A Recommendation System Based on a Subset of Raters

Bernhard Scholz
School of Information Technologies
The University of Sydney, NSW 2006, Australia
scholz@it.usyd.edu.au

Sang-Min Choi
Department of Computer Science
Yonsei University, Seoul, Republic of Korea
jerassi@cs.yonsei.ac.kr

Sang-Ki Ko
Department of Computer Science
Yonsei University, Seoul, Republic of Korea
narame7@cs.yonsei.ac.kr

Hae-Sung Eom
Department of Computer Science
Yonsei University, Seoul, Republic of Korea
haesung@cs.yonsei.ac.kr

Yo-Sub Han*
Department of Computer Science
Yonsei University, Seoul, Republic of Korea
emmous@cs.yonsei.ac.kr

ABSTRACT

Since the late 20^{th} century, the number of Internet users has noticeably increased. Recently, the number of Internet queries and the quantity of information available on the web has increased drastically. A large amount of new information is uploaded to the Web on a daily basis. However, search results are not always reliable due to the vast amount of data available on-line. As a result, users often have to repeat their searches in order to find exactly what they are looking for. To remedy this, some researchers have suggested recommendation systems. Since a recommendation system proposes information relevant to a particular query, users no longer need to repeat a search to obtain desired data. In the Web 2.0 era, recommendation systems often rely on the collaborative filtering approach, which is based on user information such as age, location, or preference. However, the traditional approach is affected by the cold-start and sparsity problems. The reason for these problems is the fact that the traditional system requires user information to operate properly. In this paper we address the sparsity problem associated with the current recommendation systems. We also suggest a new recommendation system approach and compare the performance of the proposed method with that of the traditional approach.

Categories and Subject Descriptors

H.4 [**Information Recommendation**]: Information Recommendation

*Corresponding author

ICUIMC'12, February, 20–22, 2011, Kuala Lumpur, Malaysia

General Terms

Algorithm, Experimentation

Keywords

Recommendation System, Social Group, Sparsity Problem, Media Content

1. INTRODUCTION

Since the late 20^{th} century, the Internet has become a useful tool, and most Internet users search for information on the Web every day. Users access Web services such as Google and Yahoo to search for images, songs, or video content available on the Web. In the Web 2.0 era, applications such as YouTube or Wikipedia require that users upload their information to the Internet, which has resulted in the tremendous amount of information available on the Web. This trend has also led to the appearance of social networks on the Web. Internet users can upload content to the Web directly and other users can subsequently access the uploaded content, thereby forming a social network between users on the Web. However, the vast amount of data on the Web is often an obstacle to finding information relevant to a particular query due to the existence of spam data and erroneous information, which can decrease the accuracy and reliability of search results. Because of this problem, search results have to be thoroughly scrutinized in order to identify meaningful information. Several researchers have suggested recommendation systems to resolve this problem [5, 8, 13]. When using a recommendation system, users do not need to scan through the entire list of search results because the recommendation system filters the search results and presents users with only those that are most relevant. In Web 2.0, recommendation systems often rely on the collaborative filtering approach [1, 2, 10]. The collaborative filtering approach considers user information such as ratings, age, or preferences when filtering results. Given the Web characteristics that contribute to the formation of social networks with respect to different types of content, a recommendation system based on collaborative filtering is

one usable approach that employs the properties of a social network in making a recommendation. This is because the neighbor user in the traditional approach is based on the shape of the user network and the content. The neighbor means a group of users who have similar preferences with particular user. Because the traditional collaborative filtering approach in the setting of a social network is based on user information, recommendation systems based on collaborative filtering may not perform well if there is not enough user information available [4]. One such problem is sparsity, which occurs when there is not enough user information. A lack of user information compromises the accuracy of the recommendation results. Because the neighbors are chosen according to only a few parameters, the lack of user information likens the neighbor selection process to random selection. Thus, the results of a recommendation based on this selected neighbor are very inaccurate. We suggest a recommendation method using a new approach: a subset of a rater group. We used the GroupLens movie database to implement and test the proposed method. Thus, the rater group in this case was a group of users who evaluated movies. We divided the rater group into subsets using our suggested method, and test results indicated that this method is a feasible alternative that solves the sparsity problem. In Section 2, we describe the traditional recommendation system, and in Section 3 we explain our approach. The results of tests performed using our approach and the traditional method are presented in Section 4. Finally, we conclude this paper in Section 5.

2. RELATED WORKS

2.1 Collaborative filtering based on user preferences

A collaborative filtering approach based on user preferences involves three steps. These steps are explained in detail in the following sub-sections.

2.1.1 Calculating the correlation coefficient and Selecting Neighbors

Pearson correlation coefficient [2, 10, 9] is used to determine the correlation between the preferences of the user who is seeking a recommendation and other users as follows:

$$\rho_{xy} = \frac{cov(X,Y)}{\sigma_X \sigma_Y} = \frac{\sum(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum(X_i - \bar{X})^2}\sqrt{\sum(Y_i - \bar{Y})^2}}, \quad (1)$$

where X is the user who needs recommendations, $\bar{X}$ is the average rating of X, X_i denotes the rating for the ith item by user X, and Y represents the other users.

