Discovery of Hidden Similarity on Collaborative Filtering to Overcome Sparsity Problem

Sanghack Lee, Jihoon Yang, and Sung-Yong Park

Department of Computer Science, Sogang University
1 Shinsoo-Dong, Mapo-Ku, Seoul 121-742, Korea
shlee@mllab.sogang.ac.kr, {jhyang,parksy}@ccs.sogang.ac.kr

Abstract. This paper presents a method for overcoming sparsity problem of collaborative filtering system. The proposed method is based on an intuition on a network of human (such as a friendship network with friends of a friend). This method increases the density of similarity matrix and the coverage of predictions. We use sparse training data to test the sparsity of real-world situation. Consequently, experimental results show that this method increases coverage and f-measure especially for sparse training data.

1 Introduction

To get high quality information in some domains, we have to experience the information directly or indirectly. But we can't experience all the information directly. Word-of-mouth information are often helpful when we choose a decision without direct experience.

A recommender system discovers and renders items (e.g. movies, books, etc.) which suits our preferences. Many recommender systems have been developed with collaborative filtering techniques. The terms collaborative filtering and recommender systems are used interchangeably. Content based filtering is one of information filtering techniques which uses context of items. On the other hand, collaborative filtering doesn't make use of context of items which makes it possible to recommend serendipitous items [1, 2]. Collaborative filtering is based on opinions of similar users who buys or rates same items.

However, there are two main limits in collaborative filtering: *scalability* and *sparsity* problems.

- *Scalability*: In e-commerce environment, there might be millions of customers and catalog items. Traditional collaborative filtering algorithms thus require expensive time and space complexity [3].
- *Sparsity*: With millions of items, we can't experience even 1% of them. For instance, Amazon.com has several millions of catalog items and 1% of them are more than 10,000 items which people can hardly access [3].

We will focus on the sparsity problem and suggest an approach to overcoming it. Previous works for overcoming the sparsity problem include combining content

based filtering and collaborative filtering to link contextual information among items [4, 5] and clustering items or users to reduce dimensionality [6]. (You can find more detailed descriptions in Section 4.) In this paper, we will explain how to increase coverage based on a network of similar users and the transitivity of similarity.

2 Method

A main idea of this paper is connecting or relating two users who don't have any co-rated items. We keep meeting people as friends of a friend in everyday life. Friends of a friend construct a network of human with friendship. Changing a friendship network into a similarity network, users that are connected to similar set of neighbors (middlemen) construct a network of users with similarity. Using this scheme any two users with common middlemen can be connected in the network.

As shown in Fig. 1, discovering hidden similarity (hereinafter referred to DHS) computes a similarity matrix among users based on their rates on items, and modifies the matrix incorporating the discovered hidden similarity.

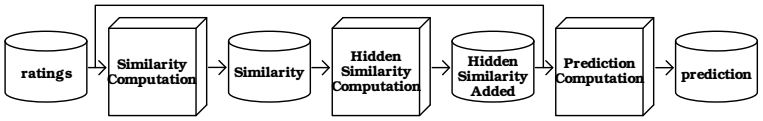

Fig. 1. A simple architecture of DHS based collaborative filtering system

2.1 Similarity Computation

The Pearson correlation and the cosine based similarity computation is mostly used in traditional collaborative filtering recommendation systems [7]. In this paper, we will use these two basic similarity computation methods:

- *Pearson Correlation*

$$w_{a,i} = \frac{\sum_{j=1}^{N} (v_{a,j} - \bar{v}_a)(v_{i,j} - \bar{v}_i)}{\sqrt{\sum_{j=1}^{N} (v_{a,j} - \bar{v}_a)^2} \sqrt{\sum_{j=1}^{N} (v_{i,j} - \bar{v}_i)^2}} \tag{1}$$

- *Cosine Based*

$$w_{a,i} = \frac{\sum_{j=1}^{N} v_{a,j} v_{i,j}}{\sqrt{\sum_{j=1}^{N} {v_{a,j}}^2} \sqrt{\sum_{j=1}^{N} {v_{i,j}}^2}} \tag{2}$$

where, $w_{a,i}$ is a similarity between two users a and i, N is the number of users in the system, $v_{i,j}$ is rating for item j by user i, and $\bar{v}_i$ is the average rating of user i. (See [7] for detailed descriptions.)

