61% and 8.33% for cold users for three data sets, respectively, comparing with the case where the similarity threshold is 0. In general, when the similarity threshold θ is set to be $0.4 \sim 0.8$, better performance can be achieved for our method, and at the same time, its coverage does not decrease much. In addition, by tuning the similarity threshold, the Merge method significantly outperforms TCF2 on all data sets.

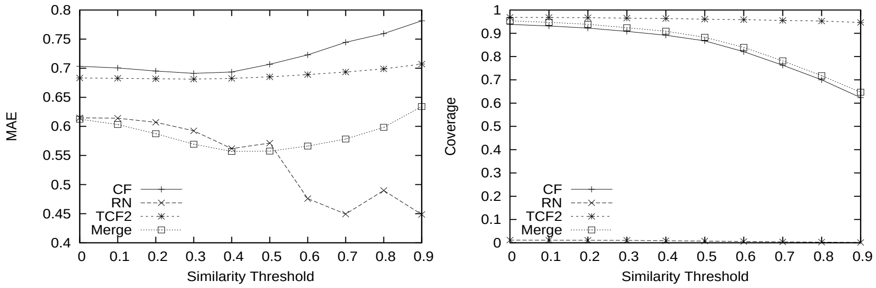

Fig. 1. The Effect of Similarity Threshold on FilmTrust

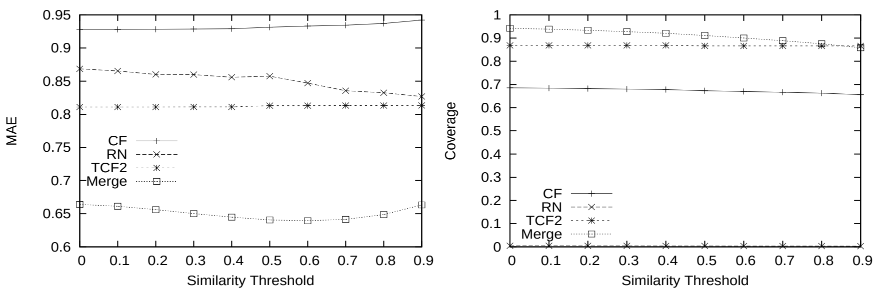

Fig. 2. The Effect of Similarity Threshold on Flixster

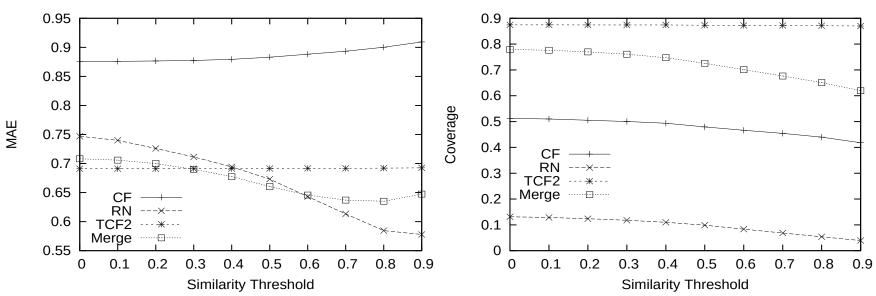

Fig. 3. The Effect of Similarity Threshold on Epinions

5 Conclusion and Future Work

Aiming to overcome the data sparsity and cold-start problems for recommender systems, we proposed a simple but effective method to incorporate trusted neighbors that are directly specified by users. The ratings of trusted neighbors are merged to represent the preference of the active user, based on which we then find similar users and generate recommendations. We conducted experiments on three real data sets and the results show significant improvement against other methods both in accuracy and coverage. We also demonstrated that it is not necessary for our method to propagate trust since incorporating direct trusted neighbors works well enough. Furthermore, by tuning the similarity threshold, better performance can be achieved for our method.

Our Merge method merges the ratings of trusted neighbors by a weighted average strategy (see Equation 1), which is shown effective in the scenario of sparse distribution of ratings. However, for the items that receive many ratings from trusted neighbors, the majority strategy that assigns the majority as the merged rating may work better, especially when the ratings are diverse (i.e. the standard deviation is large). For future work, we will investigate how the majority strategy can possibly improve the performance of our method.