Figure 1 is an example of the first step. In this figure, the Pearson correlation coefficient between U_x and U_1 is -1, and that between U_x and U_2 is 0.94. This indicates that U_2 has similar preferences to U_x. In the next step, neighbors are chosen using the results of Equation 1. In this step, a correlation coefficient value close to 1 is first selected as the threshold. Users with a correlation coefficient to U_x greater than this threshold are selected as neighbors.

2.1.2 Predicting preferences

The final step is to predict preferences based on the ratings of neighbors. This step uses Equation 2:

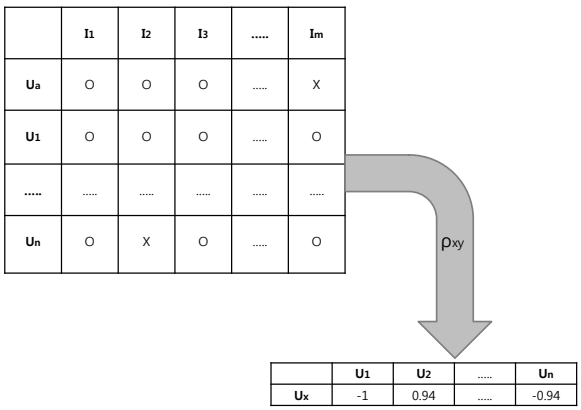

Figure 1: Calculation of Pearson correlation coefficient

$$P = \bar{X} + \frac{\sum_{Y \in raters}(Y_n - \bar{Y})\rho_{xy}}{\sum_{Y \in raters}|\rho_{xy}|}, \quad (2)$$

where $\bar{X}$ is the average rating of user X and Y_n is the rating given by the other users for the n^{th} item. Y is the average rating given by the neighbors of X for the current item. Finally, ρ_{xy} is the Pearson correlation coefficient between X and the other users Y. The raters are a set of users who input ratings for the item of interest. The result P in Equation 2 is the predicted value of an item for user X.

2.2 Known issues associated with user preference based collaborative filtering

The collaborative filtering approach is based on user preferences, which may cause some problems. First, there is the sparsity problem; this occurs when there is not enough user preference data available. If we recommend an item using neighbors selected based on a small number of ratings, then the accuracy of the recommendation will be lower than that obtained using neighbors selected based on a large number of ratings [2, 10]. The sparsity problem affects the accuracy of a recommendation determined using collaborative filtering [5, 8, 13]. The second problem is the cold-start problem. This occurs when new users or items lacking sufficient rating information are added to the database used by the recommendation system [6]. In the traditional collaborative filtering approach, neighbors are only selected from among users with ratings. This implies that a user without ratings cannot have a neighbor. In other words, the system cannot recommend an item to a new user who has no ratings before he/she inputs more than a certain number of ratings. Thus, the system has to wait until all users rate enough items before recommending items [6, 11, 12]. In this study, we focus on solving the sparsity problem using a recommendation algorithm.

2.3 Collaborative Filtering Approaches for the Sparsity Problem

2.3.1 Trust inference for the sparsity problem

Some studies have been conducted to alleviate the spar-

sity problem. Papagelis et al. [7] attempted to resolve the sparsity problem using the trust inference model. Trust inference refers to transitive associations between users. For example, source user S and target user T are linked through intermediate user N; user N is directly associated with users S and T. Thus, there are many co-rated items between user S and user N. In contrast, user T and user S do not have any co-rated items. Figure 2 illustrates the relationships between each user.

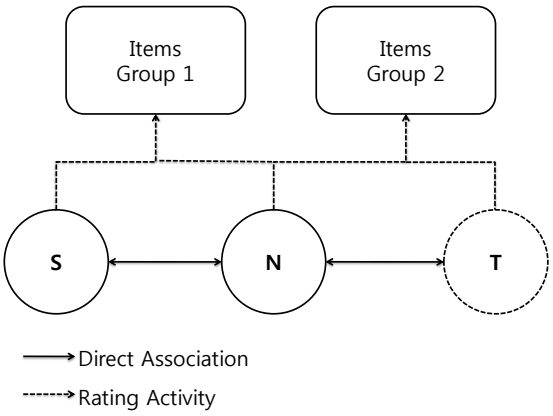

Figure 2: Rating relationship between users and items

Similarities between user S and T cannot be found using the traditional recommendation system due to sparse data. Although user S and T have rated many of the same items as user N, it is impossible to calculate the similarity between user S and user T since they have no co-rated items. Papagelis et al. [7] resolved this sparsity problem using the intermediate user. Specifically, they used the paths between the intermediate user and users S and T to determine the relationships between users S and T, that is, they used trust inference to identify relationships between users.

2.3.2 *A random walk method for the sparsity problem*

Hilmi et al. [14] suggested a novel item-based algorithm to alleviate the sparsity problem. Their goal was to enhance the similarity between items under sparse data conditions using the random walk method in order to more accurately predict user preferences The random walk method infers transition probabilities between similar items according to a finite length of a random walk through the item space to compute predictions. This method consists of three elements. The first is a similarity graph generated based on the items. Each edge in this graph corresponds to the similarity of items and each item is represented by a node; a graphical representation of the neighbors in the traditional approach. The second element is the result of a simulated random walk based on the computed rank values of items for each user. The last element is the rank scores interpreted as ratings for each user-item pair.