2.2 Hidden Similarity Computation

The discovery hidden similarity (DHS) is computed as follows:

$$w_{a,i} = \frac{\sum_k |w_{a,k}| \frac{\sum_k w_{a,k} w_{k,i}}{\sum_k |w_{a,k}|} + \sum_k |w_{k,i}| \frac{\sum_k w_{a,k} w_{k,i}}{\sum_k |w_{k,i}|}}{\sum_k |w_{a,k}| + \sum_k |w_{k,i}|}$$

$$= \frac{2 \sum_k w_{a,k} w_{k,i}}{\sum_k |w_{a,k}| + \sum_k |w_{k,i}|} \tag{3}$$

where, a and i are users without co-rated items, k is a middleman who have co-rated items for both a and i, and $w_{a,i}$ is the weighted sum of two similarity predictions. The numerator consists of two terms: one is the weighted sum of $w_{i,k}$ and the other is the weighted sum of $w_{a,k}$. The similarity predictions among users in DHS are similar to the prediction computation in Section 2.3.

2.3 Prediction Computation

– *Prediction for Pearson Correlation*

$$p_{a,j} = \bar{v}_a + \frac{\sum_{i=1}^{N} w_{a,i}(v_{i,j} - \bar{v}_i)}{\sum_{i=1}^{N} |w_{a,i}|} \tag{4}$$

– *Prediction for Cosine Based*

$$p_{a,j} = \frac{\sum_{i=1}^{N} w_{a,i} v_{i,j}}{\sum_{i=1}^{N} w_{a,i}} \tag{5}$$

where $p_{a,j}$ is the prediction for an item j of a user a. Predicted items are recommended when predicted ratings of the items are over a threshold. In our experiments, the recommendation threshold is set to 3.5 because a range of the ratings is from 1 to 5.

The hidden similarity computation in Section 2.2 has the same purpose as that for the similarity computation. Both computations yield a similarity matrix among users. By computing the hidden similarity, the similarity matrix becomes less sparse and the system overcomes the sparsity of the user similarities and the predicted ratings. Note that the equation for the hidden similarity computation (3) is similar to the equations for the prediction computation ((4) and (5)). Prediction computation predicts a rating as the weighted sum of ratings based on the similarity, and the hidden similarity computation predicts a hidden similarity as the weighted sum of similarities based on similarity. The difference between two computations is that the former is directional while the latter is not. Fig. 2 depicts these three computations.

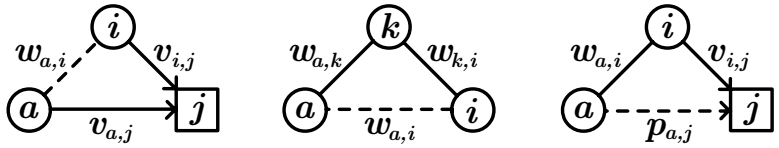

Fig. 2. Comparison among (a) similarity computation, (b) hidden similarity computation, and (c) prediction computation. Circles and rectangulars are users and items, respectively. Solid lines and dotted lines are known and unknown values, respectively.

3 Experiments

3.1 Dataset

In our experiments, we used MovieLens[1] data which contains 100,000 ratings from 943 users and 1682 movies. The sparsity of the data is 93.7%, and the number of average rated movies per user is about 106.

We then randomly divided the ratings into two parts, training and test data. We performed experiments from 2-fold to 10-fold cross validations, and additional cross validations interchanging the training and test data in order to evaluate our approach in a real-world situation where training data is very sparse.

Several different sizes of middlemen (k) in Equation 3 was tested with 5, 10, and without any restrictions.

