# RSOL: A Trust-Based Recommender System with an Opinion Leadership Measurement for Cold Start Users

Jiun-Yuan Wang and Hung-Yu Kao*

Institute of Computer Science and Information Engineering,
National Cheng Kung University, Tainan, Taiwan, R.O.C.
hykao@mail.ncku.edu.tw

**Abstract.** The cold start problem is a potential issue in computer-based information systems that involve a degree of automated data modeling. Specifically, the system cannot infer a rating for users or items that are new to the recommender system when no sufficient information has been gathered. Currently, more websites are providing the relationships between users, e.g., the trust relationships, to help us alleviate the cold start problem. In this paper, we proposed a trust-based recommender model (*RSOL*) that is able to recognize the user's recommendation quality for different items. A user's recommendation quality contains two parts: "*Rating Confidence*"- an indicator of the user's reliability when rating an item, and "*Proximity Prestige*"- an indicator of the user's influence on a trust network. In our experimental results, the proposed method outperforms the *Collaborative Filtering* and trust-based methods on the Epinions dataset.

**Keywords:** Recommendation System, Trust Network, Cold Start Problem.

## 1 Introduction

### 1.1 Background

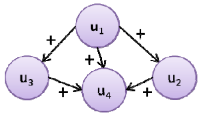
The recommender system is an important technology to help users find relevant and useful information in the information explosion era. For example, there are recommenders for movies, music [4], etc., such as MovieLens and Netflix. Recommender system analyze many factors, including the user's explicit preferences (rating history and user/item latent features), implicit preferences (the trust network), and other users' profiles, and recommend the items (movie, music, etc.) to users.

With the development of the internet, more and more websites, such as Epinions.com have provided the trust relationships between users, so trust-based recommendation methods have been highly developed. The trust-based methods use the information from the given user's neighbors in a trust network for recommendations.

In this paper, we expect to predict the ratings of users who have fewer rating profiles to be observed. To consider enough ratings for reliable users, we proposed a *Recommender System with Opinion Leadership* model (*RSOL*) that combines two indicators: *Rating Confidence* and *Proximity Prestige*. The name also represents a solution of a recommender system; that is why we named our model *RSOL*.

* Corresponding author.

R.E. Banchs et al. (Eds.): AIRS 2013, LNCS 8281, pp. 500–512, 2013.




**Rating Matrix**

| | $i_1$ | $i_2$ | $i_3$ | $i_4$ |
|---|---|---|---|---|
| $u_1$ | 3 | | 2 | |
| $u_2$ | 1 | 3 | 5 | 4 |
| $u_3$ | | 2 | | 1 |
| $u_4$ | | 1 | 3 | |

**Trust Network**

| | $u_1$ | $u_2$ | $u_3$ | $u_4$ |
|---|---|---|---|---|
| $u_1$ | X | 1 | 1 | 1 |
| $u_2$ | X | X | X | 1 |
| $u_3$ | X | X | X | 1 |
| $u_4$ | X | X | X | X |

*Example 1:* Suppose we have four users: $u_1$, $u_2$, $u_3$, and $u_4$. $u_1$ trusts $u_2$, $u_3$ and $u_4$. $u_2$ trusts $u_4$. $u_3$ trusts $u_4$. $u_4$ does not trust anyone. Additionally, we have the rating profiles of the four users. We want to know the rating of item $i_3$ rated by user $u_1$. We compare two methods, *Collaborative Filtering* and the Trust-based method, with our method-*RSOL*. We illustrated the three different methods with the figures below:

**Collaborative Filtering**

**Rating Matrix**

| | $i_1$ | $i_2$ | $i_3$ | $i_4$ |
|---|---|---|---|---|
| $u_1$ | 3 | | 2 | |
| $u_2$ | 1 | 3 | 5 | 4 |
| $u_3$ | | 2 | | 1 |
| $u_4$ | | 1 | 3 | |

**User Similarity Matrix**

| | $u_1$ | $u_2$ | $u_3$ | $u_4$ |
|---|---|---|---|---|
| $u_1$ | X | -0.422 | -0.87 | 0.157 |
| $u_2$ | | X | 0.05 | 0.69 |
| $u_3$ | | | X | -0.246 |
| $u_4$ | | | | X |

**Rating Matrix**

| | $i_1$ | $i_2$ | $i_3$ | $i_4$ |
|---|---|---|---|---|
| $u_1$ | 3 | | 2 | |
| $u_2$ | 1 | 3 | 5 | 4 |
| $u_3$ | | 2 | | 1 |
| $u_4$ | | 1 | 3 | |

