Conceptual Framework for Recommendation System Based on Distributed User Ratings

Hyun-Jun Kim[1], Jason J. Jung[1], and Geun-Sik Jo[2]

[1] Intelligent E-Commerce Systems Laboratory, School of Computer Science & Engineering, Inha University, Incheon, 402-751, Korea
{dannis,j2jung}@eslab.inha.ac.kr

[2] School of Computer Science & Engineering, Inha University, Incheon, 402-751, Korea
gsjo@inha.ac.kr

Abstract. A recommender system is an automated collaborative filtering system that helps users to decide what they are interested in by extracting their own preferences. Most studies related to recommendation system have focused on centralized environment causing several serious problems such as fraud rating and privacy issues. In this paper, however, we propose the distributed recommender system with FOAF on P2P environment. This system can recommend users without the centralized server, which keeps updating their profiles. In order to find out the most eligible users to be recommended, user grouping (selecting objects) is one of the most important processes in the whole recommendation. Thereby, we have exploited cosine-based similarity method to cluster the users in the same level. More importantly, RFR (Recommend-Feedback-Rerecommend) algorithm is suggested to improve the confidence of recommendation. For the experiment, we have used the *MovieLens* datasets and have tested the performance by using “F1-measure" and Mean Absolute Error (*MAE*). As a conclusion, we have improved the robustness of the system. Also, we have shown the possibility of distributed recommender system on semantic web.

1 Introduction

As the Internet infrastructure has been developed, recommender systems have been issued as one of solutions for information retrieval. A recommender system on E-commerce and Information retrieval has brought reduction of cost for searching by extracting information in which users are interested and listing the result that are likely to fit the user. The computation for recommendation is based on stochastic process which sparse and incomplete data, so that the result can be either correct or occasionally very wrong [1]. At the same time, in order to recommend some information to user, the system has to learn the user preference in advance [2] and this causes privacy issue. These conditions are regarded as a problem that should be solved for modeling recommender system more efficiently. To solve these problems and to increase the performance of the recommender system such as robustness, *MAE* and “F1-measure”, researchers have suggested a number of alternative approaches using various techniques [3], [4]. We will present some of the researches in next section.

In this paper, we propose a Peer-to-Peer recommender system based on FOAF ('Friend Of A Friend'). Due to the extensibility of the FOAF-based distributed environment, it is possible to aggregate information from other users who have the same interest. User preference is asserted as a property of the FOAF document format by learning user's behaviors. Once the system learns user preferences, it can classify users with respect to their own preferences on real time, and also, it can recommend items in which users are interested to the rest of them in the same group on P2P network. Furthermore, we can increase the robustness of recommendation using RFR algorithm that helps users to decide whether the recommendation is reliable or not by considering other user's rating results in the same group. Finally, we can get satisfactory result of robustness [5]. The outline of the paper is as follows. Section 2 concentrates on the recent and related researches, section 3 refers to the suggesting recommender system based on FOAF and we explain experimental result in section 4. We conclude this study with the future work in section 5.

2 Related Work

Recommender systems, regarded as a part of personalization technique [4], are mainly used in personal information systems and E-Commerce to recommend products to customers. *Tapestry* is one of the earliest systems of collaborative filtering-based recommender system [6]. The *GroupLens* system that developed by research group in University of Minnesota is based on rating for collaborative filtering of *netnews* in order to help people find relevant articles in the huge stream of available articles [7]. This system is also called ratings-based automated recommender system, because filtering is executed by the explicit opinions of people from a close-knit community [8]. There is an additional project, *MovieLens*, which is web-based recommender system for movies, based on *GroupLens* technology. They also offer an experimental data source and a framework for studying user interface issues related to recommender systems [9]. Meanwhile, there have been many studies to enhance privacy issues. In [10], John Canny tries to solve privacy problem with security technologies such as key sharing, encryption and decryption. [11] suggested *Local Profile Model* to protect personal profiles by separating personal profiles from centralized server to each user's PC. Although it can protect draining of personal profile, but the system is working on centralized server, so it is still weak in fraud rating and malicious server attack.