References

1. Chowdhury, M., Thomo, A., Wadge, B.: Trust-based infinitesimals for enhanced collaborative filtering. In: Proceedings of the 15th International Conference on Management of Data (COMAD) (2009)
2. Golbeck, J.: Computing and applying trust in web-based social networks. Ph.D. thesis (2005)
3. Jamali, M., Ester, M.: Trustwalker: a random walk model for combining trust-based and item-based recommendation. In: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 397–406
4. Liu, F., Lee, H.: Use of social network information to enhance collaborative filtering performance. Expert Systems with Applications 37(7), 4772–4778 (2010)
5. Massa, P., Avesani, P.: Trust-aware recommender systems. In: Proceedings of the ACM Conference on Recommender Systems. pp. 17–24 (2007)
6. O'Donovan, J., Smyth, B.: Trust in recommender systems. In: Proceedings of the 10th International Conference on Intelligent User Interfaces (IUI). pp. 167–174
7. Rafter, R., O'Mahony, M.P., Hurley, N.J., Smyth, B.: What have the neighbours ever done for us? a collaborative filteringperspective. In: Proceedings of the 17th International Conference on User Modeling, Adaptation, and Personalization (UMAP). pp. 355–360 (2009)
8. Ray, S., Mahanti, A.: Improving prediction accuracy in trust-aware recommender systems. In: Proceedings of the 43rd Hawaii International Conference on System Sciences (HICSS). pp. 1–9 (2010)
9. Seth, A., Zhang, J., Cohen, R.: Bayesian credibility modeling for personalized recommendation in participatory media. In: Proceedings of the 18th International Conference on User Modeling, Adaptation, and Personalization (UMAP). pp. 279–290 (2010)
10. Shi, Y., Larson, M., Hanjalic, A.: How far are we in trust-aware recommendation? In: Proceedings of the 33rd European Conference on Advances in Information Retrieval (ECIR). pp. 704–707. Springer (2011)
11. Singla, P., Richardson, M.: Yes, there is a correlation: from social networks to personal behavior on the web. In: Proceedings of the 17th International Conference on World Wide Web. pp. 655–664 (2008)
12. Victor, P., Cornelis, C., De Cock, M., Teredesai, A.: Trust- and distrust-based recommendations for controversial reviews. In: Proceedings of the WebSci'09: Society On-Line. No. 161 (2009)

Error Correction for “A Simple but Effective Method to Incorporate Trusted Neighbors in Recommender Systems”

Guibing Guo, Jie Zhang, and Daniel Thalmann

School of Computer Engineering
Nanyang Technological University, Singapore
{gguo1,zhangj,danielthalmann}@ntu.edu.sg

Abstract. This report is to correct errors for the paper “A Simple but Effective Method to Incorporate Trusted Neighbors” published in the proceedings of the 20th International Conference on User Modeling, Adaptation and Personalization (UMAP 2012). We found that the presented results in the paper are invalid due to some experimental errors. Those errors are summarized and corrected in this report and new experimental results are presented. However, the essential principle to merge the ratings of trusted neighbors to form a new rating profile for the active users is unchanged at all. In addition, the proposed method “Merge” is yet demonstrated to be more effective in solving cold start and data sparsity problems than other counterparts both in accuracy and coverage. However, a noted different conclusion is that better performance can be achieved for the Merge method when trust is propagated along the web of trust.

1 Overview of the Proposed Method

In the paper “A Simple but Effective Method to Incorporate Trusted Neighbors” [1], we proposed a simple method, called “Merge”, which can effectively incorporate trusted neighbors to resolve data sparsity and cold start problems in recommender systems. This method can be generally summarized in two steps:

1. For an active user u, merge the ratings of her trusted neighbors TN_u to form a new rating profile to represent user u’s preference. Specifically, new ratings $\tilde{r}_{u,i}$ can be generated by

$$\tilde{r}_{u,i} = \frac{\sum_{v \in TN_u} w_{u,v} r_{v,i}}{\sum_{v \in TN_u} w_{u,v}} \tag{1}$$

   where $w_{u,v}$ is the weight of trusted neighbor v from the view of user u. In [1], the trust value is used, i.e. $w_{u,v} = t_{u,v}$.
2. Based on the newly formed rating profile, traditional collaborative filtering (CF) technique is conducted to make predictions by taking into consideration

trusted neighbors TN_u as well as similar neighbors NN_u.

