International Journal of Advances in Engineering Research http://www.ijaer.com
(IJAER) 2012, Vol. No. 4, Issue No. VI, December **ISSN: 2231-5152**

TRUST AWARE RECOMMENDER SYSTEM ALLEVIATING SPARSITY AND SCALABILITY PROBLEM IN COLLABORATIVE FILTERING

Latesh Chaudhary

ABSTRACT

To deal with many everyday choices that the modern life offers, we seek out opinions of our knowledgeable friends about things we are not sure about. In the web domain, several commercial and social networking sites employ many user and item filtering techniques to recommend items (i.e. products and services) to its users. Such techniques are known as Recommender Systems. Various techniques have been proposed for performing recommendations which include content-based filtering, collaborative filtering, hybrid and others. Collaborative filtering technique which is based on matching users having similar preferences is the most popular technique used by many websites offering recommendations to its users. But it is noted that this recommender technique is limited by the problem of sparsity and scalability. It is our attempt to suggest a new technique of incorporating trust in the Collaborative Filtering Recommender System by first partitioning the user database using clustering, on the basis of trust values assigned by users to each other, in order to get a dense group of similar users and then applying the trust values obtained between the users of that group to calculate the similarity of their preferences. Once the users have thus been matched on the basis of trust, we can generate the recommendations for the current user who is the nearest neighbor of this particular partition.

RECOMMENDER SYSTEM

We have all the comforts in the world today. Our homes are full of items and gadgets like Refrigerators, Air Conditioners, Music System, Television, Microwave, etc. which make our life much more easier and relaxed. We have Mobile Phones that help us to stay connected anywhere, anytime. We have Computers with internet connection that brings the world wide knowledge literally on our fingertips. With the help of Internet distances don't matter and we can cull any information from any part of the world on host of topics ranging from the field of Agriculture to Astronomy or from Music to Medicine, etc. We are living in a world where our wants are limitless and possibilities are endless.

Yet modern lifestyle has its own pitfalls and attendant problems. The biggest problem that we face today is the problem of choice. Choices available to us over the World Wide Web are limitless and so alluring that it is almost impossible to come to an intelligent decision. Whatever we may want to do or whatever we may want to buy, there seems to be a deluge of choices available to us. Along with the problem of choice comes the problem of trust. A layman with no technical knowledge desirous of buying a simple household item like an HDTV is confronted with so much

detailed and in-depth technical jargon about it over the internet that a decision to buy a simple TV becomes a scary scenario, to say the least.

To resolve the above problems, Recommender Systems have been developed by various websites these days. These systems are meant to alleviate the tedious task of browsing through endless web pages in order to find the information relevant to us [3]. Our specific problems are resolved by these websites by: gathering information about the like-minded people in a database by creating user profiles, which is a representation of their likes and dislikes, gathering information about the items and services and then developing a large database of these items or services. This database can now be used by the Recommender System to generate advice or recommendations for a new user with similar likes and dislikes. Hence, a recommender system is able to deliver personalized, interesting, useful and sometimes profitable information to the user based on the analysis of her past behavioral patterns.

The two most popular Recommendation strategies that are widely used nowadays are, Content based and Collaborative Filtering. In Content Based approach, items which the users have accessed in the past are used to build up user profiles. Based on these user profiles, the system recommends only the data items that are highly relevant to the user profiles by computing the similarities between the data items and the user profiles. It heavily relies on the rich content descriptions of the items [12]. This approach suffers if user profiles fail to describe their interests adequately [3]. Also reliability of the recommender suffers if enough items are not rated by the user. This technique is at the mercy of the descriptive data available about the content and is limited by the features that are explicitly associated with the objects [19]. This technique in order to be effective requires extensive manual work in terms of tagging and classifying items to make them feature rich [10].

Collaborative Filtering (CF) technique employs a different approach to the whole problem of providing effective recommendations. The basic idea behind the CF approach is that in the times of need we generally go to our friends for the right advice [18]. In a way, CF approach computes the similarities between the user profiles based on the premise that users who agreed in the past i.e. rated similar items or services highly, are more likely to agree in the future too. Hence, in this approach the system recommends items to the user by matching her personal preferences or tastes with that of other users in the system. CF is found to be the most successful recommender approach to date and is used in many successful recommender systems on web [17].

