A New Strategy in Trust-Based Recommender System using K-Means Clustering

Naeem Shahabi Sani
Department of Computer Engineering
Islamic Azad University, Science and Research Branch
Tehran, Iran

Ferial Najian Tabriz
Department of Computer Engineering
Islamic Azad University, North Tehran Branch
Tehran, Iran

***Abstract*—Recommender systems are among the most important parts of online systems, including online stores such as Amazon, Netflix that have become very popular in the recent years. These systems lead users to finding desired information and goods in electronic environments. Recommender systems are one of the main tools to overcome the problem of information overload. Collaborative filtering (CF) is one of the best approaches for recommender systems and are spreading as a dominant approach. However, they have the problem of cold-start and data sparsity. Trust-based approaches try to create a neighborhood and network of trusted users that demonstrate users' trust in each other's opinions. As such, these systems recommend items based on users' relationships. In the proposed method, we try to resolve the problems of low coverage rate and high RMSE rate in trust-based recommender systems using k-means clustering and ant colony algorithm (TBRSK). For clustering data, the k-means method has been used on MovieLens and Epinion datasets and the rating matrix is calculated to have the least overlapping.**

Keywords—Recommendation systems; collaborative filtering; trust-based recommendation system; k-means; ant colony

I. Introduction

Recommended systems (RS) are designed to help and guide users in finding their desired items from large-scale datasets such as the internet [1]. The most successful RS is Collaborative Filtering (CF) technique that focuses on users' previous online behavior [2]. CF approach is categorized into model-based and memory-based groups. The first group models each user based on his online activities and predicts his interests. The second group focuses on user's rating matrix to find the most similar person to each user. The memory-based approach works in three steps. First, the similarity of users is measured usually through the Pearson Correlation Coefficient. Then, users who are the most similar to the active user are selected as his neighbors [3]. Finally, using the neighbor ensemble, the user's interests in unrated items is predicted [4].

Recommending methods based on user's feedback are used in most online trading systems such as Amazon and Netflix. CF [5] as a dominant approach used in recommender systems is spreading to web service recommendations. A new generation of CF approaches is social CF approach which uses users' social behavior for recommendation. Trust-based approaches of CF use the social activities of users to recognize trust among users and improve the accuracy of recommendation [3], [6]. However, these methods still suffer sparsity and cold-start problems of traditional recommender systems.

A. Sparsity Problem

In addition to the extremely large volume of user-service rating data, only a small number of users usually rate. Therefore, data density of user feedback is usually less than 0.1 [7]. This data sparsity causes many problems in CF approaches for recognizing similar users or services by a common similarity measure like cosine measure.

B. Cold-Start Problem

The cold-start problem, including users with few feedbacks, services with small number of rating slow-rated services, and new users with new services, is another challenge in recommendation research. Due to lack of user feedback, no similarity-based method can help with the cold-start problem.

Another problem related to the above-mentioned problems in trust-based recommender systems is the low coverage rate. This problem does not let systems to completely predict users' ratings. To solve this problem, in our proposed method, the ant colony algorithm has been used. However, trust-based recommender systems that use ant algorithm have high RMSE, i.e. the low quality of ratings predicted by the system. We have been able to solve this problem in our method.

In Section II of this paper, the proposed approach for improving the efficiency of trust-based recommender systems will be introduced which includes four steps: 1) calculating users' similarity and trust; 2) clustering based on users' trust to each other; 3) predicting the ratings; and 4) recommending N items to the user. Section III will measure the efficiency of the proposed approach based on two sets of data. Section IV includes the results of the study and a comparison between the efficiency of the proposed method with other approaches. Section V will discuss the conclusion of the study.

II. Proposed Algorithm

In this section, a new memory-based approach is introduced in order to increase the performance of trust-based recommender systems. This method is called TBRSK. The main purpose of this approach is to use the ant colony parallel with TRACCF in [8] to increase the coverage rate and predict the ratings that TRACCF is not capable of. It should be noted that when TRACCF cannot calculate the prediction, the proposed approach can find trusted friends for the active user using the ant colony and predicts the desired ratings.