$$\hat{r}_{u,j} = \frac{\sum_{v \in NN_u} s_{u,v} r_{v,j} + \sum_{v \in TN_u} t_{u,v} r_{v,j}}{\sum_{v \in NN_u} s_{u,v} + \sum_{v \in TN_u} t_{u,v}} \tag{2}$$

where $s_{u,v}$ is the similarity score between users u and v which is calculated by Pearson Correlation Coefficient (PCC) [2].

We demonstrated the effectiveness of the "Merge" method based on three real-life datasets, showing that this simple method can achieve significant improvement in comparison with other counterparts.

2 The Errors

However, some accidental errors have been discovered regarding methods "Merge" and "TCF2" after we carefully double checked the experimental implementations for all methods. The main issue is that we did not properly remove test ratings as required by the leave-one-out validation method.

- **Merge**: with the original intention to save running time, for all the test ratings reported by the same active user, we merged all the ratings of trusted neighbors. As the active user herself is regarded as one of her trusted neighbors, all her ratings including the test ratings are involved in the merging process. However, for the leave-one-out method, we should have left one rating out as the test rating and then merge the rest of the ratings of trusted neighbors.
- **TCF2**: the number "2" represents two computative iterations over trust network when predicting the missing ratings for the similar users rather than the length of trust propagation depicted in the previous paper. Originally, we implemented two TCF variants: TCF1 and TCF2. Although we achieved approximate results compared to [3] for TCF2, we noted that our version of TCF1 performed much worse than TCF2. However, as reported in [3], TCF1 should be close to TCF2 in terms of performance. After carefully checking, we discovered that in our previous implementation, TCF1 removed test ratings whereas TCF2 did not.

3 The New Merge Method

Firstly, it is important to highlight that the essential principle for our method is not changed: merging the ratings of trusted neighbors to form a new rating profile for the active users. Secondly, we are not simply presenting new results after correcting the errors we found, but also proposing some improvements so as to perfect the original method as continuous work. Last but not least, the major conclusion that our method outperforms the others will not be changed at all.

3.1 The Issues of the Original Merge Method

Although the original Merge method is quite simple, several issues may be emerging from Equations 1 and 2.

1. Although it is reported that generally trust has a high and positive correlation with similarity [4], the similarity between the active users and their trusted neighbors is not guaranteed to be high all the time. Rather, it is possible that trusted users have low similarity. From [5], we also note that trusted users with high similarity have a positive influence on the predictive accuracy after eliminating those with low similarities.
2. For a specific item, all ratings of trusted neighbors on that item will be merged, even if it has already been rated by the active user. In other words, the ratings of active users may be different and changed after the merging process. This situation is not desired as we assume that an active user will always trust herself for giving accurate ratings.
3. For a specific item that has not been rated by the active users, there are in general only a few trusted neighbors who have provided a rating for it. In fact, in most of the cases, there is only one trusted neighbor available. This can be explained by the data sparsity which leads to low possibility for users to rate many items in common. Adopting many ratings of the items that are only rated by few trusted neighbors would create much noise to the new rating profile that we aim to form for the active users. In addition, the main purpose of merging process is to approximately model user preference for the cold users. Although it may have some benefits for the other users who rated more items, the need to merge the ratings of trusted neighbors is not that strong or necessary, especially for the heavy users and considering that errors are possibly raised during that process. In the original Merge, this was not taken into account.
4. As analyzed, the merging process may raise some errors due to too few ratings available on the concerned items. Continuing to involve trust information in Equation 2 may accumulate these errors from trusted neighbors and ruin the predictive accuracy in the end.

3.2 The New Merge Method

In order to solve the aforementioned issues in Section 3.1, we make a few modifications to Equations 1 and 2.

