A Recommender System based on an Improved Simultaneous Selection Method of Query Items and Neighbors

Koki Miura, Mitsuru Takeuchi, and Yoshifumi Okada

***Abstract*—User-based collaborative filtering (CF) is the most popular and basic recommendation approach. It selects *k*-nearest neighbors with similar preferences to an active user and recommends items that are rated highly by those neighbors. In user-based CF, it is important to use the best possible query items (items input into the system) and neighbors to provide high recommendation accuracy. Our previous paper proposed a simultaneous selection method for query items and *k*-nearest neighbors for each active user. This method identifies adequate query items and *k*-nearest neighbors by repeatedly updating the selection. In this paper, we introduce new termination conditions for this updating process, which consider the quality of the recommendation, and develop a recommender system by implementing these new termination conditions. In this study, two experiments are conducted: 1) a comparison of the recommendation accuracy of this new method with that of the previous method and 2) an investigation of the effectiveness of the item/neighbor updating process. The result shows that the proposed method can significantly improve the recommendation accuracy compared to the previous method and can correctly recommend suitable items for more active users.**

***Index Terms*—recommender system, user-based collaborative filtering, item selection, neighbor selection**

I. Introduction

Recommender systems have been used successfully for a wide variety of applications. To date, many recommendation algorithms have been proposed [1]-[7]. In terms of practical applications, collaborative filtering (CF) has been the most successful approach [8]-[14]. User-based CF is particularly popular owing to its effectiveness and ease of implementation [15]. It selects the *k*-nearest neighbor users (hereafter “neighbors”) to the active user (i.e., the user asking for recommendations) based on preference similarities. This process is referred to as neighbor selection. Then, items that are rated highly by the selected neighbors are recommended to the active user. In the neighbor selection process, preference similarity between the active user and the other users is computed using their rating data for all items (hereafter “query items”) preferred by the active user. This is based on the assumption that there exist neighbors with ratings that are similar to those of the active user for all query items. However, the above assumption is not always valid because the number of items in a database is generally very large.

It is important to select the most appropriate query items and neighbors to provide the highest possible recommendation accuracy. In our previous paper [16], we proposed a simultaneous selection method for identifying adequate query items and neighbors for each active user. This method was based on an algorithm that repeatedly updates the following selection processes.

- Selection of items that are useful for discriminating between neighbors and non-neighbors
- Selection of neighbors using those query items

Recommendations for each active user were determined using the final selected query items and neighbors.

However, in that study, we employed a simple termination condition based on the number of updates or the number of items to be selected; hence, the quality of the final recommendation was not considered at all. In this study, we implement new termination conditions that evaluate the quality of the final recommendations. The main contributions of this study are as follows:

- Improving the recommendation accuracy of the previous method by employing query items/neighbors based on recommendation quality
- Developing a new user-based collaborative filtering system based on the proposed method

II. Method

In the proposed method, neighbor selection is first executed for each active user. Figure 1 shows the procedure in more detail. The details of the procedure are also described further in the text below.

A. Input

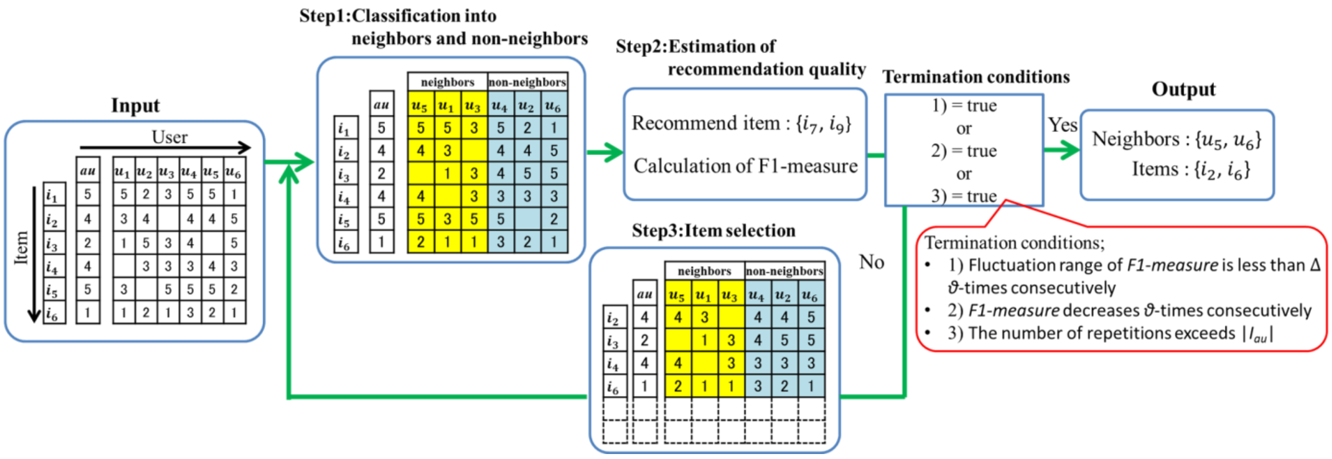
As input, the method requires a rating matrix in which each row is an item, each column is a user, and each element is rating data. Hereafter, a set of items rated by an active user *au* is represented by I_{au} and a set of items selected by the method is denoted by I_q. Initially, I_q is set equal to I_{au}. The set of users excluding the active user is denoted by $u_j \in U$.