The proposed approach has four steps, including 1) calculating users' similarity and trust; 2) clustering based on users' trust to each other; 3) predicting the ratings; and 4) recommending N items to the user. The inputs of the proposed algorithm are the Rating Matrix and Top-N. These parameters specify the rating matrix, number of clusters and number of recommendations for the target user, respectively. The input dataset is divided into training and testing datasets. Trust and similarity values are calculated for data with the help of (1) and (2), respectively. The dataset is divided into k clusters based on the trust equations. Prediction is made for all members of the testing dataset. At first, this prediction is based on (3). If this equation is not able to predict the rating i for the user u, prediction of this rating will be given to (6), but before doing the prediction step, it is needed to calculate the probability values of selecting trusted friends and finding trusted friends and this is done according to (4) and (5). Pheromone updating is provided to increase the trust rate of the target user for users who participated in the prediction and later, these users will be selected with higher probability in the future predictions. Finally, Top-N is recommended to the target user as the interested items.

A. Calculation of Trust and Similarity

At first, user's trust and similarity matrix should be calculated. The Pearson Correlation Coefficient criterion is used to calculate similarity among users [9]. Equation (1) is used to calculate the similarity between u and v users:

$$sim(u,v) = p_{u,v} = \frac{\sum(r_{u,i}-\bar{r}_u)(r_{v,i}-\bar{r}_v)}{\sigma_u\sigma_v} \quad (1)$$

Where $r_{u,i}$ and $r_{v,i}$ denote the ratings of users u and v for ith item, respectively and $\bar{r}_u$ and $\bar{r}_v$ are the average ratings of these users over all their rated items respectively σ_u and σ_v denote the standard deviation of the ratings of users u and v, respectively.

After calculating similarity among users, trust among users is calculated using (2) as follows [8]:

$$trust(u,v) = \frac{card(A_{u,v})}{card(A_u)} \quad (2)$$

Where, $A_{u,v}$ is the rating set given by users u and v, and A_u is the rating set given by user u.

B. Clustering-Based on Trust among Users

One of the problems in recommender systems is the problem of cold-start users and Sparsity. When a target user rates few data, calculating the similarity of that specific user with the others is a problem. This problem will lead to challenges in finding neighborhoods. In this approach, clustering is used to classify similar users to n clusters, in order to have clusters of users with the most similarity to each other.

Selecting neighborhoods for target users to predict their ratings is another challenge in recommender systems. CF recommender systems, which predict the ratings based on the target user's neighbors, should be able to identify the target user's neighbors.

In such systems, selecting the correct neighborhood of users, it is possible to predict highly accurate ratings for the target user. However, selecting and determining the neighbor users in recommender systems has many challenges. If a target user rates few items, calculation of trust between this user and other users in the system will be difficult which will make it hard to choose the appropriate neighborhood. Clustering is one of the methods used to solve the problem of neighborhood selection in recommender systems. Clustering locates similar users or items in a cluster. This way, the users in the target user's cluster can be used to predict the desired rating.

Although, clustering method can solve the problem of neighborhood selection in recommender systems, this method has some problems and challenges as well. Determining the correct number of clusters is a main issue in clustering-based methods, as the performance of these methods relies on determining the initial number of clusters. If the correct number of clusters is not selected at the beginning, these methods will not have high performance. The second challenge is the inappropriate number of generated clusters. Weak clustering results may lead to low-accuracy predictions and low-coverage rate of ratings. This problem happens when during the clustering process, clusters with few users are generated; thus unable to provide appropriate neighborhoods for their users.

In addition, most clustering-based systems only use the similarity criterion among users or items. As a result, clustering method is unable to cluster in the best way for cold-start users and high sparsity data. Therefore, using other factors, such as trust relationships, alongside the similarity criterion can help these methods with better categorization of users and items.

Clustering is used to use the ratings of users who are most similar to the active user. Most recommender systems that use clustering method only use the similarity criterion among users or items for clustering. However, using such criterion prevents having clusters with the problems of cold-start users and data sparsity; therefore, in this approach, trust relationships have been used for clustering. It should be noted that this action would increase the accuracy of predictions. K-means method is used to cluster users' trust in the proposed approach.

