협동적 필터링에서 고품질 예측을 위한 효과적인 추천 알고리즘

(Effective Recommendation Algorithms for Higher Quality Prediction in Collaborative Filtering)

김 택 헌 † 박 석 인 ††
(Taek-Hun Kim) (Seok-In Park)

양 성 봉 †††
(Sung-Bong Yang)

요 약 본 논문에서 우리는 추천 시스템을 위한 두 개의 정제된 이웃선정 알고리즘을 제시하고, 또한 아이템의 속성정보가 어떻게 고품질의 예측을 위해 사용될 수 있는지를 보인다. 정제된 이웃선정 알고리즘은 가상 이웃과 대체 이웃을 각각 사용하여 이행적 유사도를 기반으로 한 이웃선정 방법을 적용한다. 실험 결과는 본 논문에서 제안한 알고리즘을 적용한 추천 시스템이 다른 시스템에 비해 보다 우수한 성능을 가짐을 보여준다. 이러한 제안 시스템은 예측 품질의 저하 없이 대규모 데이터셋 문제 및 초기 참여자 문제를 극복할 수 있게 한다.

키워드 : 추천시스템, 협동적필터링, 이웃선정알고리즘, 예측품질

· 본 연구는 한국과학재단(KOSEF) 일반연구자지원사업(2010-0015846) 지원으로 수행되었음
· 이 연구에 참여한 연구자의 일부는 '2단계 BK21사업' 지원비를 받았음
· 이 논문은 2010 한국컴퓨터종합학술대회에서 '협동적 필터링 기반 고도화된 추천 시스템'의 제목으로 발표된 논문을 확장한 것임

† 정 회 원 : 고려대학교 컴퓨터학과 교수
kimthun@korea.ac.kr
†† 정 회 원 : 연세대학교 컴퓨터과학과
psi93@cs.yonsei.ac.kr
††† 종신회원 : 연세대학교 컴퓨터과학과 교수
yang@cs.yonsei.ac.kr

논문접수 : 2010년 8월 10일
심사완료 : 2010년 10월 7일

정보과학회논문지: 컴퓨팅의 실제 및 레터 제16권 제11호(2010.11)

Abstract In this paper we present two refined neighbor selection algorithms for recommender systems and also show how the attributes of the items can be used for higher prediction quality. The refined neighbor selection algorithms adopt the transitivity-based neighbor selection method using virtual neighbors and alternate neighbors, respectively. The experimental results show that the recommender systems with the proposed algorithms outperform other systems and they can overcome the large scale dataset problem as well as the first rater problem without deteriorating prediction quality.

Key words : Recommender systems, Collaborative filtering, Neighbor selection algorithm, Prediction quality

1. Introduction

A recommender system using collaborative filtering which we call it CF, calculates the similarity between the test customer who is supposed to obtain a recommendation from the recommendation system and each of other customers who have rated the items that are already rated by the test customer. Since CF is based on the ratings of the neighbors who have similar preferences, it is very important to select the neighbors properly to improve prediction quality.

There have been various researches in selecting proper neighbors based on neighbor selection methods such as the k-nearest neighbor selection, the threshold-based neighbor selection, and the clustering-based neighbor selection. They are quite popular techniques for recommender systems [1,2]. These techniques then predict customer's preferences for the items based on the results of the neighbors' evaluation on the same items.

In this paper we present two refined neighbor selection algorithms for recommender systems and also show how the attributes of the items can be used for higher prediction quality. These two proposed algorithms adopt the transitivity-based neighbor selection method using virtual neighbors and alternate neighbors, respectively. We also exploit the attributes of each item for the prediction process.

The rest of this paper is organized as follows. Section 2 describes collaborative filtering based recommender systems. In Section 3, the proposed recommender algorithms to improve prediction qua-

lity in collaborative filtering are presented and in Section 4, the experimental results are illustrated. Finally, the conclusions are given in Section 5.

2. Collaborative Filtering-based Recommender Systems

A CF method suggested by the GroupLens reduces the customer's burden of searching quite successfully [3]. It filters unnecessary information out and provides selective information to the customers. CF compares the customers based on their preferences on the items to make recommendations to the customers with similar preferences. So it is widely used in the recommender systems and is also called 'social' information filtering.

A CF-based system recommends items through building the profiles of the customers based on their preferences for each item. In CF, preferences are represented generally as numeric values which are rated by the customers. Predicting the preference for a certain item that is new to the test customer is based on the ratings of other customers for the 'target' item. Therefore, it is very important to find a set of customers, called *neighbors*, with more similar preferences to the test customer for better prediction quality.