Manuscript received July 13, 2016.

K. Miura is with Division of Information and Electronic Engineering, Muroran Institute of Technology, 27-1, Mizumoto-cho, Muroran, Hokkaido 050-8585, Japan (e-mail: miura@cbrl.csse.muroran-it.ac.jp).

M. Takeuchi is with Division of Information and Electronic Engineering, Muroran Institute of Technology, 27-1, Mizumoto-cho, Muroran, Hokkaido 050-8585, Japan (e-mail: takeuchi@cbrl.csse.muroran-it.ac.jp).

Y. Okada is with College of Information and Systems, Muroran Institute of Technology, 27-1, Mizumoto-cho, Muroran, Hokkaido 050-8585, Japan (corresponding author; telephone: +81-143-5408; fax: +81-143-5408; e-mail: okada@csse.muroran-it.ac.jp).


Input
User
Item
Step1:Classification into neighbors and non-neighbors
Step2:Estimation of recommendation quality
Recommend item : $\{i_7, i_9\}$
Calculation of F1-measure
Termination conditions
1) = true or 2) = true or 3) = true
Yes
No
Output
Neighbors : $\{u_5, u_6\}$
Items : $\{i_2, i_6\}$
Step3:Item selection

Input

	au	u_1	u_2	u_3	u_4	u_5	u_6
i_1	5	5	2	3	5	5	1
i_2	4	3	4		4	4	5
i_3	2	1	5	3	4		5
i_4	4		3	3	3	4	3
i_5	5	3		5	5	5	2
i_6	1	1	2	1	3	2	1

Step1

		neighbors			non-neighbors		
	au	u_5	u_1	u_3	u_4	u_2	u_6
i_1	5	5	5	3	5	2	1
i_2	4	4	3		4	4	5
i_3	2		1	3	4	5	5
i_4	4	4		3	3	3	3
i_5	5	5	3	5	5		2
i_6	1	2	1	1	3	2	1

Step3

		neighbors			non-neighbors		
	au	u_5	u_1	u_3	u_4	u_2	u_6
i_2	4	4	3		4	4	5
i_3	2		1	3	4	5	5
i_4	4	4		3	3	3	3
i_6	1	2	1	1	3	2	1

Termination conditions;
- 1) Fluctuation range of *F1-measure* is less than Δ ϑ-times consecutively
- 2) *F1-measure* decreases ϑ-times consecutively
- 3) The number of repetitions exceeds $|I_{au}|$

Fig. 1. Procedure of the proposed method.

B. Classification into Neighbors and Non-neighbors

The similarity between an active user *au* and another user u_j is measured by treating each user as a vector (hereafter a "rating vector") in the rating data and computing the cosine of the angle formed between the rating vectors. The cosine similarity between *au* and u_j is expressed as follows:

$$sim(au, u_j) = \frac{\overrightarrow{au} \cdot \overrightarrow{u_j}}{|\overrightarrow{au}| \times |\overrightarrow{u_j}|}, \tag{1}$$

where $\overrightarrow{au}$ and $\overrightarrow{u_j}$ are the rating vectors of *au* and u_j, respectively, and $\cdot$ denotes the dot product of the two rating vectors. Note that the similarity is calculated using only items rated by both *au* and u_j.