**Item Similarity Matrix**

| | $i_1$ | $i_2$ | $i_3$ | $i_4$ |
|---|---|---|---|---|
| $i_1$ | X | 0.006433 | -0.111443 | -0.073782 |
| $i_2$ | | X | -0.024985 | 0.093384 |
| $i_3$ | | | X | 0.045423 |
| $i_4$ | | | | X |

**RSOL**

**RC Matrix**

| | $i_1$ | $i_2$ | $i_3$ | $i_4$ |
|---|---|---|---|---|
| $u_1$ | 0.362877059 | 0.036878964 | 0.069810914 | 8.68E-16 |
| $u_2$ | 0.055500449 | 5.79E-16 | 0.034825143 | 0.150458111 |
| $u_3$ | 1.93E-16 | 0.288260094 | 0.241635502 | 0.876603364 |
| $u_4$ | 0 | 0.203888517 | 0.19119152 | 0.048372841 |

**PP Matrix**

| | PP |
|---|---|
| $u_1$ | 0 |
| $u_2$ | 0.25 |
| $u_3$ | 0.25 |
| $u_4$ | 1 |

**RC+PP Matrix**

| | $i_1$ | $i_2$ | $i_3$ | $i_4$ |
|---|---|---|---|---|
| $u_1$ | 0.362877 | 0.036879 | 0.069811 | 8.68E-16 |
| $u_2$ | 0.15275 | 0.125 | 0.142413 | 0.200229 |
| $u_3$ | 0.125 | 0.26913 | 0.245818 | 0.563302 |
| $u_4$ | 0.5 | 0.601944 | 0.595596 | 0.524186 |

$$R_{u_1,i_1} = \frac{0.1424 \times \frac{13}{4} + 0.2458 \times \frac{3}{2} + 0.5955 \times \frac{4}{2}}{0.1424 + 0.2458 + 0.5955} = 2.1702$$

The predict ratings are *User-based CF*: 1.5931, *Item-based CF*: 3, Trust-based method: 2.9375, and *RSOL*: 2.1702. The example shows us that the performance of our *RSOL* model is the best among the three methods. Significantly, *u1* is a cold start user who has fewer ratings; our *RSOL* model can address the cold start user problem. However, *Collaborative Filtering* and trust-based methods fail on the cold start user problem.

## 2 Related Work

How to infer an indirect trust is a key issue of the trust-based recommendation system. Two different approaches have been proposed for inferring the trust: model-based

[2, 5-8] and memory-based [1, 3, 10]. In model-based approaches, a model should be learned; the model stores the model parameters. In memory-based approaches, no model will be learned first; it learns by exploring neighbors from dataset.

*TidalTrust* [1] is a trust-based method; it is a modified breadth-first search algorithm in a trust network. It predicts that people who users trust highly at the shortest distance are the most important users. The *TidalTrust* algorithm explores all the users at the shortest distance from the source user, and then it averages their ratings, weighted by the trust value between the source user and the users being explored. To compute the indirect trust value between user *u* and *v*, it aggregates the trust value between *u's* direct neighbors, weighted by the direct trust values of *u* and its direct neighbors. *TidalTrust* uses ratings that are dependent on the users at shortest distance, but it does not consider that whether we should trust these users about the target item. Additionally, *TidalTrust* only considers the users who are at the shortest distance; it ignores the trustworthy users who are slightly farther from the source user in the trust network.

*MoleTrust* [10] is also a trust-based method. The idea of *MoleTrust* is similar to *TidalTrust*. *MoleTrust* also weights the ratings of trusted users with a trust score, but it considers all users up to a maximum depth. However, the larger the maximum depth is, the higher the cost of *MoleTrust*, so previous works consider the users up to a maximum depth of 6. Because *MoleTrust* considers the users who are close to the source user, within a maximum depth of 6, it does not consider different set of users who are also appropriate to target item. It loses many users who are trustworthy or have rated the target item but are far from the source user.

*TrustWalker* [3] has been introduced as a random walk method that combines a trust-based and item-based recommendation to predict the rating of single items. *TrustWalker* performs random walks on the trust network to find ratings for the target items or similar items. The prediction from *TrustWalker* is based on the ratings from these trusted users up to a certain depth (which is 6) and the similar items rated by them. However, when finding the trusted users who can appropriately predict an rating for a target item, *TrustWalker* is not dependent on the target item but only on the users on the trust network. It may lose the users who are trustworthy about the target item because of the sparsity of trust network.

## 3 Method

Our model consists of two indicators: the *Rating Confidence* for each user on different items and the *Proximity Prestige* of user on the trust network. First of all, we do the item clustering, and then calculate the distance between users' preferences and characteristic of target item and *Proximity Prestige* on different sub-network. We then return the ratings of each user with high recommendation quality. In the following subsections, we will discuss the details of our *RSOL* model.