However CF has a few problems of its own like the *cold start problem* wherein an insufficiently rated item is not recommended by the system [20] or the *small user base* of the website leads to ineffective recommendations [8] or the *popularity bias* wherein the more popular items are rated quickly and by more users, hence they are recommended the most [9]. But the two biggest drawbacks of CF technique are the problems of *sparsity* and *scalability* [15]. In order to create good quality recommendations CF should be able to compare customers who have rated at least two products in common. But it so happens that many pairs of customers have no correlation at

all as they have rated only few items from among the millions of available items [4]. Hence, in this scenario, CF algorithms will be unable to make any useful recommendation for a particular user. This problem is known as reduced coverage, and is due to *sparsity* of matching users. On the other hand, the effectiveness of the recommender system depends on the speed with which recommendations are generated for the current user, but with millions of customers and products, a typical web based recommender system running existing algorithms will suffer serious *scalability* problem [15].

We are going to propose a recommender system which can help resolve the above problems, namely, sparsity and scalability and also generate better recommendations by incorporating trust metrics.

DEFINING TRUST

Given a choice we trust the word of mouth recommendations of our friends or our friend's friends. This is so because we still credit the recommendation given by the network of our friends rather than some impersonal automatic machine generated advice [18]. Machine generated recommendations are suspect because the user is not aware who she is being matched with and how similar they really are. Do the neighbors in collaborative approach really have similar tastes and preference or they are simply yoked together by some computer algorithm. Due to the lack of the human element in the recommender system we still prefer the advice of those we know personally [4]. It is because of the above factors that the need for trust aware recommender system was felt.

Trust in an online setup can be treated as the opinion expressed about one user by the other user be it in terms of ratings or decision that such and such user is my friend or favourite. For example, a recommender system can ask the users to vote for or rate the users they trust and in this way the various votes for a particular user can be combined to create trust metrics for that user. In this manner a web or network of trust can be created with users who trust each other and recommendations can be generated from among this web or network. In this scenario it is also possible to predict the trustworthiness of an unknown user by propagating that trust. That is, if the user X trusts the user Y and the user Y trusts the user Z then it is possible to calculate the extent to which X could trust Z [11].

BACKGROUND AND PREVIOUS WORK

Many good recommender system methods, algorithms and models using CF employing the concept of trust have been proposed so far. For example, Bedi and Banati [1] proposed a model wherein they calculate trust, based on the features of the website to assess the usability of the site, thereby eliminating the use of questionnaires and surveys to determine the trust between users. O'Donovan and Smyth's hybrid recommender model [7] operates at a profile level and at the profile-item level by employing trust weighting mechanism to improve the recommendation quality. Bedi and Kaur [2], using Intuitionistic Fuzzy Sets, have in their model tried to reduce the problem of

lack of trust by incorporating social recommendation process wherein trustworthy peers become the recommender agents for the users according to user's specific tastes. Papagelis et al. [13] in their model have tried to reduce the sparsity problem (outlined in Section 2 above) by using trust inferences arrived at by propagation of trust between users through trust paths.

Although all the above models of Recommender Systems deal with and try to counter the problems associated with Trust based recommender system but they, in our opinion, have not dealt specifically with the problem of sparsity and scalability. The focus of our paper is to propose a recommender system which keeps the end user in focus and tries, by introducing various layers of refinement at all levels, to arrive at useful, actionable and effective recommendations for her. For our purpose we have extensively used the model proposed by Massa and Avesani [11] and tried to refine it by using the Clustering technique of data mining to create partitioned user database to arrive at highly focused recommendations for our user thereby eliminating the problem of sparsity and scalability in our recommender system.

EMBEDDING TRUST IN RECOMMENDER SYSTEM

In our work, we are going to outline a new method of generating more accurate and relevant recommendations by incorporating trust at two levels i.e. (a) at the level of clustering and then again (b) at the level of generating recommendations. Then we are going to suggest a further refinement by using data gained from item based collaborative filtering to further match the recommended items and come up with more broad based but relevant recommendations.

1 Embedding Trust at Level of Clustering

By employing trust values at the level of clustering we can ensure that the resulting groups contain only those users who have shown high level of trust in each other. This method tends to minimize the problem of both sparsity and scalability as items and user database are reduced to suit the need of the current user and the system doesn't concern itself with any irrelevant data at all.