Subsequently, we sort the other users in descending order of similarity and calculate the median of the similarities. Neighbors and non-neighbors are determined by a predefined threshold. Hereafter, a set of neighbors of *au* is represented by N_{au}. In this study, the threshold is set to be either the median or $1/\sqrt{2}$, whichever value is greater.

C. Estimation of Recommendation Quality

The recommendation quality is estimated using I_q as the query items. In this step, the user-based approach [15] is employed to obtain the recommended items. This approach outputs ranked items to the active user *au* according to the following steps. First, the cosine similarity between *au* and a neighbor user $nu_l \in N_{au}$ is calculated using the rating data of the query items belonging to I_q. The score of an item $i_k \in I_q$ is calculated as follows [15]:

$$score(k) = \overline{au} + \frac{\sum_{nu_l \in Z} \left(sim(au, nu_l) \times \left(nx_{lk} - \overline{nu_l}\right)\right)}{\sum_{nu_l \in Z} |sim(au, nu_l)|}, \tag{2}$$

where Z is a set of neighbors rating i_k. $\overline{au}$ and $\overline{nu_l}$ are the mean ratings for *au* and a neighbor nu_l, respectively, and nx_{lk} is a rating value for i_k by neighbor nu_l. Next, *r* items are recommended in descending order of the above scores.

Subsequently, the recommendation quality is estimated using *F1-measure* [17]. *F1-measure* is calculated using *precision* and *recall*, which are performance evaluation indices for information retrieval systems. These indices are defined as follows:

$$precision = \frac{|A \cap B|}{|A \cap C|}, \tag{3}$$

$$recall = \frac{|A \cap B|}{|B|}. \tag{4}$$

where *A* is the set of recommended items, *B* is the set of items rated highly by the active user *au*, and *C* is the set of items rated by *au*. Note that *precision* is calculated using items rated by *au*. *F1-measure* is calculated using *precision* and *recall* as follows:

$$F - measure = \frac{2 * precision * recall}{precision + recall}. \tag{5}$$

In other words, *F1-measure* is the harmonic mean of *precision* and *recall*. The above three indices take values in the range of 0–1. *F1-measure* is evaluated by the termination conditions described in section *E*.

D. Item Selection

For each item $i_k \in I_q$, we calculate a correlation ratio between neighbors and non-neighbors, which is given as follows:

$$\eta_k^2 = \frac{Sb}{St}, \tag{6}$$

where η_k^2 is the correlation ratio for i_k, *St* is the total variation, and *Sb* is the variation between classes. *St* and *Sb* are calculated as follows:

$$St = \sum_j (x_{jk} - \overline{x(k)})^2, \tag{7}$$

$$Sb = \sum_c \left(\left(\overline{x_c(k)} - \overline{x(k)}\right)^2 \cdot n_c(k)\right). \tag{8}$$

Here, *c* is a class label variable and takes values of 1 or 2, where 1 and 2 correspond to the neighbor user and

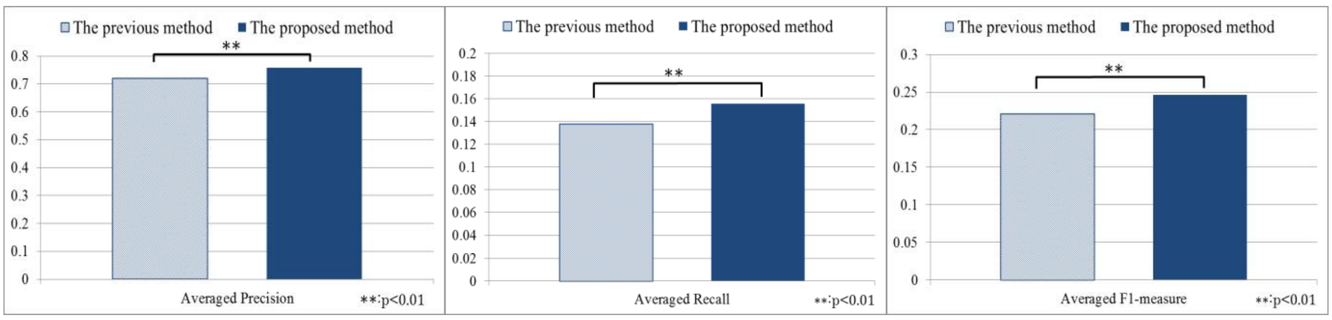


Fig. 2. Comparison results of the recommendation accuracies for MovieLens.