CF creates a set of customer's preferences for each item, compares it to other customers' preferences. CF filters the proper information through such comparisons. That is, CF calculates the degrees of similarity between the preference of the test customer and each of those of other customers using the correlation of preferences for the same item. CF then selects suitable items based on both their preferences and the similarities, and recommends the items to the test customer. CF works very well for the recommender systems in general.

In CF, Equation (1) is used to predict the preference of a customer [4]. In the equation, $w_{a,k}$ is the Pearson correlation coefficient which can be computed with the equation in [4].

$$P_{a,i} = \bar{r}_a + \frac{\sum_k \{w_{a,k} \times (r_{k,i} - \bar{r}_k)\}}{\sum_k |w_{a,k}|}. \quad (1)$$

Although CF can be regarded as a good choice for a recommender system, there is still much more room for improvement in prediction quality. A weak point of the CF is that it may use all other customers including "useless" customers as the neighbors of the test customer. Another drawback is that it never considers customer's preferences on the attributes of each item. More over it still cannot resolve the large scale dataset problem and the first rater problem [5,6]. Therefore, CF needs reinforcements such as utilizing "useful" attributes of the items as well as a more refined neighbor selection.

3. Improving the Prediction Quality in CF

We propose an effective recommendation algorithm for clustering-based recommender systems. It uses a refined neighbor selection algorithm that considers both high and low similarities with respect to the test customer and exploits the transitivity of similarity using a graph approach. The proposed algorithm also utilizes the attributes of the items in the process of prediction for high prediction quality.

3.1 Transitivity-based Neighbor Selection

It can be noted that if the similarity between customers a and b is high and the similarity between customers b and c is high, then the similarity between customers a and c is also considered as high. The same holds for low similarly. That is, we may achieve valuable information for prediction through the transitivity of similarities.

The key ideas of transitivity-based neighbor selection algorithm (TNSA) are to exploit the transitivity of similarities of the customers and to consider both higher and lower similarities in selecting neighbors. We regard a portion of the input data set as a complete undirected graph in which a vertex represents a customer and a weight on an edge corresponds to the similarity between two end points (customers) of the edge.

TNSA creates k clusters from the input dataset with the k-means clustering algorithm before selecting the neighbors based on the transitivity of similarities. It then finds the best cluster C with respect to the test customer t among the k clusters.

3.2 Neighbor Selection using Virtual Neighbors

TNSA can be reinforced with the virtual neighbor concept for selecting more valuable neighbors. A *virtual neighbor* we mean a customer who has the same properties as the test customer t. Let $TNSA_v$ denote TNSA with virtual neighbors.

$TNSA_v$ creates a virtual customer v in the best cluster C. It then searches the unmarked vertices adjacent to v who have the similarities either larger than δ_H or smaller than δ_L, where δ_H and δ_L are some threshold values for the Pearson correlation coefficients. Note that as the threshold values changes, so does the size of the neighbors. The search is performed in a breadth-first manner. That is, we search the adjacent vertices of v according to δ_H and δ_L to find the neighbors of t, and then search the adjacent vertices of each neighbor of v in turn. The search stops when we have enough neighbors for prediction.

The following describes $TNSA_v$ in detail. Right before the termination of the algorithm, the virtual customer v is removed from the set *Neighbors* which is returned as the output.

$TNSA_v$ Algorithm

Input: the test customer t and the input dataset S

Output: the set *Neighbors*

[1] Create k clusters from S with the k-means clustering method;

[2] Find the best cluster C for the test customer t;

[3] Create the virtual customer v who has the same properties as the test customer t and insert v into the best cluster C;

[4] Add v to the set *Neighbors*;

[5] If there are enough neighbors, remove the virtual customer v from *Neighbors*. Otherwise, traverse C from v in a breadth-first manner when visiting vertices (customers). The similarity of the customer is checked to see if it is either higher than δ_H or lower than δ_L. If so, let v = the customer and go to Step 4; // Note that v is not a virtual neighbor any more.

[6] Return *Neighbors*;

Note that in Step 5 the algorithm is terminated if the number of levels (depths) we searched from the virtual customer-added in Step 3 in a breadth-first manner-is greater than a fixed value. This value can be determined through various experiments.

3.3 Neighbor Selection using Alternate Neighbors

As in the section 3.2, TNSA can also be reinforced with the alternate neighbor concept for selecting more valuable neighbors. An alternate *neighbor* we mean a customer who has the best similarity with the test customer t. Let $TNSA_a$ denote TNSA with alternate neighbors.