3.2 Evaluation Metrics

There are several metrics to evaluate a recommender system. The recommendation quality can be measured by *precision* and *mean absolute error* (MAE). *Precision* is the number of correctly recommended movies over the number of all recommended movies. *MAE* is the average difference between the real ratings and predictions. *Recall* and *coverage* are used for evaluating quantity of recommendation. *Recall* is the number of correctly recommended movies over the number of all liked movies. *Coverage* is the number of predicted movies over the number of all movies in the test data. In addition, *f-measure* is the weighted combination of *precision* and *recall* as $f_1 = \frac{2precision \times recall}{precision + recall}$ which evaluates quality and quantity of the recommendation. We will use *coverage* and *f-measure* to evaluate both quality and quantity of our recommender system.

3.3 Results

Table 1 and Fig. 3 show the effect of DHS on coverage and f-measure. Experiments with a restriction on the size of middlemen makes a slight difference in

[1] http://movielens.umn.edu

Table 1. Performance of DHS-based algorithms on two methods of similarity computation

method		training percentage					
		10%		50%		90%	
		coverage	f-measure	coverage	f-measure	coverage	f-measure
Pearson correlation based	pure	68.20%	63.00%	96.79%	70.70%	98.16%	72.87%
	DHS	**93.38%**	63.76%	**99.65%**	**72.05%**	**99.86%**	**73.93%**
	10-DHS	**93.38%**	63.78%	**99.65%**	72.03%	**99.86%**	**73.93%**
	5-DHS	**93.38%**	**63.83%**	**99.65%**	72.03%	**99.86%**	**73.93%**
cosine based	pure	70.55%	65.88%	96.79%	69.22%	98.16%	69.65%
	DHS	**96.66%**	**68.62%**	**99.65%**	**70.21%**	**99.86%**	**70.50%**
	10-DHS	**96.66%**	**68.62%**	**99.65%**	70.20%	**99.86%**	**70.50%**
	5-DHS	**96.66%**	68.60%	**99.65%**	70.20%	**99.86%**	70.49%

f-measure. Without restriction, the average size was 9.42, 211.58, and 286.57 for 10%, 50%, and 90%, respectively.

As we can see, the coverage increased in DHS-based methods. In particular, the coverage increased significantly when the training data is sparse. The coverage strongly depends on the density of similarity matrix (i.e. Hidden Similarity Added in Fig. 1) because predictions per user is a union of set of movies rated by neighbors.

Surprisingly, f-measure was increased slightly against our expectation. We found that the predictions via the hidden similarity added would not be as good as via pure similarity at first. Because the hidden similarity is predicted based on the pure similarity and then the quality of the hidden may be worse than the pure. But actually, the reliability of user similarity by the Pearson correlation or the cosine based is determined by the number of co-rated items [8]. In that manner, the initially derived similarity (i.e. pure similarity) has low reliability because of sparse ratings, and adding combined similarities (i.e.

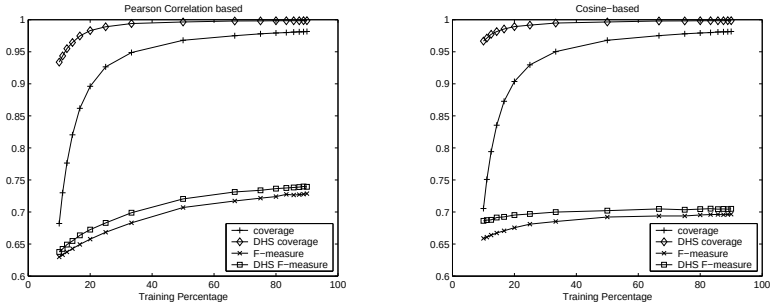

Fig. 3. Performance graphs with training percentage from 10 to 90

hidden similarity added) yields higher reliability. Also the increase of the number of available neighbors for the prediction causes better results [1]. Consequently, the DHS may have higher reliability and quality than the similarity based on sparse ratings.

4 Related Work

Collaborative filtering has been studied and varied in many ways. Its diverse methods can be categorized by several criteria [9]. One of the criteria is hybrid algorithms versus plain algorithms.

Plain algorithms are based on the user-item matrix, the user correlation, and their revised information. For example, item based collaborative filtering which alternates the role of users and items [10], LSI/SVD as a dimensionality reduction technique [11], and clustering algorithms for large dimensional sparse vectors without contextual information [12] have been developed.