C. Predicting Ratings

After calculating the similarity and trust among users, the ratings are predicted through combining users' similarity and trust values, as in (3), also used in [8].

$$p_i(u) = \bar{r}(u) + \frac{\sum_{v \in v_u}\big(\sigma.trust(u,v)+(1-\sigma).sim(u,v)\big).(r_i(v)-\bar{r}(v))}{\sum_{v \in v_u}|\sigma.trust(u,v)+(1-\sigma).sim(u,v)|} \quad (3)$$

Where, $r_i(u)$ is the given rating to the item i by user u and $\bar{r}_u$ is the average ratings given by user u; Trust (u, v) and sim(u,v) are the trust and similarity of user u to user v, that were explained in this chapter. σ is the rating weight which is a number in the range [0, 1] [10] and v_u is the set of users who are in the same cluster as the active user.

If (3) does not generate the prediction, in our proposed approach, the ant colony algorithm [9] is used to calculate the prediction. The process of calculating the prediction for an active user whose rating was not predictable by (3) is first calculated by (4) to find the probability:

$$prob_{ij}^{k} = \max((\tau_{ij})(\mu_{ij})) \tag{4}$$

Where, $\tau_{i,j}$ is the trust rate of user i to the user j. $\mu_{i,j}$ is obtained through the following equation:

$$\mu_{ij} = {}^{1}\!/_{d_{sj}} \tag{5}$$

Where, d_{sj} identifies the distance between the active user's node and j. As in other approaches, in this approach also, all distances between the active user node and j that are greater than 3, are considered 3. This is because of the complexity of the algorithm for calculating d_{sj} . After obtaining the probabilities, these values should be sorted out in the descending order of TF(S), the list of trusted friends [9].

Calculating these values, the prediction will be calculated using (6), known as Resnick equation [11].

$$r_{i,it} = \bar{r}_i + \frac{\sum_{j=1}^{top-u} \tau_{i,j}(t)(r_{j,it} - \bar{r}_j)}{\sum_{j=1}^{top-u} \tau_{i,j}} \tag{6}$$

Where, $r_{j,it}$ and $r_{i,it}$ are the ratings of the node i and the node j to the item i_t ; $\bar{r}_j$ and $\bar{r}_i$ are the average ratings of the user i and j; $\tau_{i,j}$ is the trust rate of the user i to the user j. top – u is n number of users from TF(S) based on which the prediction is done.

D. Updating Pheromone

The purpose of updating Pheromone in the ant colony algorithm mentioned above is to increase Pheromone values for edges that end in trusted friends. Therefore, gradually, the edges or routes with higher Pheromone values are selected with higher possibility than the edges with lower Pheromone values. Accordingly, the process of updating Pheromone has been demonstrated in (7):

$$\tau_{ij}(t) = (1-\rho)\tau_{ij}\,(t-1) + \Delta Q \tag{7}$$

Where, ρ is a constant value that indicates Pheromone evaporation to prevent unlimited Pheromone aggregation. ΔQ is a small value obtained from (8):

$$\Delta Q = \frac{\prod_{i=1}^{d_{sj}} \tau_{ij}(t-1)}{d_{sj}} \times \frac{T-traced_j}{T-unrated_s} \tag{8}$$

$\prod_{i=1}^{d_{sj}}$ represents the transfer of trust Pheromone from the target user node (S) to node j. d_{sj} indicates the connection level of the target user node (S) to the node j. $T - traced_j$ indicates the number of times that user j participates in the process of predicting the rating and the rating of this user is used. $T - unrated_s$ indicates number of unrated items by the target user.

The more d_{sj} increases, the greater distance between the user node and the target user node which can reduce the value of $\frac{\prod_{i=1}^{d_{sj}} \tau_{ij}(t-1)}{d_{sj}}$ and the value perceived from this equation is less trusted.

E. Recommending to the Target User

Finally, the proposed approach, based on the predicted ratings, recommends n items with the highest rating (TOP-N) as the target user's favorite items.