3 Distributed Recommender System Based on FOAF

Recently, information integration and standardization on Semantic web environment have been emerging important field [12]. Many studies have been developed using RDF and XML. FOAF [13] derived from RDFweb makes users possible to link themselves to friend of a friend by using RDF-based document for user profile. When a user publishes a document for some information with FOAF, machines are able to make use of that information, and also users can make their own friends' network by interlinking FOAF. Our recommender system is based on FOAF under distributed environment such as P2P to increase recommendation performance. The recommender system extracts each user's preference that is represented by FOAF document format. By the preferences of each user, recommender system groups users. And then, when a user evaluates an item m_i, the system determines whether m_i is good enough to recommend to another users in the same group or not. Once the system decides to recommend m_i to other users, the recommendation is only applied to level n. We can define the 'level' as the depth of friends to be known by the user directly.

3.1 Extracting User Preferences

Before grouping users according to their preferences, the system needs to know about each user's preferences. In this paper, as used in *MovieLens*, we have distinguished movie genres into 19 categories. Usually a movie can be comprised in more than one category, at the same time, a user can have more than one preference about movie genre. We used *MovieLens*-like strategy to collect users' taste. Each page shows a list that consists of movies selected randomly from an ad hoc manual list of popular movies and the rest randomly from all movies [2]. In order to extract accurate preferences from each user, we employed a probabilistic method as shown Eq.(1).

Assume there are the set of users $U=\{u_1,u_2,\ldots,u_i\}$, the set of movies that is rated by a user u_i is $M=\{m_1,m_2,\ldots,m_j\}$ and the set of genre is $G=\{g_1,g_2,\ldots,g_k\}$. The function $f(P_{ik})$ extracts i-th user's preference about a genre k. This function

$$f(P_{ik}) = \sum_{j=1}^{n} \frac{g_k(m_j)}{\mu} R_{ij} \qquad \mu = \sum_{m=1}^{n} C_m(u_i, m_j) \tag{1}$$

where C_m is a set of genres included in a movie m_j and R_{ij} means a rating value of i-th user about a movie m_j. Also μ means total count of genres that included in movies that are rated by a user u_i.

3.2 User Grouping on Distributed Environment (Object Selection Procedure)

Distributed recommender system is basically impossible to group users at once. To enhance this shortcoming, there have been many studies clustering users on real-time [14]. Because each user' profile is distributed, the grouping of users must be processed at the same time with the recommendation. Although it takes more time to group users compare with centralized system, we can overcome some drawbacks of the system. The user grouping (object selection procedure) is processed on real-time, so it can make distributed system more dynamically, for example, when a new friend is added on a user or when a user's preferences are changed, we do not need to update any part of the system. And the place where users' information is gathering does not exist. Therefore, the distributed recommender system is less dependent on environment than centralized system. It also guarantees users' privacy [10]. Grouping of users must be done before recommendation. For the grouping of users, we used cosine-based similarity that is the most common way to compute the similarity between two users. The similarity between A and B can be calculated with each vectors of A and B, $Sim(A, B) = cos(\vec{A} \cdot \vec{B})$.

$$f(G_i) = \sum_{i+1}^{n} S(\sum_{i}^{n-1} S(u_i)) \qquad S(u_i) \geq T_{optimal} \tag{2}$$

As shown in Eq.(2), the grouping is calculated by the function $f(G_i)$ and it is a summation of the function S. where, $S(a)$ means a set of similar users of user a who has preference bigger than threshold $T_{optimal}$.