1. Only trusted neighbors with high similarity are adopted as candidates for merging. That is, TN_u in Equation 1 is replaced by

$$TN'_u = \{v | s_{u,v} > \theta, v \in TN_u\} \tag{3}$$

   where θ is the similarity threshold. Accordingly, the weight in Equation 1 is changed by taking into consideration both trust and similarity values. Specifically, harmonic mean [6] is used to integrate them.

$$w_{u,v} = \frac{2 \cdot t_{u,v} \cdot s_{u,v}}{t_{u,v} + s_{u,v}} \tag{4}$$

The advantage of harmonic mean is that high value can be calculated only when both trust and similarity values are high.

2. All the ratings of the active user u will be retained and kept unchanged. Only the ratings of trusted neighbors on the other items that user u has not rated will be merged. Therefore, we restrict the range of i in Equation 1 as

$$i \in I'_u = \{j | r_{v,j} > 0 \cap r_{u,j} = 0, v \in TN'_u\} \tag{5}$$

3. Instead of incorporating all possible ratings from trusted neighbors with high risk of involving much noise, we tend to adopt ratings with high confidence. The confidence measures the degree of certainty that the merged rating is useful. Specifically, it mainly considers two aspects: the number of ratings and the conflicts between positive and negative ratings. The confidence $c_{u,i}$ of a merged rating $\tilde{r}_{u,i}$ is defined in the evidence space $< r, s >$ (refers to [7]):

$$c_{u,i} = c(r,s) = \frac{1}{2}\int_0^1 |\frac{x^r(1-x)^s}{\int_0^1 x^r(1-x)^s d_x} - 1| d_x \tag{6}$$

where $c(r, s) \in (0, 1)$, r and s refer to the number of positive, negative evidences (ratings), respectively. We define a rating $r_{v,i}$ as positive or negative by:

$$\begin{cases} r_{v,i} \text{ is positive : if } r_{v,i} > r_{med} \\ r_{v,i} \text{ is negative : otherwise} \end{cases} \tag{7}$$

where r_{med} is the median rating scale in the range from the minimum rating scale r_{min} to the maximum rating scale r_{max}. The fewer ratings issued on the same item or the higher conflicts between positive and negative ratings, the smaller the confidence value is. Hence the minimum value of the confidence is $c(1, 1) = 0.19$ with the minimum evidences[1] and highest conflicts. The confidences above 0.19 of all merged ratings are ranked and only the most k confident ratings are adopted. Note that the top k ratings include the ratings of the active user herself whose rating confidence is always 1.0. In this way, for the cold start users who rated only few items, more ratings are adopted from the merging process whereas for the heavy users who rated many items, less ratings are adopted.

4. Since the merged ratings have different confidences inherently, we propose a variant of PCC to compute user similarity (refers to Equation 2) by taking into consideration rating confidence,

$$s_{u,v} = \frac{\sum c_{u,i}(\tilde{r}_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r})}{\sqrt{\sum c^2_{u,i}(\tilde{r}_{u,i} - \bar{r}_u)^2}\sqrt{\sum (r_{v,i} - \bar{r}_v)^2}} \tag{8}$$

In addition, in order to reduce accumulated errors, we exclude trusted neighbors when predicting items' ratings in Equation 2. Only similar neighbors NN_u are used to generate recommendations, i.e.

$$\hat{r}_{u,j} = \frac{\sum_{v \in NN_u} w_{u,v} r_{v,j}}{\sum_{v \in NN_u} w_{u,v}} \tag{9}$$

[1] For only one evidence, $c(1, 0) = 0.25$ without any conflicts.

where $w_{u,v}$ is the weight between users u and v. For users $v \in TN_u$, the weight $w_{u,v}$ is the harmonic mean of trust and similarity values (refers to Equation 4; otherwise it is the similarity value $s_{u,v}$.

3.3 Experimental Results

Based on the same three real-life datasets including FilmTrust, Flixster and Epinions described in [1], we conduct a number of experiments to investigate the performance of all comparison methods. Since we are attempting to alleviate the cold start and data sparsity problems, the performance shown in the view of *Cold Users* in all results is our major concern. By setting the similarity threshold $\theta = 0.5$ (refers to Equation 3) and adjusting the number of confident ratings k to be used in forming new rating profiles, we obtain the following results shown in Tables 1, 2 and 3, corresponding to the performance on FilmTrust, Flixster and Epinions datasets, respectively. Specifically, the values of k are 30, 100, 200 for three datasets, respectively. Note that methods *TrustAll* and *RN* from [1] are eliminated as the former is a naive method without personalization (CF is our baseline method) and the latter only achieves very limited coverage. In addition, to have a fair comparison, we expand trust propagation to length 2 and set the maximum iteration for TCF to be 2 which is suggested in [3].