III. EXPERIMENT

This section shows the results of measuring the efficiency of the proposed approach (TBRSK) with several experiments and compares them with other approaches, including Trust Aware Recommender Systems (TaRS) [12], User Based and Item Based KMCF, and TRACCF. The datasets, the evaluation metrics, and the clustering techniques are explained and discussed. The experiments were done on a system with CPU Core i7 2.5GHz and 16 GB RAM. Moreover, all of the methods were implemented using MATLAB.

A. Dataset

In this research, Epinion and MovieLens datasets were used. Epinion is a product review website that started in 1999 (www.epinion.com). In this website, users can rate items from 1 to 5 and submit their personal reviews.

Users can also express their web of trust. The extracted dataset contains 13,668,319 ratings on 1,560,140 products submitted by 132,000 users. The subset in our experiment is from this dataset and it includes 500 products purchased by 5,000 customers, with the items of highest ratings.

Another dataset used in this experiment is MovieLens. This dataset is the original dataset prepared by the Group lens Research Group at the University of Minnesota, and is known for evaluating the recommendation algorithms. This dataset contains 10,000 ratings from scale 1 to scale 5 of 1,682 films by 943 users, and each user has rated at least 20 films.

B. Evaluation Metrics

There are a few evaluation metrics for recommender systems, and they are classified into two main groups: accuracy metrics and coverage metrics [13]. Accuracy metrics focus on how a system can predict the exact rating value of a specific item. The accuracy-based methods include Mean Absolute Error (MAE), Root Mean Square Error (RMSE), Precision and Recall. Choosing the metrics to evaluate recommender systems depend on the purpose of the system. In this paper, RMSE was used to evaluate and compare the accuracy of the proposed approach with other approaches.

For measuring the accuracy metrics, Mean Absolute Error (MAE) and Root Mean Square Error are usually used. MAE only considers the absolute value of the difference of the predicted and real ratings, but RMSE squares the error before summing.

$$RMSE = \sqrt{\frac{1}{N} \cdot \sum_{u,i} (p_i(u) - r_i(u))^2} \tag{9}$$

Where, N is the total number of all ratings of the users; $p_i(u)$is the predicted rating for the user u in the item i, and $r_i(u)$ is the real rating. To have lower values of RMSE, better predictions are required.

Another type of criteria in recommended systems is coverage which can be defined as "the percentage of a dataset that the recommender system is able to provide prediction for" [13].

C. K-means Clustering

The dataset is divided into two parts of training and testing (80% training and 20% testing). In first step, the dataset is divided into five-fold cross-validation subsets, as in [14], [15]. Each 80% range of the dataset is used as training and the remaining 20% is used for testing. Therefore, in each five testing experiments, there will be four training subsets and one testing subset, in a way that the training subsets do not overlap. Testing subsets do not have any overlaps as well, and in total, they make the original rating matrix. Thus, there will be five different results based on five different testing subsets, and the average of these results will be considered.

IV. Result

In this section, our purpose is to analyze the performance of our proposed method, TRBSK.

Based on previous results (II-A), we measured the similarity between the users u and v using Pearson Correlation Coefficient. We accepted a trust coefficient, and analyzed the datasets Epinion and MovieLens with 80% training and 20% testing. Number of clusters were 15 (k=15). The experimental parameter settings are listed in Table 1.

We tested five different methods: KMCF (user-based & item-based), TaRS, TRACCF, and the proposed TBRSK algorithm. The parameter ρ is the evaporation rate of Pheromone was set to 0.4. Top-U values of 10, 15, 20 and 30 were used.

Table 2 is the results of running the proposed approach on MovieLens dataset using two evaluation metrics and compares the result with KMCF (user-based and item-based), TRACCF and TaRS. The results show that compared to other approaches, the proposed approach has the least number of RMSEs while having the best Coverage Value.

Table 3 shows the results of using the proposed approach on Epinion dataset.