3. OUR APPROACH

We use a clustering method that divides raters in a database into sub-groups. Our proposed approach consists of two steps. The first step is selecting similar users, and this is done using methodology similar to that used in the traditional recommendation system. The second step is selecting an item to recommend based on the ratings of the similar users. The first step of our approach differs from the traditional method in that Pearson correlation coefficient is not used to select similar users. Instead, we use concordance between ratings given by specific users. The second step of the proposed method is also different from the traditional approach. Because Pearson correlation coefficient is not used in the first step, the predictions made by the proposed method differ from those made by the traditional system. The prediction process in the proposed system involves the use of only ratings given by similar users, while the traditional recommendation system uses ratings and the Pearson correlation coefficient.

3.1 Selecting Similar Users

To select similar users, we build an nxm matrix using a database.

	I_1	I_2	I_3	.....	I_m
U_a	O	O	O	.....	X
U_1	O	O	O	.....	O
U_2	X	O	O	.....	O
.....	.....	.....	.....	.....	.....
U_n	O	X	O	.....	O

Figure 3: User-item matrix

In Figure 3, the items in each row are movies. Thus, there are a total of m movies and n users in the matrix. O or X marks in the matrix denote whether or not a specific user watched the movie, in other words, the O means that the user evaluated the movie and the X means otherwise. Thus, the total area marked with an X is the *sparse* area. Next, we select a probe user to recommend items. In Figure 3, user U_A is the probe user. We extract all movies rated by the probe user, and randomly select a movie as a probe item that is used to predict a movie recommendation. Then, we construct a sub-matrix to determine whether or not the user evaluated the movie. Figure 4 illustrates this process.

In Figure 4, the matrix on the left is same as the matrix in Figure 3, and the matrix on the right is the sub-matrix used to determine whether or not the users other than the probe user evaluated the probe movie. In the Figure 4, probe user is user U_A and probe item is item I_m. We select item that has no rating for probe user, since we predict the rating of probe item. Thus, user U_A have no rating for item I_m. The rows in the sub-matrix are users other than the probe user. We only include users who evaluated the probe movie in the rows of the sub-matrix. The columns in the sub-matrix are items rated by the probe user, and an O means the users in

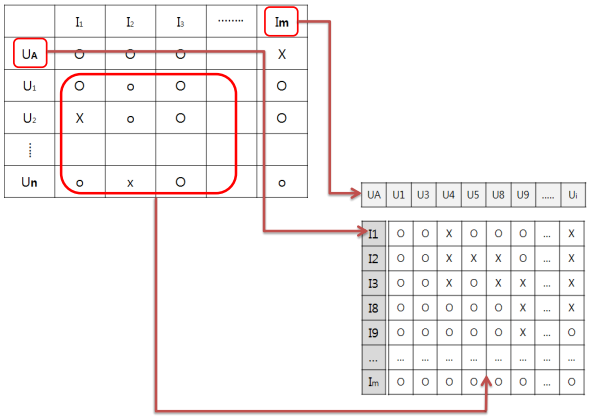

Figure 4: Constructing a sub-matrix

the sub-matrix evaluated the same movie as the probe user while the X means they did not. Next, we extract lists of users that evaluated items for each movie. Figure 5 is an example of this process.

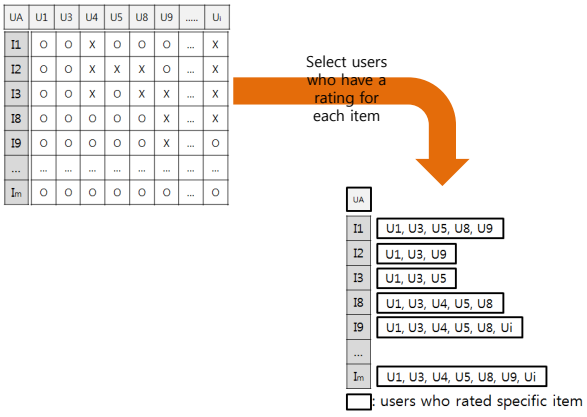

Figure 5: Constructing a user list for each item

In Figure 5, the items in each column have their own corresponding list. The elements in the list are the users who evaluated the movie. For example, in Figure 5, the list for item I_1 consists of users U_1, U_3, U_5, U_8 and U_9. Likewise, the list for item I_2 consists of users U_1, U_3, and U_9. Specifically, users U_1, U_3, U_5, U_8 and U_9 are those who have an O for item I_1, and users U_1, U_3, and U_9 are those users who have an O for item I_2. it means that the users U_1, U_3, U_5, U_8 and $U9$ have a rating for item I_1 , and the users U_1, U_3, and U_9 have a rating for item I_2 Finally, we select similar users using the ratings given by the users on each list. Figure 6 illustrates the process for selecting similar users.

