A SOMO-based Recommendation System

Mu-Chun Su Chao Yu Chen Yu-Xiang Zhao

Department of Computer Science & Information Engineering, National Central University, Taiwan, R.O.C.
E-mail address muchun@csie.ncu.edu.tw

Abstract

In a large-scale distributed network environment like the Internet, information has been increased and changed continuously. Accessing information in such dynamically changing, heterogeneous and world-wide distributed environments puts a big burden on users. A possible solution to alleviate information overload by identifying which information a user will find worthwhile is the use of recommendation systems. Recommendation systems are a kind of web intelligence techniques to make daily information filtering for people. In this paper, we propose a SOMO-based recommendation system which is able to learn personal preferences of users and provide tailored suggestions to users. Experiments on movie recommendation were conducted to test the performance of the proposed recommendation system.

Keyword: recommendation system, *optimization algorithm, self-organizing feature map, computational intelligence.*

1. INTRODUCTION

In complex and open environments, such as the Internet, recommendation systems will play an important role in helping individuals and communities address the challenges of information overload. Recommendation systems can be regarded as a kind of web intelligence techniques to make daily information filtering for people. Conventionally, there are two approaches to implementing recommendation systems: content-based (or feature-based) filtering and collaborative (also called as social or community) filtering [1]-[14].

At the early developing stage of recommendation systems, many content-based recommendation systems were simply a query-based information retrieval system. For example, search engines recommend web pages with content similar to users' queries [1]. Content-based filtering is based on content analysis of the considered items (or objects). For content-based filtering, its performance greatly depends on whether the content analysis and users' preferences can be reliably and automatically determined. Content-based filtering recommends items for a user's consumption based on correlations between the contents of the items and the user's preferences. However it has limitations: (1) it cannot filter items based on style, quality, or point-of-view, (2) it requires a source of contents, and (3) it cannot provide much in the way of serendipitous discovery.

A few years later, Goldberg et al firstly applied the collaborative filtering technology to recommendation systems [2], [3]. Collaborative filtering does not show the limitations encountered by content-based filtering. For collaborative filtering, items are selected for an active user (In this paper, the current user is referred to as the active user) when they are also relevant to users who have similar preferences. Ratings of items from many similar users are collected and recommendations are then made based on those ratings to the active user. In general, the contents of the items are ignored for the conventional collaborative filtering approach. The techniques of collaborative filtering have been attracting to not only the research area but also the commercial field.

Although collaborative filtering has been very successful in both research and practice, its performance can be further improved if the following three problems can be solved. The first problem is the selection of the user group consisting of users who have similar preferences to the active user. The success of collaborative filtering greatly depends on the correctness of the selection of the user groups. The second problem is the first-rater problem. For collaborative filtering to work, items must be evaluated by at least one user. New items cannot be recommended until some users have taken the time to evaluate. Currently, a possible solution to the problem relies on the altruism of a set of users who are willing to rate many items without receiving many recommendations. The third problem is the profile matching. Effective profile matching can improve the correct prediction rates of a recommendation system.

The recommendation system based on the particle swarm optimization provided a possible solution to the third problem [13]. In this paper, we propose a SOMO-based recommendation system. The proposed recommendation system can provide appealing solutions to the aforementioned three problems. First, the recommendation system adopts a two-phase clustering procedure to select similar users. Then it employs the som-based optimization (SOMO) algorithm to fine-tune the parameters involved in the computation of the profile matching to improve the correct prediction rates so as to provide tailored suggestions to users. The SOMO algorithm is one of our previous works [15]. It is a new approach to optimization problems based on the self-organizing feature maps. Through the self-organizing process, good solutions to an optimization problem can be simultaneously explored and exploited.

The paper is organized as follows. A brief review of the

SOMO algorithm is given in Section 2. The details of the proposed recommendation system will be discussed in Section 3. Then in Section 4 we will present the simulation results. Finally Section 5 concludes this paper.

2. BRIEF REVIEW OF THE SOMO ALGORITHM

In [15], we tried to explore the possibility of applying the SOM algorithm in continuous optimization problems. We named the new optimization algorithm the SOM-based optimization (SOMO) algorithm. Through the self-organizing process, good solutions to an optimization problem can be simultaneously explored and exploited. The outputs of the trained neural network allow us to transform a multi-dimensional fitness landscape into a three-dimensional projected fitness landscape.