3.3 RFR (Recommend-Feedback-Re-recommend) Recommendation

On the process of a recommendation, a rating can be regarded as an essential factor for the quality of the system. But, although a user rates an item with good score, we cannot say the result is reliable, because it is just a user's subjective opinion. Therefore, for the quality of recommendation, we need to aggregate same group users' rating as many as possible. Therefore we focused on this level. We assume that a level is a group of a user who has known by a user directly.

$$Level_n = FriendOf(u_i) \qquad Level_{n+1} = FriendOf(Level_n) \tag{3}$$

Recommended user C_1 will take three pieces of information from recommender in $level_{n-1}$. They will be sent by FOAF document involved the name of an item m_j, rating result and the number of rater. The rating result is both a factor that can influence a recommended user to make their decision and a criterion that can compare with threshold for recommendation. Once a user rates an m_j, the rating result is also sent to members of $level_{n+1}$. We can consider three cases as shown in Fig. 1.

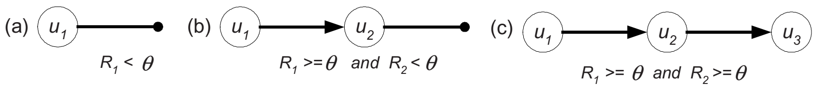

(a)
u_1
$R_1 < \theta$
(b)
u_1
u_2
$R_1 >= \theta$ and $R_2 < \theta$
(c)
u_1
u_2
u_3
$R_1 >= \theta$ and $R_2 >= \theta$

Fig. 1. Three cases of recommendation process by rating.

In Fig. 1(a), R_1 is a rating result from the user C_1. If R_1 is less than threshold θ, no recommendation is being made. And in Fig. 1(b) and Fig. 1(c), there can be two actions from the user C_2 according to their actions; 1) $C2$ rates m_j, 2) $C2$ do not rates m_j.

In case of 1), the system needs to update the rating result. But the case of 2), the system only needs to recommend another level. To update rating result on real-time, we suggested RFR (Recommend-Feedback-Re-recommend) system.

If U_1 on level 1 rates an item m_j that satisfies with threshold, the system recommends the m_j with rating result to C_1, C_2, and C_n on level 2. Due to the independency between users, each user on the same level cannot recognize another's rating result.

To update summarized rating result from all users on P2P network feedback process is needed. By this process, U_1 will take rating results from users on level 2. Feedback is a set of rating result from users, $Feedback=\{Rate(C_1),...,Rate(C_n)\}$. The first recommender, U_1 can calculate the updated rating result after feedback. Then, it will be re-recommended to C_1, C_2, and C_n, and these processes are applied to all levels (Eq.(4)). Therefore, as the level grows the rating result can be more reputable. The function *Re-recommend* is formulated as follows:

$$Re\text{-}recommend = \frac{Rate(User_i) + Feedback}{N + 1} \tag{4}$$

where N is the number of set of feedback.

4 Experiment

In this section, we analyzed performance of proposed system using “F1-measure” and *MAE*. We also examined performance of robustness against malicious action as level increases. Experiments were carried out on Pentium 1 GHz with 256MB RAM, running MS-Windows 2000 Server. The recommender system was implemented by Visual Basic 6.0, Active Server Page (ASP), and IIS 5.0.

4.1 Data Sets

We used *MovieLens* data sets to experiment suggesting recommendation system. *MovieLens* is a web-based research recommender system. The data set contains 1,000,000 anonymous ratings of approximately 3,900 movies made by 6,040 users who joined the site in 2000. For the experiment, we selected 450 users to use only about 66,926 rating dataset [9]. By users’ rating data, we can extract all users’ preferences. After we got the preferences, we made a matrix of user-user similarity that has 450 rows and 450 columns. Then, we also made the Most Similar Users (MSU) set by cosine-based similarity method.