Figure 6 is an example of the process for selecting similar users for item I_1. For example, there are five total users who evaluated item I_1: U_1, U_3, U_5, U_8, and U_9. The rating given by the probe user for item I_1 was 3. The table on the right in Figure 6 shows the ratings given by the five users. In this situation, users U_3, U_8, and U_9 can be selected as similar users since rating of probe user is 3 and users U_3, U_8, and

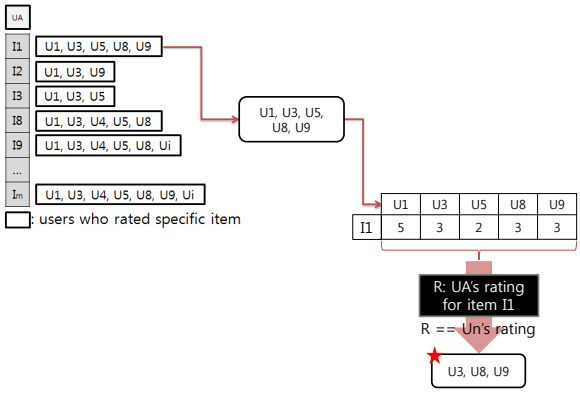

Figure 6: Process for selecting similar users for item I1

U_9 are also 3 for the probe item. In summary, we first select the probe user. Then, we randomly select a probe item from the database, and subsequently group items rated by the probe user. We extract users who evaluated an item at least once based on a list of items. Finally, we select similar users using ratings given by the extracted users and the probe user. Two changeable variables exist in our approach. The first variable corresponds to the degree of overlap between rated items. Specifically, we include users who evaluated items also rated by the probe user using a list of selected items as shown in Figure 5. The second variable represents the range of ratings. We determine the range of error for a specific item, and then select similar users using this range. For example, in Figure 6, the rating given by the probe user for item I_1 is 3, and we set the range of error to be ± 1. In this case, we can select users U_3, U_5, U_8, and U_9 as similar users. The reason is that the range of error is ± 1 and rating of probe user for probe item is 3. The rating of users U_3, U_8, and U_9 for probe item are 3. Thus, these three users are selected as similar users. Additionally, user U_5 is also selected as similar user since rating of user U_5 for probe item is 2, and this value is in the range of error of probe user. In other words, rating of probe user is 3 and range of error is ± 1. thus the rating 2 is in the range of error.

If we do not select a range, we can only select users U_3, U_6, and U_9 as similar users.

3.2 Determining the Prediction Score

To calculate the prediction score, we calculate the average of the ratings given by similar users for the probe item. This average rating is assigned as the prediction score. We compute the prediction score using Equation 3.

$$P = \frac{\sum_{U \in raters} U_r}{n}, \tag{3}$$

In Equation 3, n is the number of similar users, and corresponds to the ratings given by similar users for the probe item.

4. EXPERIMENT AND ANALYSIS

4.1 Database

We used an open movie database called GroupLens [1]. The GroupLens database has three sub-databases: a movie dataset, a user dataset, and a rating dataset. Table 1 summarizes the characteristics of each dataset.

Table 1: GroupLens database

Database	Attributes
Movie Dataset	MovieID::Title::Genre
User Dataset	UserID::Gender::Age::Occupation::Zip-code
Rating Dataset	UserID::MovieID::Rating::Timestamp

The movie dataset contains IDs, titles, and genre combinations for all movies in the database. There were a total of 3,883 movies in the database at the time of this study. The user dataset contains IDs, genders, ages, occupations, and zip codes of all users, and there were 6,040 total users at the time of this study. The rating dataset contains user IDs, movie IDs, and timestamps for all ratings.

4.2 Testing of Our Approach and the Traditional Recommendation System

4.2.1 *Comparison of Recommendation Results*

We first implemented the traditional user based recommendation system using Pearson correlation coefficients. Figure 7 illustrates the testing process for the traditional recommendation system.

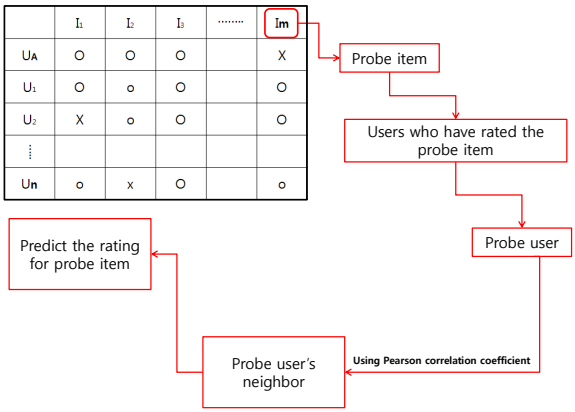

Figure 7: Testing process for the traditional recommendation system

We first randomly selected an item in the database as the probe item. Then, we selected one of the users who had evaluated the probe item as the probe user. We chose one of the users in the rater group for the probe item because we wanted to compare the prediction score determined by our approach and the traditional system to the real rating given by the probe user. In other words, we evaluated the performance of the two approaches in terms of the difference between the real rating given by the probe user and the prediction score computed using the recommendation systems. We used Equation 4 to compute this difference.

$$RS = \sum_{i=1}^{n} |f_i - y_i|, \tag{4}$$

In Equation 4, f_i is the i^{th} real rating of the probe user, and y_i is the i^{th} prediction score. Next, we calculated the Pearson correlation coefficient corresponding to the relationship between the probe user and the other users. When the Pearson correlation coefficient was greater than 0.5, we selected that user as a neighbor. Finally, we calculated the prediction score using the ratings given by the neighbor. We performed a total of 1,000 test iterations, and then calculated the overall average difference between the actual user ratings and the prediction scores. We used Equation 5 to determine the mean absolute error.