TABLE I. Experimental Parameter Setting

σ	ρ	TOP-U
[0,1]	0.4	10,15,20,30

TABLE II. MovieLens Dataset

Algorithms	RMSE	Coverage[%]
KMCF(ItemBased)	1.2982647	94.551
KMCF(UserBased)	0.923775	93.716
TRACCF	0.81	97.184
TaRS	0.814716	96.918
TBRSK	**0.699822**	**99.946**

TABLE III. Epinion Dataset

Algorithms	RMSE	Coverage[%]
KMCF(ItemBased)	1.1001171	97.251
KMCF(UserBased)	0.599789	93.281
TRACCF	0.61	96.811
TaRS	0.612315	97.751
TBRSK	**0.5798225**	**100.00**

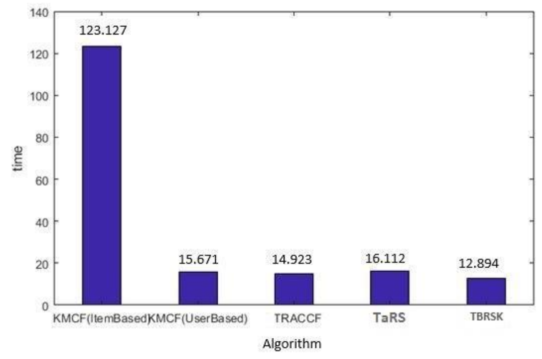

Fig. 1. Time costs of different algorithms in MovieLens dataset.

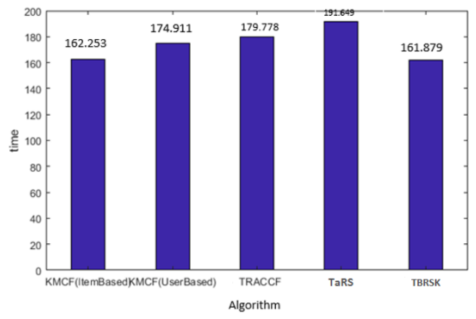

Fig. 2. Time costs of different algorithms in Epinion dataset.

The results of Fig. 1 and 2 shows that time duration in the proposed approach are better than other approaches. That is, the proposed approach offers better results in less time.

Table 4 presents the results of measuring RMSE in several Top-Ns in Epinion dataset. The results show that in all Top-N values, the proposed approach has the least RMSE compared to other algorithms, and for low values of Top-N (Top5 and Top10), the proposed approach shows better RMSE.

TABLE IV. Epinion RMSE for Different Algorithms

Algorithms	Top10	Top15	Top20	Top30
KMCF(item-based)	1.100171	1.100171	1.100172	1.100173
KMCF(user-based)	0.599789	0.599787	0.599789	0.599791
TRACCF	0.609999	0.609999	0.610001	0.610001
TaRS	0.612311	0.612313	0.612318	0.612321
TBRSK	**0.579821**	**0.579821**	**0.579823**	**0.579825**

TABLE V. MovieLens RMSE for Different Algorithms

Algorithms	**Top10**	**Top15**	**Top20**	**Top30**
KMCF(item-based)	1.298265	1.298265	1.298264	1.298265
KMCF(user-based)	0.923775	0.923775	0.923777	0.923774
TRACCF	0.809999	0.809999	0.810001	0.810001
TaRS	0.814711	0.814714	0.814718	0.814721
TBRSK	**0.699821**	**0.699821**	**0.699823**	**0.699824**

The results of the experiment in Tables 2 and 3, and Fig. 1 and 2, are demonstrating algorithm's runtime, coverage and RMSE. Most trust-enhanced recommendation algorithms only analyze trust among users, without considering the interests and requests of users. This will improve coverage with high accuracy. However, our proposed method offers a trust relationship consisting of trust degree and users similarities, in order to increase accuracy and coverage and reduce time.

Tables 2 and 3 show the comparison of coverage rate between our proposed method and KMCF, (TaRS), and TRACCF methods. As it is shown, our proposed algorithm has a better coverage rating, because if users' ratings cannot be calculated based on trust, they would be calculated through ant colony algorithm. This will increase coverage rate and reduce RMSE.

Tables 4 and 5 show the RMSE rate for different top-Ns. Compared to the rest of the algorithms our proposed algorithm has the lowest RMSE value.