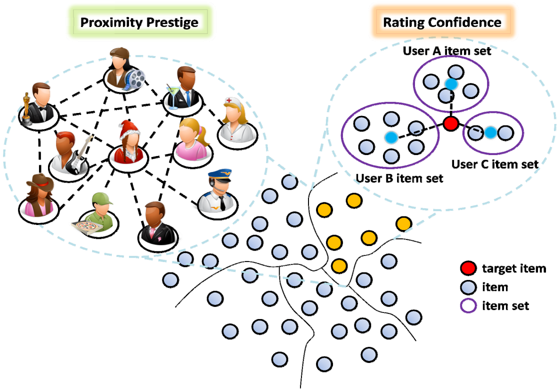



**Fig. 1.** Concepts of Rating Confidence and Proximity Prestige

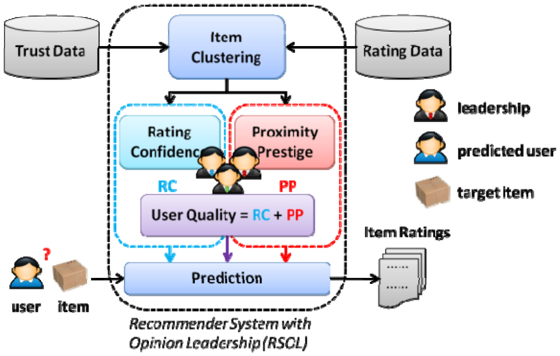



**Fig. 2.** Recommender system with opinion leadership model

### 3.1 Rating Confidence

When we want to predict the rating for a cold start user on an unrated target item, we need to know users who should I trust toward the target item. For the reason, we cluster all items. Due to the items in same cluster have similar characteristic on ratings for each other; they will be helpful to the target item rating prediction. We employ *K-Means* clustering method to find similar characteristic of items on a sparse data. Employing latent feature vectors to cluster sparse data has been used successfully in the previous study, such as [13].

$$RC(u, c_k) = \log_{10}\left(\frac{Max\ D}{d_{u,c_k}}\right) \quad (1)$$

$$\text{where } \mathrm{D} = \{\mathrm{d}_{\mathrm{u,c_k}} | \mathrm{d}_{\mathrm{u,c_k}} = |\hat{\mathrm{h}}_{\mathrm{u}} - \mathrm{v}_{\mathrm{c_k}}|\}, \text{for each user } \hat{\mathrm{h}}_{\mathrm{u}} = \frac{\sum_{i \in \mathrm{I}_{\mathrm{u,c_k}}} \mathrm{r}_{\mathrm{u,i}} \times \mathrm{h}_{\mathrm{i}}}{\sum_{i \in \mathrm{I}_{\mathrm{u,c_k}}} \mathrm{r}_{\mathrm{u,i}}}$$

Here, $c_k$ denotes the center of the $k^{th}$ cluster, which the target item belongs to. $\mathrm{U}_{\mathrm{c_k}}$ denotes a set of users who have rated at least one of the items in cluster $c_k$. $\mathrm{v}_{\mathrm{c_k}}$ is the vector of $c_k$. $h$ is a vector that represents an item. Additoinally, we use the Euclidian distance to compute the confidence of a user about the characteristics of the target item. $j_u$ denotes the preference vector of the user $u$ and $i$ is the target item's characteristic vector.

$$d(i, j_u) = \sqrt{(i_1 - j_{u1})^2 + (i_2 - j_{u2})^2 + \cdots + (i_d - j_{ud})^2} \tag{2}$$

We have two types of *Rating Confidence*, Global and Local. Global *Rating Confidence* calculates the distance between the user and the centroids of the clusters. Every item in the same cluster will have the same *Rating Confidence* for a user who is involved in the cluster. Local *Rating Confidence* calculates the distance between the users and the items in a cluster. Every item will have different *Rating Confidence* for different users involved in the cluster.

## Item Representation

*Matrix Factorization* [6] decomposes the ratings matrix into two lower dimension matrices $P \in R^{|U| \times d}$ and $Q \in R^{|I| \times d}$ which contain corresponding vectors with length $k$ for every user and item. The resulting dot product, $q_i^T p_u$, captures the interaction between user $u$ and item $i$ – the user's overall interest in the item's characteristics.

$$\hat{r}_{ui} = q_i^T p_u \tag{3}$$

To determine the latent feature vectors ($p_u$ and $q_i$), the system minimizes the regularized squared error on the set of observed ratings:

$$\min_{P^*, Q^*} \sum_{(u,i) \in R_O} \left(r_{ui} - P_u Q_i^T\right)^2 + \lambda(\|P_u\|^2 + \|Q_i\|^2) \tag{4}$$

Here, $Ro$ is the set of the $(u,i)$ pairs for which $r_{ui}$ is observed.