Table 1. The Performance on FilmTrust

MAE/RC							
Views	Approaches						
	CF	MT1	MT2	TCF1	TCF2	Merge1	Merge2
All	0.703 93.83%	0.852 21.20%	0.795 27.95%	0.714 94.92%	0.719 95.19%	0.703 94.01%	0.704 94.09%
Cold Users	**0.744** **39.64**%	0.853 17.11%	0.880 23.19%	0.751 39.97%	0.751 40.79%	**0.732** **47.70**%	0.753 48.52%
Heavy Users	0.705 94.95%	0.854 21.53%	0.797 28.25%	0.716 90.06%	0.722 96.32%	0.705 94.99%	0.706 95.04%
Opin. Users	1.469 87.63%	1.268 14.43%	1.156 15.46%	1.475 87.63%	1.460 88.66%	1.456 87.63%	1.447 90.72%
Black Sheep	1.237 90.63%	1.228 19.94%	1.243 24.82%	1.248 91.19%	1.244 92.15%	1.238 90.63%	1.238 90.67%
Contr. Items	2.106 62.58%	2.358 16.04%	2.418 21.38%	2.265 78.93%	2.257 81.76%	2.107 62.58%	2.109 62.89%
Niche Items	0.986 53.92%	1.031 14.04%	1.011 19.35%	1.014 64.20%	1.012 66.79%	0.986 54.46%	0.985 55.03%

Comparing with CF, all other methods achieve better performance for cold-start users in all datasets except in the FilmTrust where only our method *Merge1* outperforms it both in accuracy and coverage. CF cannot achieve large portion of predictable items, especially for the large datasets (Flixster and Epinions[2]) and the accuracy is usually bad. It indicates that CF suffers from cold

[2] FilmTrust is a rather small dataset relative to Flixster and Epinions.

Table 2. The Performance on Flixster

MAE/RC							
Views	Approaches						
	CF	MT1	MT2	TCF1	TCF2	Merge1	Merge2
All	0.928 68.56%	1.060 12.36%	0.932 71.37%	0.870 80.92%	0.850 85.23%	0.929 74.01%	0.897 80.72%
Cold Users	**1.153** **3.27**%	1.127 8.11%	1.005 52.69%	1.047 12.97%	0.923 21.41%	1.043 37.07%	**1.001** **52.53**%
Heavy Users	0.913 85.59%	1.046 13.29%	0.917 75.55%	0.856 94.25%	0.844 96.56%	0.905 86.35%	0.873 90.61%
Opin. Users	1.494 74.80%	1.574 12.65%	1.487 72.37%	1.465 92.97%	1.453 97.90%	1.493 75.25%	1.474 82.10%
Black Sheep	1.320 76.21%	1.300 13.59%	1.288 75.53%	1.286 90.57%	1.278 94.15%	1.316 76.89%	1.302 83.05%
Contr. Items	1.830 30.64%	1.847 2.33%	1.833 27.63%	1.921 60.93%	1.977 82.31%	1.840 33.56%	1.872 43.41%
Niche Items	1.068 11.77%	1.195 0.66%	1.021 11.23%	1.096 31.10%	1.087 49.08%	1.068 12.90%	1.087 16.85%