V. Conclusion

Due to high volume of information in most systems like online stores, social networks, etc., users face many difficulties finding items. To avoid this issue and reduce the searching time to find desired items, recommender systems should filter information. Recommender systems examine priorities of users that have previously ranked items and recommend the best items. This will reduce the time, and assist users in finding their desired items in a huge database. CF methods used to recommend items for a target user based on the items ranked by similar users. Such systems usually find similar neighbors to the target user. Contrary to previous methods that attempted to find similar neighbors with the target user, trust-based CF approach attempts to create a neighborhood of the users' trust network. These systems recommend items based on the trust relationship between the users.

The proposed method in this paper is a mixture of similarity-based and trust-based methods that will first calculate users' similarities and trusts and then predict the ranks by mixing the two methods. If this method is not able to predict the ranks due to data dispersion and/or cold start, an ant colony-based algorithm is used to predict the rankings. This will increase the coverage rate of the proposed algorithms compared to other algorithms. Having calculated the ranks, K-Means method (Section III-C) is used to reduce the overlaps. To verify, the proposed method was compared to several other methods using Epinion and MovieLense datasets, and RMSE as the evaluation criterion.

The results clearly show that the proposed method has an acceptable coverage rate and low RMSE, due to using ant colony algorithm that does not have the common problems of recommender systems.

References

[1] H. Liang, Y. Xu, Y. Li and R. Nayak, "Personalized recommender system based on item taxonomy and folksonomy", Proceedings of the 19th ACM international conference on Information and knowledge management - CIKM '10, 2010.

[2] C. Hwang and Y. Chen, "Using Trust in Collaborative Filtering Recommendation", New Trends in Applied Artificial Intelligence, pp. 1052-1060, 2007.

[3] H. Kaur and D. Jain, "Optimizing the Number of Neighbors in Trust Based Recommender Systems.," Journal of Comput Science Issues, vol. 10, no. 4, pp. 230–238, 2013

[4] P. Victor, N. Verbiest, C. Cornelis, and M. D. E. Cock, "Enhancing the Trust-Based Recommendation Process with Explicit Distrust," ACM Trans Web, vol. 7, no. 2, pp. 1–19, 2013.

[5] X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," Advances in Artificial Intelligence, vol. 2009, pp. 1–19, 2009.

[6] M. Norwati, V. Wong Pei, and S. Nasir, "User recommendation algorithm in social tagging system based on hybrid user trust," Journal of Computer Science, vol. 9, no. 8, p. 1008, 2013.

[7] L. Shao, J. Zhang, Y. Wei, J. Zhao, B. Xie, and H. Mei, "Personalized QoS Prediction forWeb Services via Collaborative Filtering," IEEE International Conference on Web Services (ICWS 2007), 2007.

[8] C. Birtolo and D. Ronca, "Advances in Clustering Collaborative Filtering by means of Fuzzy C-means and trust," Expert Systems with Applications, vol. 40, no. 17, pp. 6997–7009, 2013.

[9] P. Bedi and R. Sharma, "Trust based recommender system using ant colony for trust computation," Expert Systems with Applications, vol. 39, no. 1, pp. 1183–1190, 2012.

[10] R. Burke, "Hybrid Web Recommender Systems," The Adaptive Web Lecture Notes in Computer Science, pp. 377–408, 2007.

[11] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An Open Architecture for Collaborative Filtering of Netnews," Proceedings of the 1994 ACM conference on Computer supported cooperative work - CSCW 94, 1994.

[12] P. Massa and P. Avesani, "Trust-aware recommender systems," Proceedings of the 2007 ACM conference on Recommender systems - RecSys 07, 2007.

[13] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, "Evaluating collaborative filtering recommender systems," ACM Transactions on Information Systems, vol. 22, no. 1, pp. 5–53, Jan. 2004.

[14] C.F. Tsai and C. Hung, "Cluster ensembles in collaborative filtering recommendation," Applied Soft Computing, vol. 12, no. 4, pp. 1417–1425, 2012.

[15] A. Bilge and H. Polat, "A comparison of clustering-based privacy-preserving collaborative filtering schemes", Applied Soft Computing, vol. 13, no. 5, pp. 2478-2489, 2013.