Thus, *Matrix Factorization* characterizes every user and item by assigning them a latent feature vector. We use the item feature vector $q_i$ to represent each item.

*Example 2:* Fig. 3 is an example of *Rating Confidence*. Suppose we have three users: Alan, Bobby, and Claire. None of them have watched movie 1; Alan has watched movie 2 and 3. Bobby has watched movie 4 and 5. Claire has watched movies 6 and 7. Training the *Matrix Factorization* model with $k = 2$ yields two matrices $P$ and $Q$ consisting of user and item factor vectors:

$$\mathrm{R} = \begin{bmatrix} 0 & 5 & 4 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 2 \end{bmatrix}, \ \mathrm{Q} = \begin{bmatrix} 0.75 & 0.83 \\ 1.1 & 0.2 \\ 0.1 & 1.2 \\ 0.2 & 1.2 \\ 0.9 & 1.0 \\ 1.3 & 0.3 \\ 0.9 & 1.1 \end{bmatrix} \xrightarrow{\text{yields}} \mathrm{p}_{\mathrm{u,m1}} = \begin{vmatrix} 0.65 & 0.64 \\ 0.375 & 1.15 \\ 1.16 & 0.56 \end{vmatrix}$$

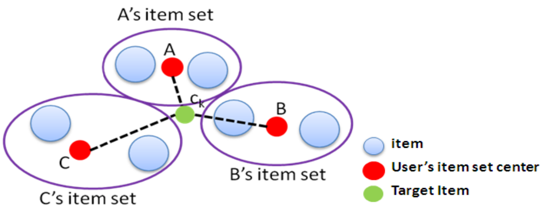



**Fig. 3.** An example of using *Rating Confidence* to find trustworthy users

Here, matrix *R* is the user item rating matrix and matrix *Q* is the item latent features learned by the *Matrix Factorization* model. $P_{u,m1}$ denotes the centroids representing the three users preferences. The distance is between the centroid of the user's item set and the target item's feature vector ($Dist(Alan, m1) = 0.214$, $Dist(Bobby, m1) = 0.492$, $Dist(Claire, m1) = 0.490$). The shorter the distance between user and target item is, the higher the *Rating Confidence* of the users ($RC(Alan, m1) = log\frac{0.492}{0.214} = 0.3615$, $RC(Bobby, m1) = log\frac{0.492}{0.492} = 0$, $RC(Claire, m1) = log\frac{0.492}{0.490} = 0.0017$). In this example, Alan has the highest *Rating Confidence* toward the target item among these three users.

## 3.2 Proximity Prestige

Prestige as a measure of the prominence, applies only to directed graphs, taking into account the differences between sending and receiving relationships. *Proximity Prestige* is the average distance between users *i* and another user *j* that is in user *i's* influence domain:

$$Proximity = \frac{\sum_j dist(n_j, n_i)}{I_i} \qquad Proximity\ Prestige = \frac{I_i/(|U|-1)}{Proximity} \tag{5}$$

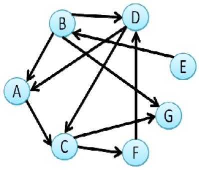
Here, $I_i$ is the influence domain, and $dist(n_j, n_i)$ is the distance from user *i* to user *j*.

This indicator is the ratio between the number of users in the influence domain and the average distance of these users to user *i*. If user *i* is unreachable, PP = 0; if all users are directly tied to user *i*, *PP* = 1. If the numerator is large, the value of PP will be large. The meaning of the numerator is the number of users that user *i* will influence. Additionally, if the denominator is small, the value of PP will be large. The meaning of the denominator is the distance between the user *i* and the users in his influence domain. If the user *i* is closed to these users, it is more probable that other users will trust this user *i*.

We have two types of *Proximity Prestige*, Global and Local. Global *Proximity Prestige* considers a user prestige in the original trust network. We consider every user's influence domain and the average shortest path between the users in the influence domain. Local *Proximity Prestige* considers a user's prestige in the sub-network. The networks are divided according to the users included to a cluster. There are different trust networks for every cluster, and we consider every user's influence domain and the average shortest path between the other users in these sub-networks.