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |f_i - y_i|, \tag{5}$$

In Equation 5, f_i and y_i are the same as in Equation 4 and n represents the total test iterations (1,000). The result of MAE means that average difference between real rating and prediction score. If the MAE value is close to 0 then we can consider that the prediction score is precise. The method used to test our approach was very similar to that used for the traditional method. We first selected a probe item and then extracted a user from the rater group for the probe item. Next, we selected items rated by the probe user and constructed a sub-matrix using a list of items. Finally, we selected similar users using the sub-matrix and used Equation 2 to determine the prediction score. We used two separate testing conditions to compare the approaches. First, probe items were randomly selected by the traditional method and our approach, and the resultant MAE prediction values based on these randomly selected items were compared. Second, the same probe items were used by each system in determining the prediction values. This test condition enabled us to more directly compare the accuracy of the two approaches than the first condition.

Table 2: MAE values for our approach and the traditional method when the probe items were randomly selected

MAE Value for Our Approach			MAE Value for the Traditional Method
#	Case 1	Case 2	
1	0.8531	0.8522	0.8796
5	0.8407	0.8467	
10	0.8460	0.8506	
15	0.8521	0.8450	
20	0.8510	0.8081	

4.2.2 *Comparison based on randomly selected probe items*

Table 2 shows the MAE values for our approach and the traditional method based on randomly selected probe items. The numbers in the # column correspond to the number of

[1]http://www.grouplens.org/node/12

ratings that overlap with the probe user's items. We selected 1, 5, 10, 15, and 20 as the overlap variables. In $Case1$, the user's rating was equal to that given by the probe user, and in $Case2$ the user's rating differed by 1 from that given by the probe user. The right side of Table 2 shows the rating predicted by the traditional recommendation system, which was 0.8796.

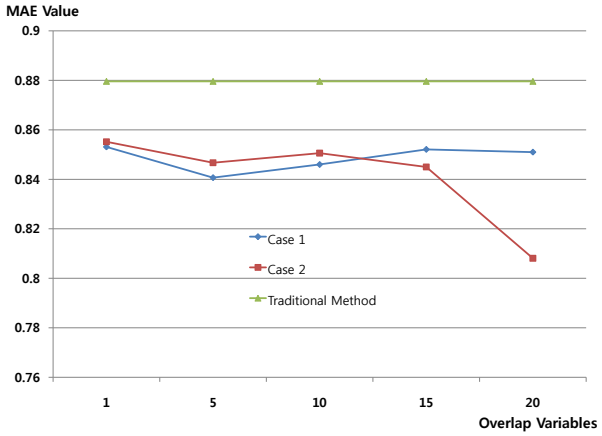

Figure 8: MAE values for the traditional method with respect to overlapping variables of our approach

Figure 8 is a graph of the data in Table 2. The overlapping variables of our approach are on the $x-axis$ and the MAE values are given on the $y-axis$. We performed a total of 1,000 test iterations and calculated the MAE values based on the results of these 1,000 iterations. The single curve corresponding to the traditional method in the Figure 8 represents the difference between two curves corresponding to the MAE values determined by the traditional method for $Cases1$ and 2. The variables on the $x-axis$ were applied for both $Case1$ and $Case2$. Thus, the overlap variables on the $x-axis$ cannot be applied to the composite curve for the traditional method. In Figure 8, all MAE values determined by our approach are less than those determined by the traditional method. The maximum value achieved with our approach was 0.8552, indicating an improvement in the prediction made using our approach.

4.2.3 Comparison based on the same probe items

In this test of our approach, we used the same conditions as those used for the comparison based on randomly selected probe items. We chose 1, 5, 10, 15, and 20 as the overlap variables and these were also applied to Cases 1 and 2. The only selectable variable in the traditional method was the threshold of the Pearson correlation coefficient. We set the threshold to be 0.5, that is, we selected users with a greater than 0.5 correlation as neighbors. We computed a total of ten MAE values. These ten MAE values were calculated using separate test iterations. We computed MAE values for different numbers of test iterations: 100, 200, 300, âĂę, and 1,000.

Figure 9 is a graph of MAE values determined using our approach and the traditional method based on the same probe items when the overlap variable was 1 for $Cases1$ and 2. Each graph illustrates the MAE values when different numbers of test iterations were applied. In each graph, the $x-axis$ indicates the number of test iterations and the MAE values are denoted on the $y-axis$. In the top graph, the overlap variable is 1 for $Case1$ and in the bottom graph the overlap variable is 1 for $Case2$. In each graph it is evident that MAE values calculated using our approach are better than the MAE values determined using the traditional method when the same probe items are used. Additionally, the performance of our approach remains more stable than that of the traditional approach as the number of test iterations increases. In $condition1$ of Figure 9, the MAE values determined by our approach are stable when the number of test iterations is greater than 300. In comparison, the MAE values determined by the traditional method are stable when the number of test iterations is greater than 500. In $condition2$ of Figure 9, our approach yields stable results when the number of test iterations is greater than 200 while the results of the traditional method are stable at greater than 600 iterations. These findings suggest that stable results can be achieved when fewer items are used to predict recommendation scores using our approach under $conditions1$ and 2.