To apply the SOM algorithm to solve optimization problems we need to make several modifications. First, there is only one input pattern, $\underline{x} = (1,\cdots,1)^T$ which will be continuously presented to the network to be trained. The dimensionality of the input pattern $\underline{x}$ is equal to the one of the parameters of the optimization problem. Second, the activation function for each neuron in the network is changed to be the fitness function for the optimization problem to be solved instead of a sigmoid function or a Gaussian function. Let the function, $f(\cdot)$, represent the fitness function of the optimization problem to be solved. It contains the degree of success with which patterns of parameters optimize the values of the optimization problem. If the goal of the optimization problem is to minimize the objective function then we can take the negative of the objective function as the fitness function. Otherwise, we may directly use the objective function of the optimization problem to be the fitness function.

Each weight vector, $\underline{w}_j$, represents a possible solution to the optimization problem. While we generate $M \times N$ weight vectors a population of $M \times N$ solutions is used to explore the optimization space for searching a relatively good region in the optimization space. Then the exploitation for locating the very best solution is achieved via updating the weight vectors to be more like the winning neuron's weight vector. Therefore, exploration and exploitation can be simultaneously taken in searching for optima through the optimization space via the self-organizing process. Four essential processes involved in the proposed SOMO algorithm are as follows:

Step 1. Initialization: We may choose one of the two efficient weight initialization schemes proposed in [16]-[17] for initializing the weight vectors to accelerate the optimization process. After the initialization procedure, a small amount of random noise is added to each weight vector. Of course, we still may choose random values for the initial weights, $\underline{w}_j$'s.

Step 2. Winner Finding: Present an input pattern $\underline{x} = (1,\cdots,1)^T$ to the network and find the winning neuron j^* with the largest fitness using the maximum-fitness criterion:

$$j^* = \underset{1 \le j \le M \times N}{Arg\max} \; f(w_{j1} \times x_1, \cdots, w_{jn} \times x_n) = \underset{1 \le j \le M \times N}{Arg\max} \; f(\underline{w}_j) \quad (1)$$

Step 3. Weights Updating: Adjust the weight vectors of the winner and its neighbors, using the following rule:

$$\underline{w}_j(k+1) = \underline{w}_j(k) + \Lambda_{j^*,j}[\underline{w}_{j^*}(k) - \underline{w}_j(k)] \quad for\ 1 \le j \le M \times N \quad (2)$$

$$\Lambda_{j^*,j} = \lambda\left(1 - \frac{d_{j^*,j}}{\sqrt{M^2 + N^2}}\right) \quad (3)$$

where the parameter λ is a real-valued constant pre-defined by the user.

Step 4. Iterating: Go to step 2 until either a pre-specified number of iterations is achieved or some kind of termination criteria is satisfied.

One thing should be emphasized is that (2) is a little different from the updating formula used in the original SOM algorithm. In (2) the weight vector, $\underline{w}_j$, is adjusted to be more like the winning neuron's weight vector, $\underline{w}_{j^*}$, instead of the input pattern, $\underline{x}$, as in the original SOM algorithm. In addition, a small amount of random noise is added to the updated weight vector. The higher the fitness the smaller amount of random noise is added.

3. THE SOMO-BASED RECOMMENDATION SYSTEM

Ideally, we should select the best match profile from the entire database of user profiles to give tailored suggestions to the active user; however, this becomes infeasible when the database is large. To alleviate the computational load of searching the whole database, we may partition the users into several user groups and then search the best match profile just from the user group consisting of users with the most similar tastes to the active user. Therefore, the success of collaborative filtering is highly dependent on whether we can effectively find the user group consisting of users whose tastes are most similar to that of the active user. Traditional collaborative filtering either randomly select a fixed number of users from the database or cluster the users into several user groups based on whole user profiles consisting of the item rating values and demographic information (such as age, gender, occupation, etc). To simultaneously solve the first two aforementioned problems we proposed to partition the items into N_i item groups and then partition users into N_u user groups under each item group. Via this two-level clustering procedure, we can effectively select the most match user group and recommend new items. To improve correct prediction rates we employ the SOMO algorithm to

fine-tune the parameters involved in the computation of the profile matching.

Figure 1 shows the flowchart of the SOMO-based recommendation system. The procedure is described as follows:

Step 1: Apply a clustering algorithm to partition the items into N_i item groups based on their simple attributes (e.g. the genre information of movies, categories of the products, etc). The number of item groups N_i is specified in advance. To reduce the computation burden in order to accelerate the response time, the K-means clustering algorithm, which is a simple and fast clustering method, is a good choice.

Step 2: For each item group, partition the users into N_u user groups based on the average ratings on the items belonging to the specific item group. The number of user groups N_u is identical to the number of levels of the ratings. Therefore, we will know which user group most like or dislike the corresponding item group. Using this two-phase clustering procedure, users belonging to the same user group for each item group will be more similar to each other than the convention one-phase clustering approach.