*Example 3:* Fig. 4 is an example of *Proximity Prestige*. Suppose we have seven users: Alan, Bobby, Claire, David, Eric, Federer, and Gerel. Alan trusts Claire. Bobby trusts Alan, David, and Gerel. Claire trusts Federer and Gerel. David trusts Alan and Claire. Eric trusts Bobby. Federer trusts David. Gerel does not trust anyone.

$$\mathrm{Dist} = \begin{bmatrix} 0 & 1 & 3 & 1 & 2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 & 3 & 2 & 0 \\ 3 & 1 & 2 & 0 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 3 & 1 & 2 & 4 & 0 & 0 \\ 2 & 1 & 1 & 2 & 2 & 3 & 0 \end{bmatrix}$$




$$\mathrm{M_{PP}} = \begin{bmatrix} \mathrm{A} & 0.53 \\ \mathrm{B} & 0.16 \\ \mathrm{C} & 0.55 \\ \mathrm{D} & 0.55 \\ \mathrm{E} & 0 \\ \mathrm{F} & 0.416 \\ \mathrm{G} & 0.654 \end{bmatrix}$$

$$\mathrm{PP_A} = \frac{5/6}{9/5} = 0.46,\ \mathrm{PP_B} = \frac{1/6}{1/1} = 0.16, \mathrm{PP_C} = \frac{5/6}{9/5} = 0.46,\ \mathrm{PP_D} = \frac{5/6}{9/5} = 0.46$$

$$\mathrm{PP_E} = \frac{0/6}{0/0} = 0, \mathrm{PP_F} = \frac{5/6}{12/5} = 0.0027, \mathrm{PP_G} = \frac{6/6}{11/6} = 0.54$$

**Fig. 4.** An example of using *Proximity Prestige* to find trustworthy users

Here, *Dist* stores the distance between the all users and $M_{PP}$ store the *Proximity Prestige* of all users. We also show the operation of *Proximity Prestige* for all users above. A: 0.53, B: 0.16, C: 0.55, D: 0.55, E: 0, F: 0.416, G: 0.654. As User G has the most people in his influence domain (A: 5, B: 1, C: 5, D: 5, E: 0, F: 5, G: 6) and the average distance between user G and other users is short (A: 1.8, B: 1, C: 1.8, D: 1.8, E: 0, F: 2.4, G: 1.83), user G has the highest value for *Proximity Prestige* of the seven users. Thus, user G has the highest prestige in the trust network.

### 3.3 Rating Prediction and Explaining Recommendation

The values of the RC and PP metrics are used in conjunction with the *RSOL* model to present item-dependent trust-based recommendations. When we want to know the rating that a user would give a target item, the recommended rating is computed by the ratings of a set of users who are trustworthy. The selection process considers a user's RC toward the target item and the PP that a user has on trust network:

$$r_{u_g,i} = \frac{\sum_{\{u | r_{u,i} \neq 0 \text{ and } i \in I_{c_k}\}} \left[ \left( RC_{u,i} + PP_u \right)/2 \right] \times \bar{r}_{c_k}(u)}{\sum_{\{u | r_{u,i} \neq 0 \text{ and } i \in I_{c_k}\}} \left( RC_{u,i} + PP_u \right)/2} \qquad \bar{r}_{c_k}(u) = \sum_{\{j | j \in I_{c_k} \text{ and } r_{u,j} \neq 0\}} \frac{r_{u,j}}{|I_{c_k}|} \tag{6}$$

Here, $u_g$ denotes the given user, $i$ denotes the target item, $I_{c_k}$ denotes a set of items in a cluster $k$, and $\bar{r}_{c_k}(u)$ denotes the average rating of user $u$ in cluster $c_k$. $RC_{u,i}$ denotes the RC of user $u$ for target item $i$, and $PP_u$ denotes PP of user $u$ in trust network. $r_{u_g,i}$ -denotes the predicted rating of given user $u_g$ for target item $i$.

# 4 Experiments

## 4.1 Dataset Description and Experiment Design

The Epinions dataset [9] is very sparse (99.99% and 99.97%). It contains 49k users with at least one rating, of which 16k users (34.3%) are cold start users who have less than 5 ratings (similar to previous works [3, 10]). It is important to consider the performance of the recommendation system for cold start users. The statistics for the Epinions rating data are summarized in Table 1.

**Table 1.** Statistics for the Epinions dataset

| Rating Data | Epinions |
|---|---|
| #-of-User | 49,288 |
| #-of-Item | 139,783 |
| #-of-Rating | 664,824 |
| Min Rating | 1 |
| Max Rating | 5 |
| Avg. Rating | 3.99 |
| Rating Sparsity | 99.98% |

| Trust Data | Epinions |
|---|---|
| Nodes | 49,288 |
| Edges | 487,183 |
| Avg. Node Degree | 19.77 |
| Avg. Shortest Path | 4 |
| Diameter | 14 |
| Avg. Trustor | 2,070 |
| Avg. Trustee | 3,338 |
| Trust Network Sparsity | 99.96% |

**Table 2.** The number of ratings, users, and items in four types of cold start users. CS-1 denotes cold start user who has one rating, and so on

| | Density | #Rating | #User | #Item |
|---|---|---|---|---|
| CS-1 | 0.000192 | 7,739 | 7,739 | 5,201 |
| CS-2 | 0.000362 | 7,874 | 3,937 | 5,518 |
| CS-3 | 0.000485 | 8,751 | 2,917 | 6,188 |
| CS-4 | 0.000619 | 9,268 | 2,317 | 6,461 |

Table 2 shows the number of the cold start users, ratings they have, and items included, and the density of the user item matrix.