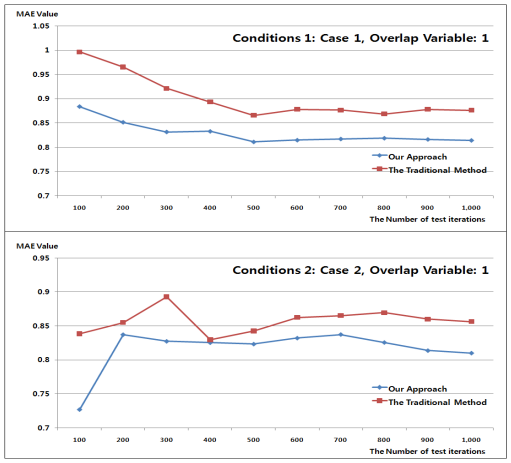

Figure 9: MAE values determined using our approach and the traditional method based on the same probe items when the overlap variable was 1 for Cases 1 and 2

Figures 10, 11, 12, and 13 are the graphs of MAE values determined using our approach and the traditional method based on the same probe items when the overlap variable was 5, 10, 15, and 20 for Cases 1 and 2. In each graph, $x-axis$ and $y-axis$ are same with the Figure 9. In the top graph, Case is 1 and in the bottom graph Case is 2. Most MAE values calculated using our approach in each figures are better than the results of traditional method. This implies that our approach is more precise than previous method in same condition. Although we can find similar stable points, MAE values that calculated using our approach are better than the traditional method in all case of this. The stable points mean the number of iterations that MAE values do not change unpredictably. Thus, use of the recommendations determined by our approach to supplement a sparse area can yield more accurate recommendation results, thereby providing an alternative solution for the sparsity problem.

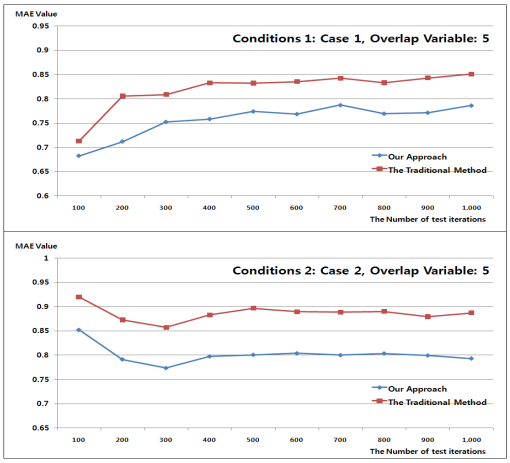

Figure 10: MAE values determined using our approach and the traditional method based on the same probe items when the overlap variable was 5 for Cases 1 and 2

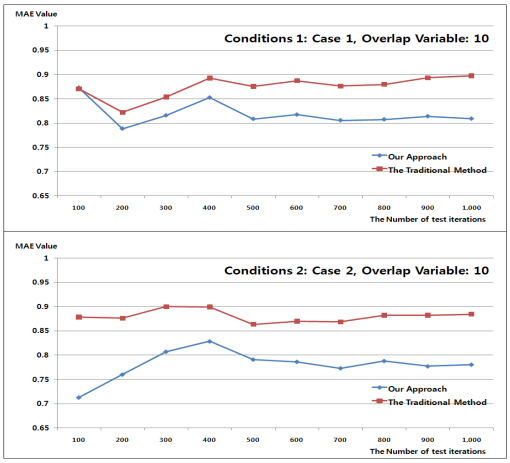

Figure 11: MAE values determined using our approach and the traditional method based on the same probe items when the overlap variable was 10 for Cases 1 and 2

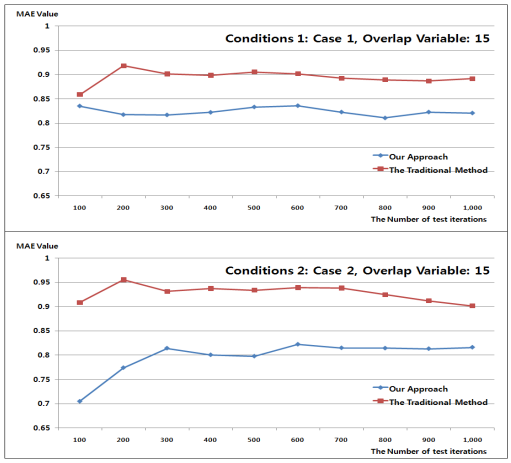

Figure 12: MAE values determined using our approach and the traditional method based on the same probe items when the overlap variable was 15 for Cases 1 and 2

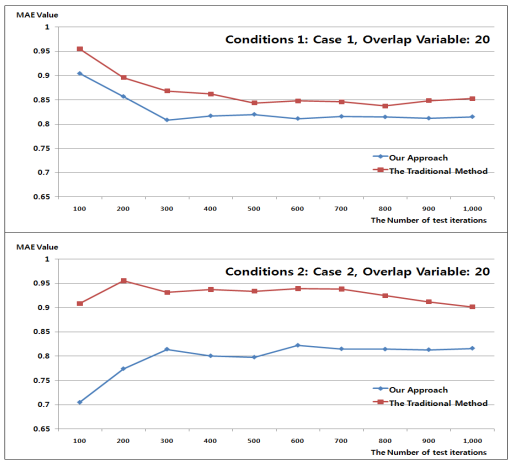

Figure 13: MAE values determined using our approach and the traditional method based on the same probe items when the overlap variable was 20 for Cases 1 and 2