Step 3: After the two-phase clustering procedure, we can then use the collaborative filtering technique to make predictions of items for an active user. For an active user, the prediction for an item is computed as follows:

$$P_{A,j} = \overline{R_A} + \frac{\sum_{i \in A\text{'s neighbor}} sim(A,i) * (R_{i,j} - \overline{R_i})}{\sum_{i \in A\text{'s neighbor}} sim(A,i)} \qquad (4)$$

where $P_{A,j}$ represents the predicted rating for the active user on item j, $\overline{R_A}$ is the mean rating for user A, $sim(A,i)$ represents the degree of similarity between user A and user i *who is a member of the most match user group*, $R_{i,j}$ denotes the rating that user i has given on item j. The value of $sim(A,i)$ will affect the correct prediction rate of the system; therefore, it is better to optimize it by some optimization algorithms. In our system the value of $sim(A,i)$ is fine-tuned by the SOMO algorithm such that the mean square error of the predicted ratings and the truly ratings is minimized.

The early-rater problem can be easily solved by identify which item group new items belong to and then recommend these new items to the user group which have given the highest ratings for the item group.

4. EXPERIMENTAL RESULTS

We used the dataset collected from the MovieLens (http://www.movielens.umd.edu), a well-known web-based movie recommendation system, to test the performance of the system. The dataset contained ratings from 943 users and 1682 movies. While the movie information in the dataset included genres, theaters, video release dates, etc, the user information contained demographic information such as age, gender and occupation. We divide the dataset into a training set consisting of one third of the data set and a testing set consisting of the remaining data.

Currently, we only used the genres information to cluster the movies in the training set into 9 item groups such as action, drama, comedy, fantasy, horror, western, animation, documentary, and war. Then for each type of movies, users are further partitioned into five user groups based on their average ratings for this particular type of movies, as shown in Fig. 2.

For the comparison purpose, we conducted two experiments conducted in [13] to evaluate the SOMO-based recommendation system.

Experiment 1: Each of the first 10 users was picked as the active user in turn, and 10 out of 943 users were picked randomly and used to provide recommendations.

Experiment 2: Each of the first 50 users was picked as the active user in turn, and 50 out of 943 users were picked randomly and used to provide recommendations.

The performance comparison is based on two measures: 1) ***Zero tolerance***- the accuracy of the system is found by calculating the percentage of the number of ratings that the system predicted correctly out of the total number of available ratings by the current active user and 2) ***At-Most-One tolerance***- same as zero tolerance but if the difference between the predicted and actual ratings **is** less than or equal to one then this predicted rating is considered to be correct. The experimental results are shown in Fig. 3. By examining the results, several observations can be found. The performance of the SOMO-based recommendation system was the highest one among the four recommendation systems. In addition, our SOMO-based recommendation system did not utilize the demographic information (such as age, gender, occupation, etc) like other three systems did. This is an advantage of the proposed system since most of users may hesitate to give correct personal information on web sites. The speed of execution of our SOMO-based recommendation system is faster than the PSO-based recommendation system, and the GA-based recommendation system.

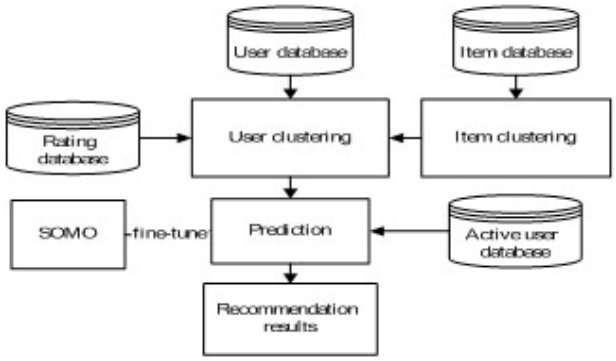

Fig. 1. The flowchart of the SOMO-based recommendation system.

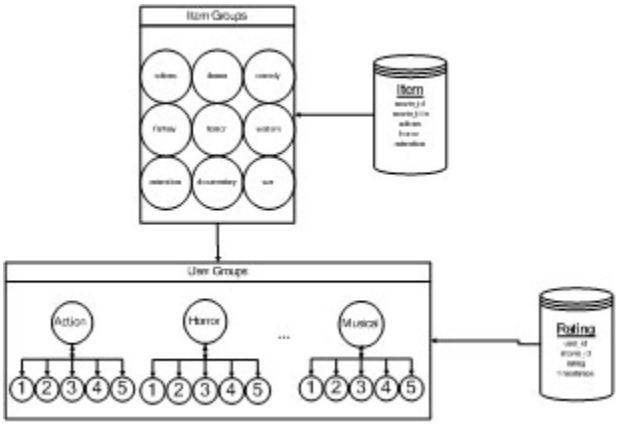

Fig. 2. The item groups and the user groups.