## 4.2 Comparison Methods and Evaluation Metrics

In our experiments, we compare the results with two baselines and three state-of-the-art methods. The following is the description of the labels we use to denote the methods:

*User-based CF*: We implemented the user-based Collaborative Filtering method [12], with the *Pearson Correlation* as the similarity measure. *Item-based CF*: We implemented the item-based *Collaborative Filtering* method [11] with the *Pearson Correlation* as similarity measure. *TidalTrust* is the trust-based approach from a previous study [1], proposed by Golbeck. *MoleTrust*: This is the approach in a paper [10], which is similar to *TidalTrust*. We use max_depth=6 for *MoleTrust* as well. *TrustWalker* is the approach in a paper [3], which combine the trust-based and item-based recommendations. $RSOL_{RC(Global)}$ and $RSOL_{RC(Local)}$: This method is one version of our *RSOL* model in which we only consider the RC metric for all <user, item> pairs. $RSOL_{PP(Global)}$ and $RSOL_{PP(Local)}$: This method is another version of our *RSOL* model in which we only consider the PP metric for all users on a trust network in different item clusters. $RSOL_{All}$: This is the full version of the *RSOL* model. We combine the two user metrics to help us choose the trustworthy users.

We perform leave-one-out cross validation in our experiment which is the same as the previous works [1, 10, 11]. In the leave-one-out cross validation, we try to predict a target item rating by using the remain ratings and the trust relationships between users in trust network. In our experiment, the evaluation metric we use to measure the error is the Root Mean Squared Error (*RMSE*) which is defined as follows:

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{<u,i>\in \mathrm{R_{train}}} (\mathrm{r_{ui}} - \hat{\mathrm{r}}_{\mathrm{ui}})^2}{|\mathrm{R_{train}}|}} \tag{7}$$

As the paper [3] discussed, the purpose of using trust is primarily enhancing the *Coverage* without sacrificing the *Precision*. We use the *Coverage*, *Precision*, *F-Measure* metric that is mentioned in the paper.

$$\mathrm{Precision} = 1 - \frac{\mathrm{RMSE}}{4} \tag{8}$$

$$\mathrm{F-Measure} = \frac{2 \times \mathrm{Precision} \times \mathrm{Coverage}}{\mathrm{Precision} + \mathrm{Coverage}} \tag{9}$$

### 4.3 Evaluation Results

Fig. 5 is the results of *RMSE* for different values of the parameter *k*, which is one of the versions of the *RSOL* model. We use a different threshold for *RC* to select the reference users to conduct our experiments. The result of the $RSOL_{RC(Local)}$ is better than the $RSOL_{RC(Global)}$, because $RSOL_{RC(Local)}$ considers the *RC* of every user for different items. We can select different set of the trustworthy users according to the target items in a cluster that we want to predict. In contrast, $RSOL_{RC(Global)}$ just selects the same set of the trustworthy reference users for different target items in a cluster.