non-neighbor user classes, respectively. x_{jk} is a rating value for i_k of u_j. $\overline{x_c(k)}$ is the mean value of ratings for i_k in class c, and $\overline{x(k)}$ is the mean value of ratings for i_k over the two classes. $n_c(k)$ is the number of users who rate i_k in the two classes.

Subsequently, we compute the $F0$ value to estimate the difference between neighbors and non-neighbors. The $F0$ value for i_k is calculated as follows:

$$F0 = \frac{(n-p)\eta_k^2}{(p-1)(1-\eta_k^2)}, \qquad (9)$$

where n is the number of users rating i_k and p indicates the number of classes. In this study, the items with an *F0* value of 2.0 or more are regarded as the discriminative items between neighbors and non-neighbors. The items in I_q are replaced by these discriminative items.

E. *Item/neighbor Update and Termination Conditions*

In our previous study [16], the following two termination conditions were used:

1) The number of repetitions exceeds $|I_{au}|$
2) There is no item to be selected

In this study, we use *F1-measure* to evaluate the recommendation quality as part of the termination conditions. The proposed method stops the update if at least one of the following three termination conditions is satisfied:

1) The fluctuation range of the *F1-measure* is less than Δ over θ consecutive iterations
2) *F1-measure* decreases over θ iterations consecutively
3) The number of repetitions exceeds $|I_{au}|$

Conditions 1) and 2) have been newly added in this study. Condition 1) means that there is no significant change in the recommendation quality when updating. Condition 2) means that the recommendation quality degrades with updating. Condition 3) is employed to deal with cases other than conditions 1) and 2), such as repetitive large fluctuations.

F. *Output*

The steps detailed in Sections A to E are repeated until at least one of the termination conditions is satisfied. Subsequently, the best update is determined by identifying the best *F1-measure* from the first update to the last one. Finally, the method outputs I_q (i.e., the final selected query items) and N_{au} (i.e., the final selected neighbors) from the best update.

III. Evaluation Experiments

We developed two recommender systems, one implementing the previous method and the other the proposed method. Using two benchmark datasets, we conducted two experiments: 1) a comparison of the new recommendation accuracy with that of the previous method and 2) an investigation of the effectiveness of the item/neighbor updating process.

A. *Datasets and Parameter Settings*

We used two datasets, MovieLens and Jester [18], which are benchmark datasets for recommender systems. The number of users in each dataset was varied for the evaluation experiments. MovieLens comprises about 100,000 rating data (integer values of 1–5) from 900 users for 1,682 movies. Jester consists of about 210,000 rating data (real values in the range from −10.0 to +10.0) from 3,000 users for 100 jokes.

The parameters used in this study are as follows: $r = 100$ for MovieLens, $r = 20$ for Jester, $\Delta = 0.01$, and $\theta = 2$.

B. *Comparison of the Recommendation Accuracy with that of the Previous Method*

We compared the recommendation accuracies of the proposed method and the previous method. The experiment was conducted by three-fold cross-validation [16]. For each active user, we recommended items to the active user using the selected query items/neighbors and computed recommendation accuracies from the three indices. For the respective indices, we calculated the average recommendation accuracy for all active users and compared the two methods.

C. *Effectiveness of the Item/neighbor Update*

We investigated the effectiveness of updating the query items and neighbors. The method with the update is the proposed method and that without the update is the existing user-based approach [15]. In this experiment, the recommendation accuracy of each user was compared between the proposed method and the existing user-based approach.