$TNSA_a$ finds an alternate customer a in the best cluster C. $TNSA_a$ then searches the unmarked vertices adjacent to a who have the similarities either larger than δ_H or smaller than δ_L, as in $TNSA_v$. The search is also performed in a breadth-first manner.

The following describes $TNSA_a$ in detail. When the algorithm is terminated, the alternate customer a is not removed from the set because it is the best neighbor for the test customer t.

$TNSA_a$ Algorithm

Input: the test customer t and the input dataset S

Output: the set *Neighbors*

[1] Create k clusters from S with the k-means clustering method;

[2] Find the best cluster C for the test customer t;

[3] Find the alternate customer a who has the best similarity with the test customer t in the best cluster C;

[4] Add a to *Neighbors*;

[5] If there are enough neighbors, return *Neighbors*. Otherwise, traverse C from a in a breadth-first manner for visiting vertices (customers). The similarity of the customer is checked to see if it is either higher than δ_H or lower than δ_L. If so, let a = the customer and go to Step 4; // Note that a is not an alternate neighbor any more.

Note that the alternate customer is not removed from the set *Neighbors* because it is the best neighbor for the test customer t.

3.4 Prediction using the Attributes

For using the attributes in prediction we use Equation (2) as a new prediction formula in order to predict customer's preferences more accurately [4]. In this equation, $A(\overline{r_{a,i}})$ and $A(\overline{r_{k,i}})$ are the average attribute values of customers a and k, respectively.

$$P_{a,i} = A(\overline{r_{a,i}}) + \frac{\sum_k \{w_{a,k} \times (r_{k,i} - A(\overline{r_{k,i}}))\}}{\sum_k |w_{a,k}|}. \quad (2)$$

There are a lot of items which have different attributes with respect to the item for new prediction. Therefore, if we retrieve the attributes of the items more accurately and use them for the prediction process with Equation (2), then we can achieve more accurate prediction quality than the case without considering the attributes, because the customers are in general very sensitive to the attributes of the items.

4. The Experiments

4.1 Experimental Settings

In the experiments we used the MovieLens dataset of the GroupLens Research Group [7]. We used two types of evaluation metrics; they are *prediction accuracy* and *recommendation list accuracy*. One of the statistical prediction accuracy metrics is the mean absolute error (MAE); it is the mean of the errors of the actual customer ratings against the predicted ratings in an individual prediction [8]. Precision and recall are also used for evaluating the recommendation list in the Information Retrieval Community [9]. And the standard F-measure is used in order to evaluate the quality as a single measure which is given by Equation (3) [10].

$$F - measure = \frac{2 \cdot Presision \cdot Recall}{Presision + Recall} \quad (3)$$

For the experiment, we have chosen randomly 10% of customers out of all the customers in the dataset as the test customers. The rest of the customers are regarded as the training customers. For each test customer, we chose ten different movies randomly that are actually rated by the test customer as the test movies. The final experimental results are averaged over the results of ten different test sets for a statistical significance.

For comparing each method, we have implemented three recommendation systems for the experiments. The first one is the recommendation system only with the clustering-based CF, called *KCF*. The second one is KCF with virtual neighbors, called $TNSA_v$. The third one is KCF with alternate neighbors, called $TNSA_a$. For each system, we have implemented two different settings. One is each system without the attribute information and the other is those with the attribute information, we call those KCF^a for KCF, $TNSA^a_v$ for $TNSA_v$, and $TNSA^a_a$ for $TNSA_a$, respectively.

4.2 Experimental Results

The experimental results are shown in Fig. 1 and Fig. 2. We determined the parameters which gave us the smallest MAEs and the largest precision, recall and F-measure through various experiments. As shown in these figures, each system has been tested both with and without utilizing the attributes. The results in those figures show that the systems using the attributes outperform those without considering them. For the prediction accuracy and the recommendation list accuracy, the improvement ratio of each system to KCF is shown in Table 1.

The experimental results show us that $TNSA^a_v$ outperforms other systems for both the prediction accuracy and the recommendation list accuracy. We also found that the prediction quality of $TNSA^a_a$ is better than other systems except $TNSA^a_v$. And KCF^a shows also as good prediction quality as KCF.

In the table the prediction accuracy improvement ratio of $TNSA^a_v$ to KCF is more than 10%. And the recommendation list accuracy improvement ratio of $TNSA^a_v$ to KCF is more than 5% for each metric. In $TNSA^a_a$ the improvement ratio is more than 9% for MAE and is 4% and over for precision, recall, and F-measure.