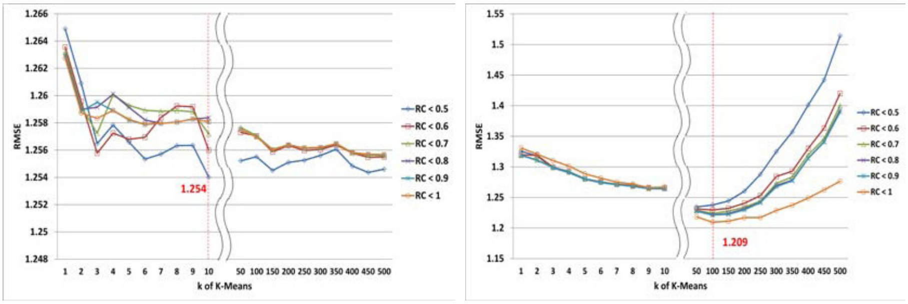



**Fig. 5.** RMSE of $RSOL_{RC(Global)}$ and $RSOL_{RC(Local)}$ for different values of k

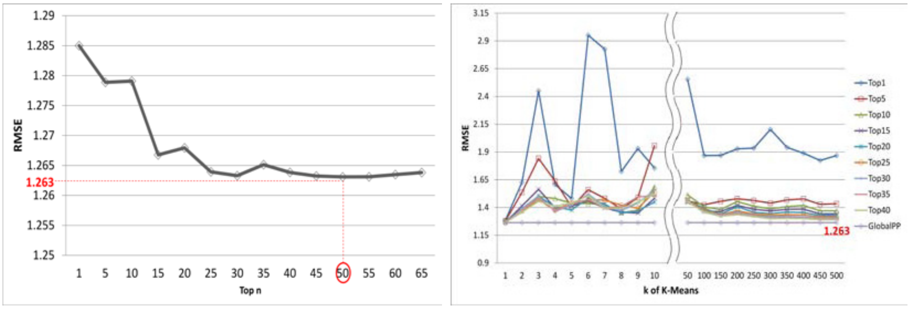



**Fig. 6.** RMSE of $RSOL_{PP(Global)}$ and $RSOL_{PP(Local)}$ for different values of k

Fig. 6 is the *RMSE* for different values of the parameter *k* in two versions of the *RSOL* model. We use Top-n users with highest *PP* to select the reference users to perform our experiments. Fig. 10 shows us that the result of Global *PP* is better than Local *PP*. Now we have two metrics, Local *RC* and Global *PP*, that have smaller square errors. We combine these two metrics to help us find trustworthy users.

Fig. 7 shows us that the combination of Global *PP* and Local *PP* is the best of the four previously mentioned versions of our *RSOL* model. By using two metrics, we can find the trustworthy users who are have the highest recommendation confidence for the target items and prestige in the trust network. The two metrics help us decide which benefits the predictions.

As shown in Table 3, also shows the *F-Measure* together with *Precision* and *Coverage* for all methods. When comparing the *RMSE* of “Local” and “Global” shows that considering the effect of items in different cluster reduces the square error. It shows that all four versions of *RSOL* model outperform all other methods according to the combination of precision and coverage. Notably, *RSOL*'s coverage is 29.83% more than that of *TrustWalker*, which makes *RSOL* model is best in terms of *F-Measure*.

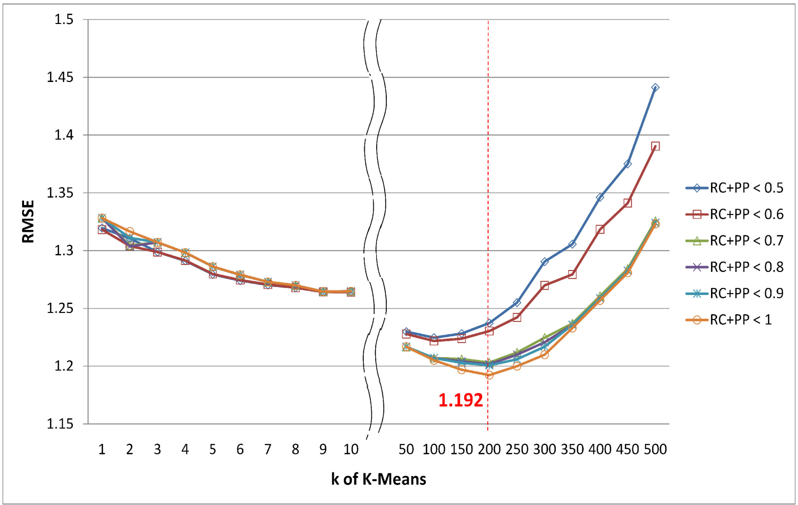



**Fig. 7.** RMSE of the *RSOL* model for different values of k

**Table 3.** Summary of all comparison methods in terms of *Precision*, C*overage*, and *F-measure*

| Method | Cold Start User | | |
|---|---|---|---|
| | **RMSE** | **Coverage (%)** | **F-Measure** |
| **User-based CF** | 1.464 | 16.34 | 0.259 |
| **Item-based CF** | 1.295 | 21.26 | 0.316 |
| **TidalTrust** | 1.244 | 60.92 | 0.626 |
| **MoleTrust** | 1.532 | 57.75 | 0.594 |
| **TrustWalker** | 1.192 | 74.22 | 0.701 |
| **$\text{RSOL}_{\text{RC(Global)}}$** | 1.254 | 100 | 0.814 |
| **$\text{RSOL}_{\text{PP(Global)}}$** | 1.263 | 100 | 0.811 |
| **$\text{RSOL}_{\text{RC(Local)}}$** | 1.209 | 100 | 0.821 |
| **$\text{RSOL}_{\text{PP(Local)}}$** | 1.250 | 100 | 0.817 |
| **$\text{RSOL}_{\text{All}}$** | 1.192 | 100 | 0.825 |

In Fig. 8, each node represents a user and each edge is a trust relationship between two users. On the right, the color of the nodes corresponds to their coreness value. The node degree scale is also displayed on the left, showing the maximum degree of the network. We show the Top 100 users selected by one version of our *RSOL* model – *Proximity Prestige*. We found that these users located on the center of the image (these nodes are large and red), so we can explain that our network has the characteristic of a core/periphery structure. The core is a complete subgraph and the periphery is a collection of nodes that do not interact with each other. The core nodes have short path distances between a pair of nodes. Because the core nodes control the flows between peripheral nodes, selecting users is helpful for the predictions.