4.2.4 *Solving the Sparsity Problem Using Our Approach*

We sought to determine whether our approach could solve the sparsity problem using a test dataset comprised of a probe item, a training user, training items, and actual items. The probe item was defined as the item to be recommended, and the training user was one of the users who had evaluated the probe item. Thus, the probe item was first to be extracted from the database. The training items were those items for which a prediction was needed, and we considered the training items to be the sparse area. These training items were randomly selected from a set of items evaluated by the training user. The actual items were the set of items used to make predictions for the training items (sparse area). Actual data was also selected from the set of items evaluated by the training user. To summarize, we first predicted ratings for the training items, and then we used ratings for the actual items in conjunction with the predicted ratings for the training items to predict user preference for the probe item. Figure 14 illustrates the components of the test dataset used to determine whether our approach could overcome the sparsity problem and Figure 15 is a flow-chart summarizing the testing process.

Two values were computed in this experiment. The first was the predicted rating for the probe item based on the ratings of actual items and the predicted ratings for the training items. The second was the predicted rating for the probe item based on real ratings given by the training user. Both values were calculated based on the same items, and the only difference in the computations was the ratings of the training items. When we calculated the first value, we use predicted ratings for the training items while the second value was determined based on real ratings provided by the training user. We calculated each value a total of 1,000 times, and as with the previous tests, we used two variables, the number of overlapped items and the rating range.

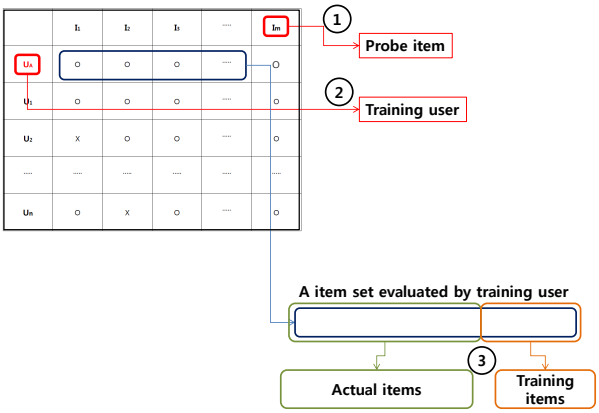

Figure 14: Components of the test dataset used to address the sparsity problem

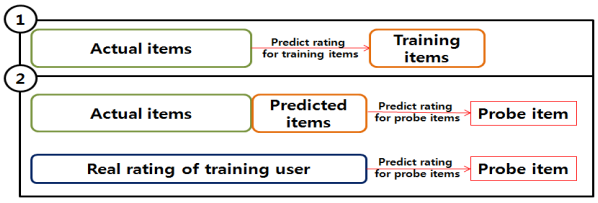

Figure 15: Testing process

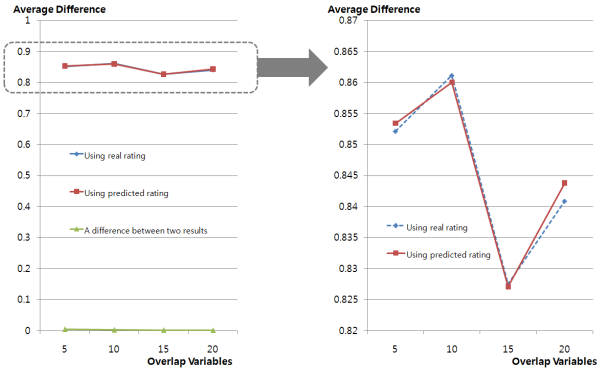

Figure 16: Test result when the rating range was 0

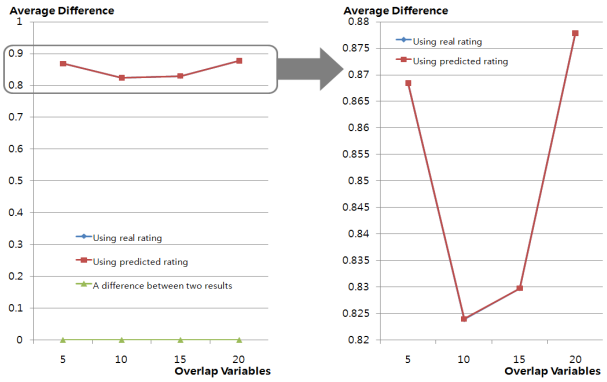

Figure 17: Test result when the rating range was 1

Figure 16 and 17 are graphs of the results for each testing condition. Figure 16 shows the results for a rating range of 0 and Figure 17 shows the results for a rating range of ± 1. The results for the sparse area are floating values, and thus the rating range is the difference between the predicted floating value and the rating given by a subset of users, which is less than 1. In each graph, the number of overlapped items is on the $x-axis$ and the average difference between the predicted recommendation result and the real rating of the probe item is on the $y-axis$. 'Real rating' refers to the determination of recommendation results for the probe item based on real ratings given by the training user for the training items, and 'predicted rating' refers to the recommendation results determined based on predicted ratings for the training items. 'Difference between the two results' denotes a difference between the recommendation results for the probe item. The values plotted in Figures 16 and 17 are listed in Tables 3 and 4.