4.2 Performance Measure

In this experiment, first of all, we employ “F1-measure” widely used in information retrieval community to find the optimal threshold $T_{optimal}$ for user grouping procedure. As shown in Eq.(5), “F1-measure” is calculated by *precision* and *recall*. So we defined *precision* as the ratio of *hit set* size to the *prediction set* size, and *recall* as ratio of *hit set* size to the MSU set size. Whereas, *prediction* is defined as the set of recommendation generated by the system and *hit set* is defined as the intersection of prediction set and MSU set which can be written as ‘$hit\ set = prediction\ set \cap MSU\ set$’. For example, if a user u_1 recommends an item m_j to u_2 and u_3, we can define *prediction set* as ‘$prediction\ (m_j) = \{u_1 \rightarrow u_2, u_1 \rightarrow u_3\}$’.

$$precision\ (P) = \frac{size\ of\ hit\ set}{size\ of\ prediction} \qquad recall\ (R) = \frac{size\ of\ hit\ set}{size\ of\ MSU\ set} \tag{5}$$

Also, for the measurement of the system's accuracy, we used *MAE* as shown in Eq.(6). Where N_p is defined as the number of *prediction*.

$$MAE = \frac{\sum |MSU - prediction|}{N_p} \tag{6}$$

4.3 Experimental Results

For the $T_{optimal}$ of user grouping procedure, we used "F1-measure" with 124 items. After we made FOAF network, recommended each items to users who are randomly selected. We carried out the experiment while changing threshold from 0.1 to 1.

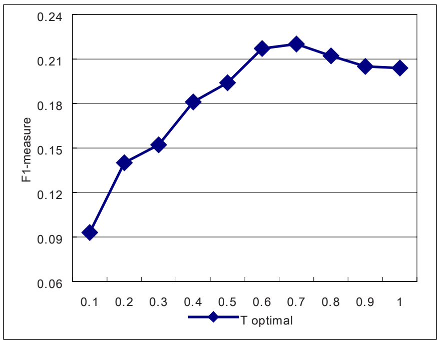
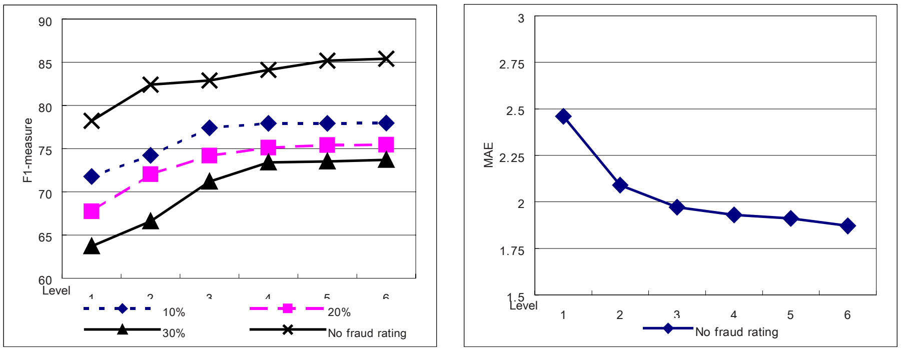


Fig. 2. Variation of the *F1-measure* as changing threshold, $T_{optimal}$.

In Fig. 2., when we set threshold to 0.7, it shows the best result, that is, this leads to the best performance for user grouping (object selection). When any users are recommended on FOAF network, the users propagate the recommendation to their friends who have similarity that is bigger than 0.7 of threshold.

Then, we have used *MAE* to show how the accuracy changes as level increase. In Fig. 3(a), we tested changing values through 6 levels. When we use a data set without any fraud rating, the result is continuously getting lower than former level, $level_{n-1}$. Since the number of users is exponentially increased as a level increase, the system can aggregate as many users' opinion. It means that even there are some of malicious ratings, users can take more accurate recommendation than centralized system as level increase. We have also shown the performance of the system in Fig. 3(b) using "F1-measure" to show robustness. As illustrated in graph, we involved some of malicious ratings in terms of 10%, 20%, and 30%. Generally, even when malicious ratings are included, system's performance is increased as level increase as we already show in Fig. 3(a). When there is no fraud rating, in Fig. 3(b), the performance evaluated by "F1-measure" is improved about 9.17%, and in case of 10%, 20%, and 30% of fraud ratings are included, the performance is improved by 8.02%, 11.35%, and 15.69% respectively.



(a) (b)

Fig. 3. Variant of *mean absolute error* (*MAE*) as level increase (a) and system performance measured by "F1-measure" in case of malicious ratings are included (b).