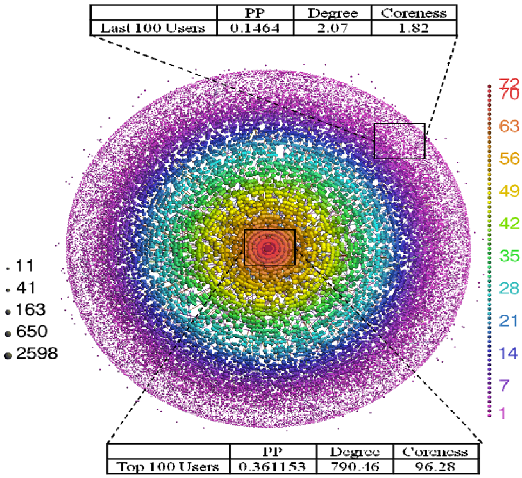



**Fig. 8.** The visualization of the nodes degree and coreness in a trust network

## 5 Conclusions

Recommender systems are important technology for helping users find relevant and useful information in the information explosion era. The *Sparsity* of the user item ratings data not only fails to compute the similarity between two users but also forces the trust-based methods to consider the ratings of indirect neighbors who may not be trustworthy, which may decrease the performance of the recommender system. To address this problem, we proposed the *Recommender System with Opinion Leadership* (*RSOL*) model to consider the user's recommendation quality, which includes the user's RC for different items and the user's influence on the trust network. *RSOL* is an item-dependent model that can consider a set of appropriate user for each different item. We performed an evaluation on the Epinions dataset; the results of experiments show that *RSOL* outperforms both *Collaborative Filtering* methods and trust-based methods, especially in terms of *Coverage*.

This study suggests two interesting directions for future work. First, we want to evaluate the *RSOL* model on other available datasets. Second, in addition to the cold start user problem, the cold start item problem is also a more and more important task for recommender systems. We plan to investigate the extension of the *RSOL* model for this task. In addition, the trust concept we considered in this paper does not integrate text information, such as the user's reviews of items. However, people may trust or distrust people on certain item because of the reviews they have written. User reviews may be a good indicator of the intensity of the underlying text effect. Combining the review and trust information models for recommendations is also a direction for future work.

## References

[1] Golbeck, J.A.: Computing and applying trust in web-based social networks, University of Maryland at College Park. p. 199 (2005)
[2] Jamali, M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the Fourth ACM Conference on Recommender Systems, pp. 135–142. ACM, Barcelona (2010)
[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. ACM, Paris (2009)
[4] Koenigstein, N., Dror, G., Koren, Y.: Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In: Proceedings of the Fifth ACM Conference on Recommender Systems, pp. 165–172. ACM, Chicago (2011)
[5] Koren, Y.: Factorization meets the neighborhood: a multifaceted collaborative filtering model. In: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 426–434. ACM, Las Vegas (2008)
[6] Koren, Y., Bell, R., Volinsky, C.: Matrix Factorization Techniques for Recommender Systems. Computer 42(8), 30–37 (2009)
[7] Ma, H., Yang, H., Lyu, M.R., King, I.: SoRec: social recommendation using probabilistic matrix factorization. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management, pp. 931–940. ACM, Napa Valley (2008)
[8] Ma, H., Zhou, D., Liu, C., Lyu, M.R., King, I.: Recommender systems with social regularization. In: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, pp. 287–296. ACM, Hong Kong (2011)
[9] Massa, P., Avesani, P.: Trust-aware bootstrapping of recommender systems. In: ECAI Workshop on Recommender Systems. Citeseer (2006)
[10] Massa, P., Avesani, P.: Trust-aware recommender systems. In: Proceedings of the 2007 ACM Conference on Recommender Systems, pp. 17–24. ACM, Minneapolis (2007)
[11] Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: Proceedings of the 10th International Conference on World Wide Web, pp. 285–295. ACM, Hong Kong (2001)
[12] Wang, J., de Vries, A.P., Reinders, M.J.T.: Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 501–508. ACM, Seattle (2006)
[13] Xu, W., Liu, X., Gong, Y.: Document clustering based on non-negative matrix factorization. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, pp. 267–273. ACM, Toronto (2003)