IV. Results and Discussion

A. *Comparison of Recommendation Accuracy with that of the Previous Method*

Figs. 2 and 3 compare the recommendation accuracies of

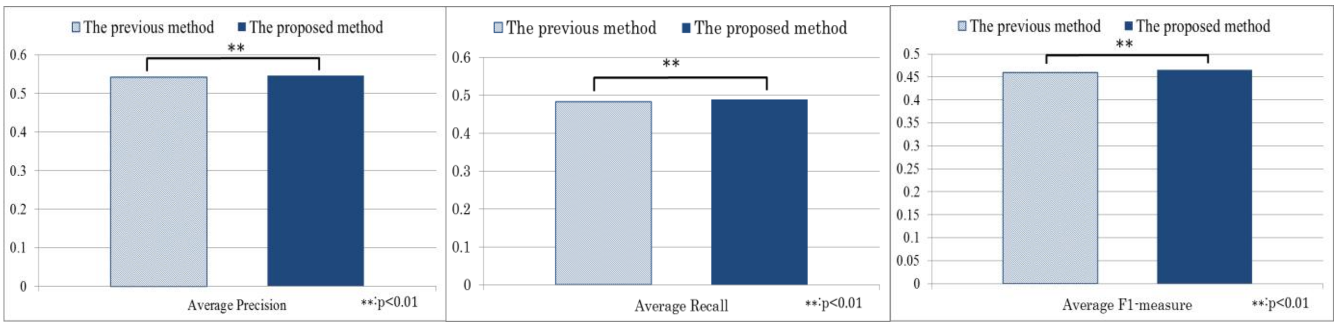


Fig. 3. Comparison results of the recommendation accuracies for Jester.

the proposed method and previous method. In both datasets, the scores of the proposed method are significantly improved compared to those of the previous method. This means that the proposed method can correctly recommend more items suitable for active users by selection of query items/neighbors based on the recommendation quality.

B. *Effectiveness of the item/neighbor update*

Figs. 4 and 5 are the histograms for the number of updates of all active users. As for MovieLens, the updates are completed in a range from 2 to 7 iterations. In Jester, the updates are completed after 2 or 3 iterations for most active users.

Figs. 6 and 7 are the results of the recommendation accuracy comparison. The vertical axis indicates the score of the proposed method and the horizontal axis indicates the score of the method without updating (i.e., the existing user-based method). Each point in the figures indicates an individual active user. In the figure, if the point is located on the upper side of the line, it shows that the recommendation accuracy is improved by the update. In both datasets, most of the points are located on the upper side of the line. This shows that the update contributes effectively to improving the recommendation accuracy.

Tables I and II summarize the number of active users who show an "increase," "no change," or a "decrease" in each index. These tables show that the number of active users whose scores are improved increases significantly for all indices.

V. Conclusion

In [16], we proposed a recommender system method that enables simultaneous selection of adequate query items and neighbors for each active user. In this study, new termination conditions for the update are introduced into the method to account for the recommendation quality. Moreover, a recommender system implementing the new termination conditions was developed. Using two benchmark datasets, we conducted two experiments: 1) a comparison of the recommendation accuracy of the proposed method with that of the previous method and 2) an investigation of the effectiveness of the item/neighbor update process. The result showed that the proposed method can significantly improve the recommendation accuracy compared to that of the previous method and can correctly recommend suitable items for more active users.

In the future, we will implement the forward selection method [19] to identify better combinations of items.

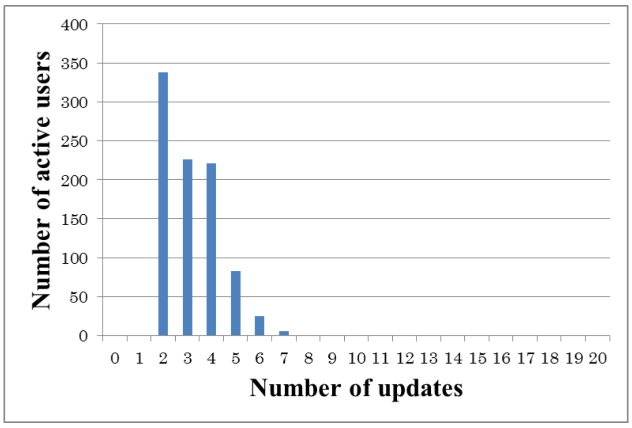


Fig. 4. Histogram of the number of updates across all active users for MovieLens.