Partitioning of User database

Data clustering is the method in which items are grouped together on the basis of some similarity. It is a process in which the given data is subdivided into groups of meaningful or useful groups known as clusters [6]. Clustering techniques work by identifying groups of users who appear to have similar preferences. Once the clusters are created, predictions for an individual can be made by averaging the opinions of the other users in that cluster [16]. First, we are going to partition the users of a collaborative filtering system using a clustering algorithm and use those partitions as neighborhoods [15]. In our approach we are going to use the trust values given by users at this level to partition the user database. A collaborative filtering algorithm using this idea first applies a clustering algorithm on the user-item database to divide the database into **p** number of partitions of varying size on the basis of trust metrics. In the next step, the neighborhood for the active user $\mathbf{u_a}$ is

selected by looking into the partition in whose users she has shown very high level of trust. The selected partition, A_i is then used as the neighborhood for that active user i.e. $\mathbf{u_a}$.

This method has two benefits—first, it reduces the sparsity of the dataset as a particular user i.e. $\boldsymbol{u_a}$ is now part of a neighborhood of similar users and second, because of the dimensionality reduction due to the partitioning of the large user database and with the use of a static pre-computed neighborhood, the prediction generation is much faster i.e. scalable.

Applying Trust Metrics on the Partitioned User Database

Once the user database has been partitioned on the basis of trust values, we can apply the trust metrics on the partition ($\mathbf{A_i}$) to which the current user ($\mathbf{u_a}$) belongs. To do this we have followed the modified version of the model proposed by Massa and Avesani [11]. Their model is applicable to the user database as a whole but in our case as we have already partitioned the user database on the basis of trust between each user to arrive at a partition with users most closely matching the active user ($\mathbf{u_a}$). Hence, we believe that our model is a step further in the process of arriving at better, relevant and more useful recommendations for the current user.

Although we have a partition for the current user with similar users as her neighbors but this partition can itself be very big and predictions based on the neighbors within the cluster will be no better than the traditional models of collaborative filtering technique. Here we apply the model proposed by Massa and Avesani [11], to our partition consisting of the neighbors of the current user. This model is composed of various modules and matrices as indicated by the following figure:

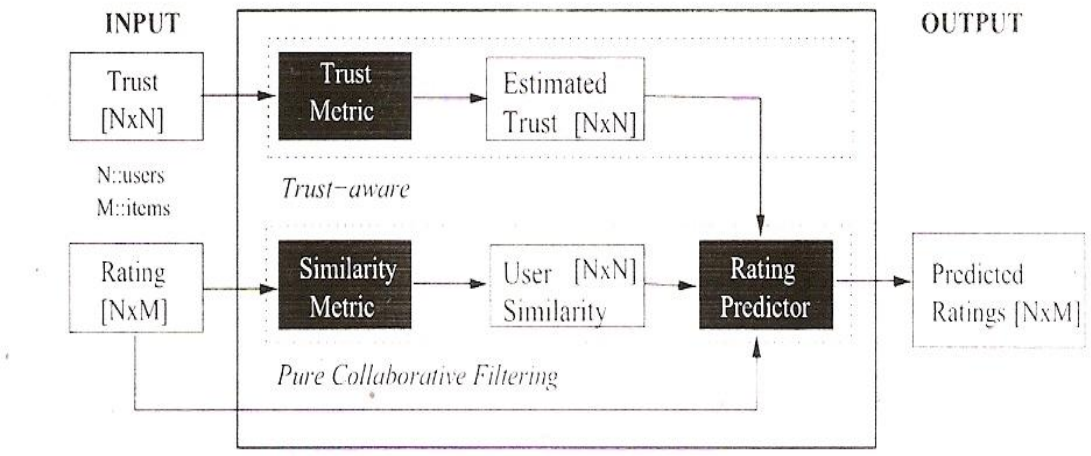

Figure 1: Trust-Aware Recommender Systems Architecture

The whole system takes as input all the trust statements of the users in the user's ($\boldsymbol{u_a}$) partition $\mathbf{A_i}$ in the form of a trust matrix. It also uses as input all the ratings given by all the users in the partition $\mathbf{A_i}$ in the form of a rating matrix. This system produces as an output a matrix of predicted ratings that users would assign to the items. This output matrix is used by the

recommender system to make recommendation of the most liked items to the user. All it does is to select from the row of predicted ratings relative to the user, the items with highest trust values. Let us now look at every single module and the underlying algorithm for the better understanding of the whole system.