No matter how many fraud ratings are included in a rating data set, as level increase, the performance is improved 11.68% in terms of the three cases in average. Especially, when the percentage of fraud rating is increased, we can get more efficient improvement of the performance. As a result, these three fraud ratings brought only 9.71% of degradation of robustness, compared with original rating set (no fraud rating set).

5 Conclusions and Future Work

As a growing demand of distributed environment, we suggested P2P collaborative filtering system. The proposed system showed two major improvements. First, it can collect other user's ratings by RFR algorithm. So this helps users to make a decision effectively. Second, it is robuster than the centralized recommender system using clustering algorithms such as *k-NN*. When a malicious user tries to give wrong information to other users on the system, in fact, there has not been a way to protect. The proposed system, as shown in experiment Fig. 3(b), consists of P2P users can filter this malicious rating out. As increasing a level, users on the FOAF network can get filtered recommendation from their friends. Meanwhile, there still remain two major defects. First, we need to find a way to keep a consistency of the system. Because P2P is made of peer users, for the quality of performance, it needs as many users to rate items for recommendation. Second, *MAE* in single level is still bigger than the centralized system. Therefore we also need to improve this problem in future.

Acknowledgement. This research was supported by University IT Research Center Project (31152-01).

References

1. Herlocker, J. L., Konstan, J. A., Riedl, J.: Explaining Collaborative Filtering Recommendations. In Proc. of ACM 2000 Conf. on CSCW (2000)
2. Rashid, A. M., Albert, I., Cosley, D., Lam, S. K., McNee, S. M., Konstan, J. A., Riedl, J.: Getting to Know you: Learning New User Preferences in Recommender Systems. In Proc. of the 7th Int. Conf. on Intelligent User Interfaces (2002) 127–134
3. Resnick, P. and Varian, H.R.: Recommender Systems. Comm. of ACM 40, 3 (1997) 56–58
4. Schafer, J.B., Konstan, J.A., Riedl, J.: Recommender Systems in E-Commerce. In Proc. of the ACM Conf. on Electronic Commerce (1999)
5. O'Mahony, M., Hurley, N., Kushmerick, N., Silvestre, G.: Collaborative Recommendation: A Robustness Analysis. ACM Tran. on Internet Technology, Special Issue of Machine Learning for the Internet (2002)
6. D. Goldberg, D. Nichols, B. M. Oki, D. Terry.: Using collaborative filtering to weave an information tapestry. Comm. of the ACM, 35(12) (1992) 61–70
7. Rensnick, P., Iacovou, N., Suchak, M., Bergstrom, P., Riedl, J.: GroupLens : An open architecture for collaborative filtering of Netnews. In Proc. of ACM Conf. on Computer Supported Cooperative Work (1994)
8. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Item-Based Collaborative Filtering Recommendation Algorithms. In Proc. of the 10th Int. World Wide Web Conference (2001)
9. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Analysis of recommendation algorithms for E-commerce. In Proc. of ACM'00 Conf. on Electronic Commerce (2000) 158–167
10. Canny, J.: Collaborative filtering with privacy. In Proc. of the IEEE Sym. on Research in Security and Privacy (2002) 45–57
11. Sarwar, B. M., Konstan, J. A., Riedl, J.: Distributed Recommender Systems: New Opportunities for Internet Commerce. Chapter 20, Internet Commerce and Software Agents: Cases, Technologies and Opportunities. Idea Group Pubs (2001) 372–393
12. Davies, J., Fensel, D., Harmelen, F. A.: Towards the Semantic Web: Ontology-driven Knowledge Management. John Wiley & Sons, Ltd (2003)
13. http://rdfweb.org/foaf/
14. Nejdl, W., Wolpers, M., Siberski, W., Schmitz, C., Schlosser, M., Brunkhorst, I., Loser, A.: Super-Peer-Based Routing and Clustering Strategies for RDF-Based Peer-To-Peer Networks. In Proc. of the Twelfth International World Wide Web Conference (2003)