In the results we found that both $TNSA^a_v$ and $TNSA^a_a$ improve the prediction quality significantly for all the cases. On the other hand the prediction qualities of both $TNSA_v$ and $TNSA_a$ improved not much. But $TNSA_v$ showed as good prediction quality as KCF for the prediction accuracy metrics.

From the experimental results, we can conclude that it is better for us to use either virtual neighbors or alternate neighbors in the recommender systems using the attribute information. Therefore, the clustering-based recommender system with one of the refined neighbor selection methods along with utilizing the attribute information can solve the very large scale dataset problem without dete-

riorating prediction quality. In addition, a recommender system using an alternate neighbor can be a choice for the first-rater problem in collaborative filtering.

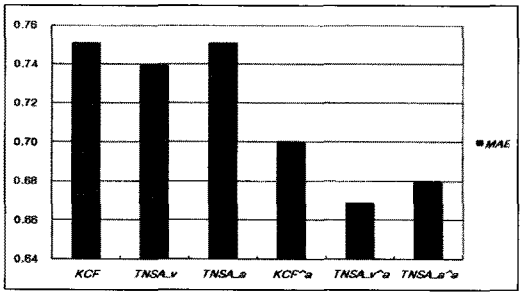

Fig. 1 Experimental results for prediction accuracy metrics

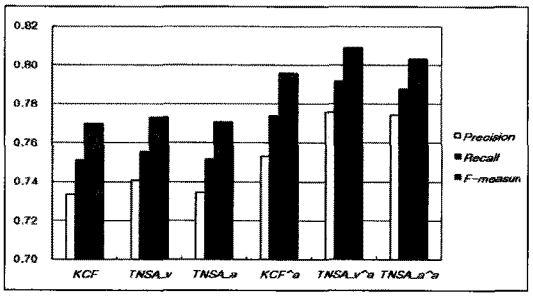

Fig. 2 Experimental results for recommendation list accuracy metrics

Table 1 The improvement ratio(%) to KCF

	MAE	Precision	Recall	F-measure
TNSAv	1.55	0.98	0.43	0.56
TNSAa	0.03	0.13	0.09	0.09
KCFa	6.77	2.65	3.35	3.03
TNSAav	11.00	5.77	5.08	5.41
TNSAaa	9.56	5.55	4.33	4.90

5. Conclusions

In this paper we proposed a transitivity-based neighbor selection method that can use either virtual neighbors or alternate neighbors in searching for the useful neighbors. We also showed that utilizing the attributes of the items in collaborative filtering improves prediction quality. The experimental results showed that the system with the proposed methods provides the better prediction quality than others. From the experimental results, we could observe the followings:

- The neighbor selection based on the transitivity of similarity can select meaningful customers as the neighbors.
- The neighbor selection using either a virtual neighbor or an alternate neighbor is useful to select valuable neighbors in collaborative filtering.
- The recommender systems using the attribute information have improved the prediction quality, compared with the ones without using it.
- The clustering-based collaborative filtering using the proposed methods can be an answer to the large scale dataset problem and also to the first rater problem without deteriorating prediction quality.

References

[1] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl, "Analysis of Recommendation Algorithms for E-Commerce," *Proc. of ACM E-Commerce 2000*, 2000.

[2] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl, "Recommender Systems for Large-Scale E-Commerce: Scalable Neighborhood Formation Using Clustering," *Proc. of the Fifth International Conference on Computer and Information Technology*, 2002.

[3] J. A. Konstan, B. Miller, D. Maltz, J. L. Herlocker, L. Gordon, and J. T. Riedl, "GroupLens: Applying Collaborative Filtering to Usenet News," Communications of the ACM, vol.40, 1997.

[4] T. H. Kim and S. B. Yang, "Using Attributes to Improve Prediction Quality in Collaborative Filtering," *Proc. of EC-Web 2004*, 2004.

[5] J. Riedl, T. Beaupre, J. Sanders, "Research Challenges in Recommenders," *Proc. of ACM Recommender Systems 2009 Tutorial*, 2009.

[6] X. N. Lam, T. Vu, T.D. Le, and A.D.Duong, "Addressing Cold-start Problem in Recommendation Systems," *Proc. of ICUIMC'08*, 2008.

[7] MovieLens dataset, GroupLens Research Group, url: http://www.grouplens.org/.

[8] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," *Proc. of UAI 1998 Conference*, 1998.

[9] H. Nguyen and P. Haddawy, "The Decision-Theoretic Interactive Video Advisor," *Proc. of UAI 1999 Conference*, 1999.

[10] R. J. Mooney and L. Roy, "Content-Based Book Recommending Using Learning for Text Categorization," *Proc. of ACM Conference on Digital Libraries*, 2000.