Trust Metric module— The trust metric module takes as input the trust network, represented as **N x N** trust matrix (**Fig.2**), and exploits the trust propagation in order to predict, for every user, how much she could trust every other user. Hence, we get an Estimated Trust matrix. In this matrix the cell **[i, j]** represents the predicted trust value of the user $\mathbf{u_i}$ about user $\mathbf{u_j}$ Although Massa and Avesani do use the concept of trust propagation in their model but herein lies the difference between their model and ours. Their model takes the whole user database at once hence it is necessary for them to consider the distance of every other user from the source users as an important factor. In our model, as we had already partitioned the user database hence we are dealing with only those users who are already close to each other. Hence, we don't have to fix a maximum propagation distance in our computation of trust values.

$$\begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & \cdots & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & \cdots & \cdots & a_{2,n} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & \cdots & a_{i,j} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{n,1} & a_{n,2} & \cdots & \cdots & \cdots & a_{n,n} \end{bmatrix}$$

Fig. 2: N X N Trust Matrix

Similarity Metric Module— Computing the similarity of the current user against every other user is one of the basic steps of Collaborative Filtering technique. The task of similarity metric is to compute the correlation between two users and produce the output as **N x N** User Similarity matrix in which $\mathbf{i^{th}}$ row contains the similarity values of the $\mathbf{i^{th}}$ user against every other user. This correlation value is used as a weight for the user ratings, with the implicit logic, that if a user rates in a way similar to the current user then her ratings are useful in predicting the ratings of the current user. The most popular technique to arrive at a correlation value is Pearson Correlation Coefficient [14].

$$w_{a,u} = \frac{\sum_{i=1}^{m} \left(r_{a,i} - \bar{r}_a\right)\left(r_{u,i} - \bar{r}_u\right)}{\sqrt{\sum_{i=1}^{m} \left(r_{a,i} - \bar{r}_a\right)^2 \sum_{i=1}^{m} \left(r_{u,i} - \bar{r}_u\right)^2}}$$

where the **a** stands for the current user whose correlation with the user **u** is being calculated for the item **i**.

It should be kept in mind that this correlation only works on overlapping items. If two users have only one item rated by both, then the correlation is not meaningful. This fact is not a limitation for the users of our model because due to the partitioning of the user database, in our model, only the closely matched users are part of the cluster we are working with. Also we can always modify the partitioning algorithm to include only those users in our partition who have rated a specified number of items.

Rating Predictor— This step is a usual last step of Collaborative Filtering. The predicted rating of the item **i** for the current user **a**, is the weighted sum of ratings given to the item **i** by the **k** neighbors of **a** [5]:

$$p_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^{k} w_{a,u}\left(r_{u,i} - \bar{r}_u\right)}{\sum_{u=1}^{k} w_{a,u}}$$

Here $\mathbf{p_{a,i}}$ gives us the predicted rating for the current user *a* for the item ***i***. Neighbors can be taken from the User Similarity matrix or from the Estimated Trust matrix and the weights $\mathbf{w_{a\,u}}$ are cells in the chosen matrix.

Another option for us is to combine the two matrices in order to produce an output matrix that embeds both the information i.e., user similarity and the estimated trust. Usually these matrices are very sparse, hence this strategy could be very useful in reducing sparseness and thus providing more neighbors for every single user.

Matching Recommended Items with Other Similar Items

Once we get enough recommendations by embedding the trust metrics as outlined above we can further refine these recommendations by matching each recommended item with the other items in the cluster to which that recommended item belongs. Suppose the required number of recommended items is 10. Now by embedding trust at the level of clustering and then further adding trust at the level of generating recommendations, as explained in section 4.1.1 and 4.1.2 already, we can generate 5 recommendations. Our suggestion is that to improve variety in our recommendations so that they don't contain just the most popular items but better reflect the taste of the current user and also provide her with something genuinely new, we look for the closest matching item for each of the 5 items in its own item cluster. In this way we will be able to pick one more item each for already chosen 5 items. Hence we will have the required number of 10 recommendations.

CONCLUSION

We have seen in this work how with the application of trust metrics we can effectively utilize a data mining technique i.e., clustering to limit the very large database of both the users and the items to more relevant trust-based partitions. Then by further re-application of trust metrics on these partitions based on the specific user's preferences or interests we can arrive at pertinent and effective

recommendations for her. Our proposed solution can be seen as a step by step refinement technique where first, we employ trust metrics to find the relevant group of users or items based on our current user's tastes and preferences and second, we re-introduce trust metrics to this group to further limit our field of calculation to relatively fewer but most closely matched set of users on the basis of their trust values in relation to the trust shown by the current user. As an additional refinement, to make the recommendations more broad based, we match the recommended items with the most similar items using item-based collaborative filtering technique.