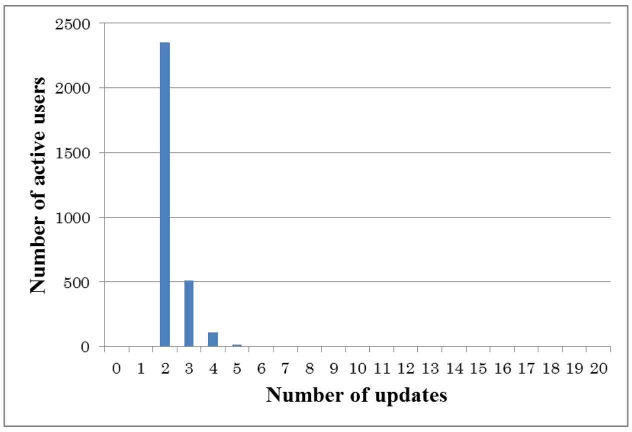


Fig. 5. Histogram of the number of updates across all active users for Jester.

References

[1] D. Bridge, M. Göker, L. McGinty, and B. Smyth, "Case-based recommender systems," The Knowledge Engineering Review, vol. 20, no. 3, pp. 315-320, 2005.

[2] R. Burke, P. Brusilovsky, A. Kobsa, and W. Nejdl, "Hybrid web recommender systems," The Adaptive Web: Methods and Strategies of Web Personalization, Germany, Heidelberg: Springer, pp. 377-408, 2007.

[3] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, "Eigentaste: A constant time collaborative filtering algorithm," Information Retrieval, vol. 4, no. 2, pp. 133-151, 2001.

[4] M. J. Pazzani, "A framework for collaborative, content-based and demographic filtering," Artificial Intelligence Review, vol. 13, no. 5-6, pp. 393-408, 1999.

[5] C. Thompson, M. Göker, and P. Langley, "A personalized system for conversational recommendations," Journal of Artificial Intelligence Research, vol. 21, pp. 393-428, 2004.

[6] K.H.L. Tso-sutter, L.B. Marinho, and L. Schmidt-Thieme, "Tag-aware recommender systems by fusion of collaborative filtering algorithms,"

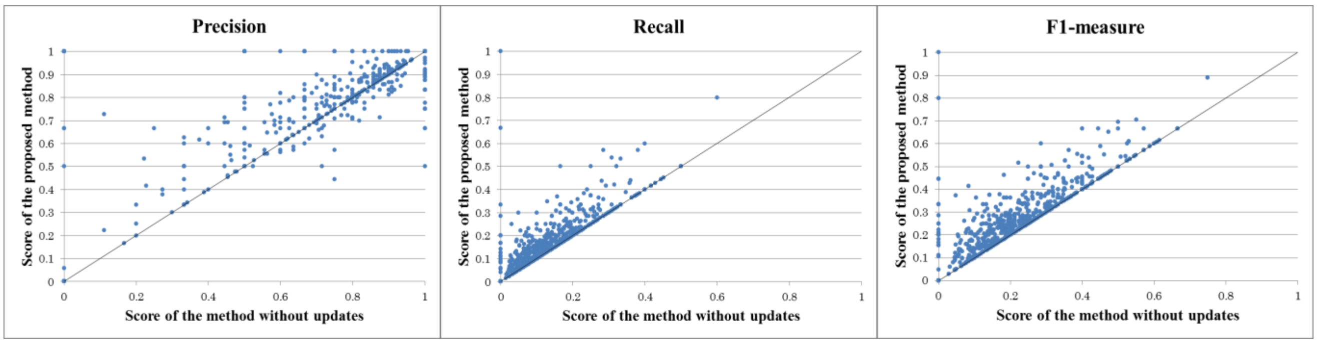


Fig. 6. Results of the recommendation accuracy assessment for MovieLens.