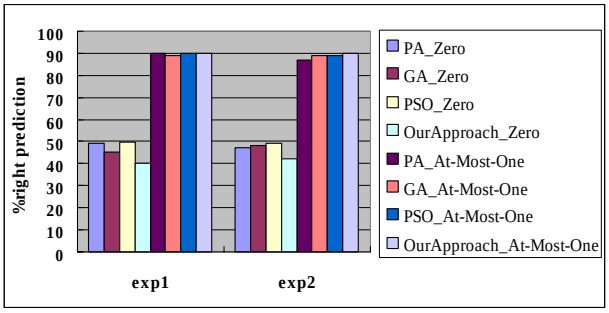

Fig. 3. Experimental results.

5. CONCLUSIONS

In this paper, we propose a SOMO-based recommendation system that is able to learn personal preferences of users and provide tailored suggestions to users. Moreover, the proposed recommendation system can provide appealing solutions to the following three problems: (1) the selection of user group, (2) the early-rater problem, and (3) the profile matching problem. The performance was verified through two experiments.

Acknowledgement This work is supported by the MOE Program for Promoting Academic Excellent of Universities under the grant number EX-91-E-FA06-4-4, the National Science Council, Taiwan, R.O.C, under the NSC 93-2524-S-008-002, and the Ministry of Economic Affairs under the 93-EC-17-A-02-S1-029.

6. REFERENCES

[1] G. Salton and McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.

[2] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, January 1992.

[3] D. B. Terry. A tour through tapestry. *In Proc. ACM Conf. On Organizational Computing Systems (COOCS)*, pages 21–30, 1993.

[4] P. Resnick, N. Iacovou, M. Suchak, P. Bergstorm, and J. Riedl. Grouplens: An open architecture for collaborative filtering of netnews. *In Proc. ACM Conf. on Computer-Supported Cooperative Work*, 1994.

[5] U. Shardanand and Maes. Social information filtering: Algorithms for automating.

[6] D. Gupta, M. Digiovanni, H. Narita, and K. Goldberg. Jester 2.0: A new linear-time collaborative filtering algorithm applied to jokes. *In Proc. ACM-SIGIR Workshop on Recommender Systems: Algorithms and Evaluation*, 1999.

[7] D. B. Hauver. Flycasting: Using collaborative filtering to generate a play list for online radio. *In Int. Conf. on Web Delivery of Music*, 2001.

[8] K. Wittenburg, D. Das, W. Hill, and L. Stead. Group asynchronous browsing on the world wide web. *In Proc. Of Fourth International World Wide Web Conference*, pages 51–62, 1995.

[9] J. B. Schafer, J. Konstan, and J. Riedl. Recommender systems in e-commerce. *In Proc. 1st ACM. Conf. on Electronic Commerce (EC'99)*, 1999.

[10] B. M.Sarwar, J. A. Konstan, A. Borchers, J. Herlocker, B. Miller and J. Riedl, "Using Filtering Agents to Improve Prediction Quality in the GroupLens Research Collaborative Filtering System," *Computer Supported Cooperative Work*, 1998.

[11] J. A. Alspector, A. Kolcz, and N. Karunanithi, "Comparing feature-based and clique-based user models for movie selection," *Proc. of the Third ACM Conference on Digital Libraries*, 1998.

[12] N. Good, J. Ben Schafer, J. A. Konstan, A. Borchers, B. Sarwar, J. Herlocker, and J. Riedl, "Combining collaborative filtering with personal agents for better recommendations," *AAAI/IAAI*, pp. 439-446, 1999.

[13] S. Ujjin and P. J. Bentley, "Particle swarm optimization recommender system," *Proc. of the IEEE Swarm Intelligence Symposium*, pp. 124-131, 2003

[14] Q. Li and B. M. Kim, "Clustering approach for hybrid recommender system", *Proc. of the IEEE/WIC International Conference on Web Intelligence*, pp. 33-38, 2003.

[15] M. C. Su, Y. X. Zhao, and J. Lee, 25-29 July 2004, "SOM-based Optimization," *IEEE International Joint Conference on Neural Networks IJCNN*, pp. 781-786, 2004.

[16] M. C. Su and H. C. Chang, "Fast self-organizing feature map algorithm," *IEEE Trans. on Neural Networks*, vol. 13, no. 3, pp. 721-733, May 2000.

[17] M. C. Su, T. K. Liu, and H. T. Chang, "An efficient initialization scheme for the self-organizing feature map algorithm," in *IEEE Int. Joint Conference on Neural Networks*, pp. 1906-1910, Washington, D.C, 1999.