There was only a slight difference between each result, suggesting that accurate recommendation results can be expected, despite the fact that predicted ratings were used to supplement the sparse area. This is because we used a subset of the rater group. For example, when user A wants a recommendation but has only a few ratings, our recommendation system selects a subset of users from the rater group and determines the difference between user A's ratings and those of the other subset of users. It is evident in Table 4 that when the rating range is ± 1, the difference between the average ratings determined by our recommendation system and the actual user-assigned ratings was less than 1. Thus, if we fill the sparse area based on ratings predicted by our proposed method, then recommendation results similar to those based on real ratings can be expected. As a consequence, our proposed method solves the sparsity problem that commonly affects recommendation systems.

5. CONCLUSIONS

Traditional recommendation systems require a certain size of user preference data to identify groups of users and recommend items based on these groups. If there is not sufficient data available, then the system becomes very unreliable because of the sparsity problem. To solve this problem, we have suggested a recommendation system that uses a subset

Table 3: Table for Test Result When the Rating Range was 0

The number of overlap	Real rating	Predicted rating	Difference between the two results
5	0.852157	0.853437	0.00425
10	0.861166	0.860046	0.00226
15	0.827485	0.827054	0.001437
20	0.840886	0.8438	0.00078

Table 4: Table for Test Result When the Rating Range was 1

The number of overlap	Real rating	Predicted rating	Difference between the two results
5	0.868465	0.868463	0.000055
10	0.823885	0.823971	0.000086
15	0.829737	0.829737	0
20	0.87782	0.877867	0.000044

of a particular rater group. Herein we compared the performance of our approach and the traditional recommendation system in the context of the sparsity problem. First, we calculated MAE values based on randomly selected probe items and compared the MAE values generated using each approach based on these selected probe items. Additionally, we calculated MAE values using both approaches based on the same probe items, which allowed a more direct comparison of MAE values between the two approaches. The results of each comparison suggested that our approach is superior to the traditional method. Second, we demonstrated that our method can alleviate the sparsity problem. We selected training items and used our method to determine prediction scores based on this items. Then, we used these predicted results as actual items. Finally, we compared MAE values calculated using real users' ratings and predicted items. We determined that there are small differences between results using real users' ratings and predicted scores. Further, the proposed system outperforms the traditional recommendation system when data are sparse and thus alleviates the sparsity problem. In the future, we aim to apply our approach to larger open databases and to conduct experiments with more users. We also plan to utilize the open APIs of content-sharing sites with category information, such as Yahoo Music or YouTube, in our future studies.

6. ACKNOWLEDGMENTS

The Yonsei university research group was supported by the Basic Science Research Program through NRF funded by MEST (2010-0009168).

7. REFERENCES

[1] R. M. Bell and Y. Koren. Lessons from the netflix prize challenge. *SIGKDD Explorations*, 9(2):75–79, 2007.

[2] D. Billsus and M. J. Pazzani. Learning collaborative information filters. In *ICML*, pages 46–54, 1998.

[3] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *SIGIR*, pages 230–237, 1999.

[4] K. Honda, A. Notsu, and H. Ichihashi. Collaborative filtering by sequential extraction of user-item clusters based on structural balancing approach. In *Fuzzy Systems, 2009.*, pages 1540–1545, 2009.

[5] Z. Huang, H. Chen, and D. D. Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):116–142, 2004.

[6] M. Ishikawa, P. Géczy, N. Izumi, T. Morita, and T. Yamaguchi. Information diffusion approach to cold-start problem. In *Web Intelligence/IAT Workshops*, pages 129–132, 2007.

[7] M. Papagelis, D. Plexousakis, and T. Kutsuras. Alleviating the sparsity problem of collaborative filtering using trust inferences. In *iTrust*, pages 224–239, 2005.

[8] A. Popescul, L. H. Ungar, D. M. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *UAI*, pages 437–444, 2001.

[9] B. Sarwar, G. Karypis, J. Konstan, and J. Rie. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295, 2001.

[10] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Analysis of recommendation algorithms for e-commerce. In *ACM Conference on Electronic Commerce*, pages 158–167, 2000.

[11] A. I. Schein, A. Popescul, L. H. Ungar, and D. M. Pennock. Methods and metrics for cold-start recommendations. In *SIGIR*, pages 253–260, 2002.

[12] T. Y. Tang and G. I. McCalla. Utilizing artificial learners to help overcome the cold-start problem in a pedagogically-oriented paper recommendation system. In *AH*, pages 245–254, 2004.

[13] D. C. Wilson, B. Smyth, and D. O'Sullivan. Sparsity reduction in collaborative recommendation: A case-based approach. *IJPRAI*, 17(5):863–884, 2003.

[14] H. Yildirim and M. S. Krishnamoorthy. A random walk method for alleviating the sparsity problem in collaborative filtering. In *RecSys*, pages 131–138, 2008.