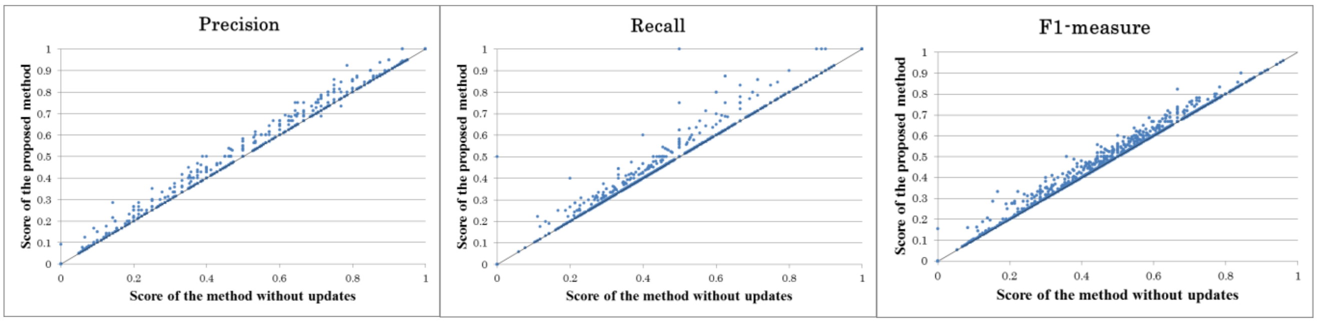


Fig. 7. Results of the recommendation accuracy assessment for Jester.

TABLE I
NUMBER OF ACTIVE USERS IN EACH INDEX FOR MOVIELENS.

index	increase	no change	decrease	rate of increase
precision	227	594	79	0.252
recall	384	516	0	0.427
F1-measure	414	486	0	0.46

TABLE II
NUMBER OF ACTIVE USERS IN EACH INDEX FOR JESTER.

index	increase	no change	decrease	rate of increase
precision	333	2665	2	0.111
recall	202	2798	0	0.067
F1-measure	338	2662	0	0.112

Proceedings of ACM Symposium on Applied Computing Fortaleza, Ceara, Brazil, pp. 1995-1999, 2008.

[7] M. Zanker, “A collaborative constraint-based meta-level recommender,” Proceedings of ACM Conference on Recommender Systems, Lausanne, Switzerland, ACM Press, pp. 139-146, 2008.

[8] B. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, “Item-based collaborative filtering recommendation algorithms,” Proceedings of 10th International Conference on World Wide Web, Hong Kong, ACM, pp. 285-295, 2001.

[9] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P.Kriegel, “Probabilistic memory-based collaborative filtering,” IEEE Transactions on Knowledge and Data Engineering, vol. 16, no. 1, pp. 56-69, 2004.

[10] G. Linden, B. Smith, and J. York, “Amazon.com recommendations: item-to-item collaborative filtering,” Internet Computing, IEEE, vol. 7, no. 1, pp. 76-80, 2003.

[11] J. Herlocker, J. A. Konstan, and J. Riedl, “An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms,” Information Retrieval, vol. 5, no. 4, pp. 287-310, 2002.

[12] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, “Collaborative filtering recommender system,” The Adaptive Web, Lecture Notes in Computer Science, vol. 4321, pp. 291-324, 2007.

[13] M. A. Ghazanfer, and A. Prugel–Bennett, “Building switching hybrid recommender system using machine learning classifiers and collaborative filtering,” IAENG International Journal of Computer Science, vol. 37, no. 3, pp272-287, 2010.

[14] X. Zhong, G. Yang, L. Li, and L. Zhong, “Clustering and correlation based collaborative filtering algorithm for cloud platform,” IAENG International Journal of Computer Science, vol. 43, no. 1, pp108-114, 2016.

[15] X. Su, and T. M. Khoshgogtaar, “A Survey of Collaborative Filtering Techniques,” Advances in Artificial Intelligence, vol. 2009, 2009.

[16] K. Miura, M. Takeuchi, and Y. Okada, "Simultaneous selection method of query items and neighbors in a recommender system", Lecture Notes in Engineering and Computer Science: Proceedings of The International MultiConference of Engineers and Computer Scientists 2016, IMECS 2016, 16-18 March, 2016, Hong Kong, pp. 36-39.

[17] X. Wang, T. Nagashima, K. Fukuta, Y. Okada, M. Sawai, H. Tanaka, and T. Uozumi, “Statistical method for classifying cries of baby based on pattern recognition of power spectrum,” International Journal of Biometrics, vol. 2, no. 2, pp. 113-123, 2010.

[18] GroupLens_Research: http://www.grouplens.org

[19] H. Mitsubayashi, S. Aso, T. Nagashima, and Y. Okada, “Accurate and robust gene selection for disease classification using a simple statistic,” Bioinformation, vol. 3, no. 2, pp. 68-71, 2008.