Hybrid algorithms often use demographic or content data to achieve high quality recommendation. For instance, collaborative filtering combined with content based methods [5, 13] , a content-boosted method to create a pseudo user-ratings vector for overcoming sparsity [4], collaborative filtering via content and demographic data based approach [14], and clustering items to improve the quality [6] have been designed.

Our method is a plain algorithms based on the DHS that renders denser user similarity.

5 Summary and Future Work

In this paper, we have presented the DHS algorithm to overcome the sparsity problem by connecting more users. We have found that similarities through not only movie ratings but also user similarities work well. Major contribution is that hidden similarity computation have showed how to capture a transitive similarity relationship with simple and neat equations [15]. Even though our proposed method is based on the intuition of the human network, it increases coverage of traditional collaborative filtering systems without losing precision and recall. Future works will include extension of DHS:

- *System integration:* We suggest integrating systems with number of co-users in the same domain. Systems with content based filtering and with collaborative filtering also can be integrated by combining their similarity matrices into a single similarity matrix.
- *Repeated DHS:* DHS discovers the similarities between two users when the length of a shortest path between two users is 2. This fact implies two users of d-length shortest path can compute the similarity by repeating DHS at most $\lceil \log_2 d \rceil$ times because the length of shortest path is halved in every execution of DHS. In this manner, we can discover similarities between any connected users.

References

1. Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J.: An algorithmic framework for performing collaborative filtering. In: SIGIR '99: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, CA, ACM Press (1999) 230–237
2. Polcicova, G., Slovak, R., Navrat, P.: Combining content-based and collaborative filtering. In: ADBIS-DASFAA Symposium. (2000) 118–127
3. Linden, G., Smith, B., York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. IEEE Internet Computing **7** (2003) 76–80
4. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: Proceedings of the Eighteenth National Conference on Artificial Intelligence, Edmonton, Canada (2002) 187–192
5. Maneeroj, S., Kanai, H., Hakozaki, K.: Combining dynamic agents and collaborative filtering without sparsity rating problem for better recommendation quality. In: Proceedings of the Second DELOS Network of Excellence Workshop on Personalisation and Recommender Systems in Digital Libraries. (2001)
6. O'Connor, M., Herlocker, J.: Clustering items for collaborative filtering. In: Recommender Systems Workshop at 1999 Conference on Research and Development in Information Retrieval, Berkeley, CA (1999)
7. Heckerman, D., Kadie, C., Breese, J.S.: Empirical analysis of predictive algorithms for collaborative filtering. In: Uncertainty in Artificial Intelligence. Proceedings of the Fourteenth Conference. (1998) 43–52
8. Ariyoshi, Y.: Improvement of combination information filtering method based on reliabilities. IPSJ SIGNotes Fundamental Infology (1998)
9. Vozalis, E., Margaritis, K.G.: Analysis of recommender systems' algorithms. In: Proceedings of the Sixth Hellenic-European Conference on Computer Mathematics and its Applications - HERCMA 2003. (2003)
10. Sarwar, B.M., Karypis, G., Konstan, J.A., Riedl, J.: Item-based collaborative filtering recommendation algorithms. In: World Wide Web. (2001) 285–295
11. Sarwar, B., Karypis, G., Konstan, J., Riedl, J.: Application of dimensionality reduction in recommender systems–a case study. In: ACM WebKDD 2000 Web Mining for E-Commerce Workshop. (2000)
12. Kohrs, A., Merialdo, B.: Clustering for collaborative filtering applications (1999)
13. Good, N., Ben Schafer, J., Konstan, J.A., Borchers, A., Sarwar, B., Herlocker, J., Riedl, J.: Combining collaborative filtering with personal agents for better recommendations. In: Proceedings of the 6th National Conference on Artificial Intelligence (AAAI-99); Proceedings of the 11th Conference on Innovative Applications of Artificial Intelligence. (1999) 439–446
14. Pazzani, M.J.: A framework for collaborative, content-based and demographic filtering. Artificial Intelligence Review **13** (1999) 393–408
15. Billsus, D., Pazzani, M.J.: Learning collaborative information filters. In: Proceedings of the Fifteenth International Conference on Machine Learning (ICML-98), Madison, WI, Morgan Kaufmann (1998) 46–54