We admit that this is just a proposal or outline for generating recommendations using trust at two levels. We sincerely hope that our effort at the suggestion of this new technique of generating recommendations by introducing trust factor at the level of clustering and then again in the clustered group to generate top N recommendations for the current user will prompt others to work towards refinement and implementation of the technique outlined here.

REFERENCES

[1] Bedi, P. and Banati, H. (2006) 'Assessing User Trust to Improve Web Usability.' In Journal of Computer Science 2(3), 283-287

[2] Bedi, Punam and Kaur, Harmeet. (2006) 'Trust based Personalized Recommender System.' INFOCOM Journal of Computer Science, 5(1): 19-26.

[3] Bilgic, M. (2004) 'Explanation for Recommender Systems: Satisfaction vs. Promotion'. Undergraduate Honor Thesis, University of Texas at Austin, 1-24, May.

[4] Bonhard, P. (2005) 'Who do trust? Combining recommender systems and social networking for better advice' In Proceedings of the Workshop Beyond Personalization 2005, in conjunction with the International Conference on Intelligent User Interfaces IUI'05, San Diego, CA, pp. 89-90.

[5] Eaton, A. (2004). 'RVW model for syndicating and aggregating reviews.' http://www.pmbrowser.info/rvw/0.2

[6] Fung, Glenn. (2001) 'A Comprehensive Overview of Basic Clustering Algorithms.' IEEE, June 2001, 1-37

[7] O' Donovan, J and Smyth, B. (2005) 'Trust in Recommender Systems Proceedings of the 10th international conference on Intelligent user interfaces, 167-174.

[8] Hendler, J. and Golbeck, J. (2004) 'Accuracy of Metrics for Inferring Trust and Reputation.' In Proceedings of 14th International Conference on Knowledge Engineering and Knowledge Management, October 5-8, Northamptonshire, UK.

[9] McLeod, D. and Chen, A. (2005) "Collaborative Filtering for Information Recommendation Systems", Encyclopedia of Data Warehousing and Mining, Idea Group.

[10] Massa, P. (2003) 'Trust-Aware Decentralized Recommender System: PhD research proposal.' Department of Information and Communication Technology, University of Trento, 29th May.

[11] Massa, P. and Avesani, P. (2004) 'Trust-aware Collaborative Filtering for Recommender Systems.' In Proceedings of the International Conference on Cooperative Information Systems, (CoopIS).

[12] Melville, P., Mooney, R. and Nagarajan, R. (2001) 'Content-boosted collaborative filtering'. In P.Melville, R.J.Mooney, and R.Nagarajan. Content-boosted collaborative filtering. In Proceedings of the 2001

[13] Papagelis, M., Plexousakis, D. and Kutsuras, T. (2005) 'Alleviating the Sparsity Problem of Collaborative Filtering Using Trust Inferences.' Proceedings of the 3rd International Conference on Trust Management, iTrust.

[14] Pearson K. (1900) 'Mathematical contribution to the theory of evolution: VII, on the correlation of characters not quantitatively measurable'. Phil. Trans. R. Soc, London, 1-47.

[15] Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. (2001) 'Item-based Collaborative Filtering Recommendation Algorithms.' Proceedings of WWW.

[16] Sarwar, B., Karypis, G., Konstan, J. and Riedl, J. (2002) 'Recommender Systems for Large-Scale E-Commerce: Scalable Neighborhood Formation Using Clustering.' In 5th International Conference on Computer and Information Technology (ICCIT).

[17] Shardanand, U. and Maes, P. (1995) 'Social information filtering: Algorithms for automating 'Word of Mouth'.' Proceedings of CHI '95, ACM, xx+598, 210-17.

[18] Sinha, R. and Swearingen, K. (2001) 'Comparing Recommendation made by Online Systems and Friends.' In the proceeding of the DELOS-NSF Workshop on Personalization and Recommender Systems in Digital Libraries, Ireland.

[19] Takeuchi, Y. and Sugimoto, M. (2006) 'City Voyager: An Outdoor Recommendation System Based on User Location History.' UIC, 625–636.

[20] Taschuk, Morgan. (2007) 'A Hybrid Knowledge-based/Content-based Recommender System in the Bluejay Genome Browser.' Undergraduate Honours Thesis, University of Calgary.