Table 3. The Performance on Epinions

MAE/RC							
Views	Approaches						
	CF	MT1	MT2	TCF1	TCF2	Merge1	Merge2
All	0.876 51.24%	0.845 26.34%	0.852 57.64%	0.867 70.28%	0.864 77.48%	0.863 56.67%	0.850 63.84%
Cold Users	**1.032** **3.22**%	0.756 6.57%	0.916 22.06%	0.982 7.16%	0.941 10.45%	0.943 15.98%	**0.920** **22.90**%
Heavy Users	0.873 57.41%	0.847 29.28%	0.848 62.40%	0.862 77.11%	0.860 83.73%	0.858 62.85%	0.845 70.14%
Opin. Users	1.120 49.99%	1.060 19.99%	1.124 52.02%	1.127 72.70%	1.124 82.17%	1.115 54.19%	1.106 62.78%
Black Sheep	1.246 55.72%	1.199 20.06%	1.259 53.73%	1.266 79.04%	1.269 87.58%	1.244 59.45%	1.243 67.98%
Contr. Items	1.598 45.40%	1.481 22.87%	1.646 57.81%	1.694 71.90%	1.753 84.01%	1.617 53.31%	1.639 63.18%
Niche Items	0.835 12.16%	0.743 7.84%	0.811 23.65%	0.849 30.30%	0.857 42.33%	0.839 16.13%	0.847 21.52%

start severely. When only direct trusted neighbors are used (*MT1, Merge1*), our method achieves better accuracy and coverage in FilmTrust and Flixster. But in Epinions, MT1 works better than our method in accuracy but much worse in coverage. It shows that MT1 may have a good accuracy in some dataset, but not consistently in all datasets. When trust is propagated in length 2 (*MT2, Merge2*), both accuracy and coverage are increased in Flixster and Epinions whereas only coverage is increased in FilmTrust. Nevertheless, our method outperforms MT2 in FilmTrust and Flixster, and comparative results in Epinions. It may conclude that using trust information to assist in modeling user preference is more effective than to predict items' ratings. TCF methods generally obtain better coverage in all data views except cold users, especially in the views of *Contr. Items* and *Niche Items*. This is because more similar users can be involved in predictions by merging the ratings of the trusted neighbors. However, for cold users, TCF functions badly due to the limitation that it relies on CF to find similar users before it can apply trust information on them. As we know, CF is not effective to do so for cold users. This fact leads to bad performance of TCF methods. In contrast, our method is not subject to the ratings of cold users themselves to find similar users. Instead, trust information is merged in order to form a more concrete rating profile for the cold start based on which CF is able to find similar users and hence generate recommendations.

3.4 Discussion

According to the results presented in Section 3.3, generally and consistently we can come to a conclusion that the Merge method outperforms the other approaches both in accuracy and coverage. This conclusion is consistent with that in [1], although we do not achieve comparative improvements in predictive performance. However, different from [1], we observe that when trust is propagated along the web of trust, our method achieves better performance. This may be explained by the fact that confidence value will be more accurate when more evidences are available and hence will achieve better rankings of confidences of merged items. On the other hand, in comparison with CF, our method has also yet positive influence on the other data views. Specifically, it increases the coverage to a large extent and keeps a comparative accuracy. It may conclude that the Merge method is dedicated to solving cold start and data sparsity problems while bringing positive effects on the others. This sheds some lights on the future work of combining our strategy with some other approaches, aiming to improve the performance of all data views.

References

1. Guo, G., Zhang, J., Thalmann, D.: A simple but effective method to incorporate trusted neighbors in recommender systems. In: Proceedings of the 20th International Conference on User Modeling, Adaptation and Personalization (UMAP). (2012)
2. Adomavicius, G., Tuzhilin, A.: Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. (2005) 734–749
3. Chowdhury, M., Thomo, A., Wadge, B.: Trust-based infinitesimals for enhanced collaborative filtering. In: Proceedings of the 15th International Conference on Management of Data (COMAD). (2009)
4. Singla, P., Richardson, M.: Yes, there is a correlation: from social networks to personal behavior on the web. In: Proceedings of the 17th International Conference on World Wide Web. (2008) 655–664
5. Ray, S., Mahanti, A.: Improving prediction accuracy in trust-aware recommender systems. In: Proceedings of the 43rd Hawaii International Conference on System Sciences (HICSS). (2010) 1–9

6. O'Donovan, J., Smyth, B.: Trust in recommender systems. In: Proceedings of the 10th International Conference on Intelligent User Interfaces. (2005) 167–174
7. Wang, Y., Singh, M.: Formal trust model for multiagent systems. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI). (2007) 1551–1556