IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received March 29, 2019, accepted April 27, 2019, date of publication May 2, 2019, date of current version May 23, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2914556

Clustering-Based Collaborative Filtering Using an Incentivized/Penalized User Model

CONG TRAN[1,2], (Student Member, IEEE), JANG-YOUNG KIM[3], WON-YONG SHIN[2], (Senior Member, IEEE), AND SANG-WOOK KIM[4]

[1]Department of Computer Science and Engineering, Dankook University, Yongin 16890, South Korea
[2]Department of Computational Science and Engineering, Yonsei University, Seoul 03722, South Korea
[3]Department of Computer Science, University of Suwon, Hwaseong 18323, South Korea
[4]Department of Computer Science and Engineering, Hanyang University, Seoul 04763, South Korea

Corresponding author: Won-Yong Shin (wy.shin@yonsei.ac.kr)

This work was supported by the National Research Foundation of Korea (NRF) through the Basic Science Research Program, Ministry of Education, under Grant 2017RID1A1A090 00835, and Ministry of Science and ICT, under Grant NRF-2017R1A2B3004581.

ABSTRACT Giving or recommending appropriate content based on the quality of experience is the most important and challenging issue in recommender systems. As collaborative filtering (CF) is one of the most prominent and popular techniques used for recommender systems, we propose a new *clustering-based* CF (CBCF) method using an incentivized/penalized user (IPU) model only with the ratings given by users, which is thus easy to implement. We aim to design such a simple clustering-based approach with no further prior information while improving the recommendation accuracy. To be precise, the purpose of CBCF with the IPU model is to improve recommendation performance such as *precision*, *recall*, and F_1 score by carefully exploiting different preferences among users. Specifically, we formulate a constrained optimization problem in which we aim to maximize the *recall* (or equivalently F_1 score) for a given *precision*. To this end, users are divided into several clusters based on the actual rating data and Pearson correlation coefficient. Afterward, we give each item an *incentive/penalty* according to the preference tendency by users within the same cluster. Our experimental results show a significant performance improvement over the baseline CF scheme without clustering in terms of *recall* or F_1 score for a given *precision*.

INDEX TERMS Clustering, collaborative filtering, F_1 score, incentivized/penalized user model, Pearson correlation coefficient, recommender system.

I. INTRODUCTION

People are likely to have an increasing difficulty in finding their favorite content effectively since extensive collections of video, audio, papers, art, etc. have been created both online and offline. For example, over hundreds of feature films and hundreds of thousands of books have been produced and published every year in the US. However, one person would read at most about 10,000 books in his/her life, and then he/she must choose his/her favorite books among them. On the one hand, recommender systems have been developed and used in diverse domains (e.g., the movie industry, the music industry, and so on) by helping people to select appropriate content based on individual preferences [1]. Especially, online commerce industries such as Amazon.com and Netflix have successfully exploited how to increase customer loyalty. For example, Amazon.com and Netflix have generated much of their sales by providing personalized items through their own recommender systems [2], [3].

While diverse recommender systems such as personalized recommendations, content-based recommendations, and knowledge-based recommendations have been developed, collaborative filtering (CF) is one of the most prominent and popular techniques used for recommender systems [4], [5]. CF methods are generally classified into memory-based CF and model-based CF. In model-based CF, training datasets are used to develop a model for predicting user preferences. Different machine learning techniques such as Bayesian networks, clustering, and rule-based approaches can also be utilized to build models. An alternating least squares with

The associate editor coordinating the review of this manuscript and approving it for publication was Zhong-Ke Gao.

weighted λ-regularization (ALS-WR) scheme is a representative example of model-based CF. ALS-WR is performed based on a matrix factorization algorithm and is tolerant of the data sparsity and scalability [6], [7]. The main advantages of model-based CF are an improvement of prediction performance and the robustness against the data sparsity. However, it has some shortcomings such as an expensive cost for building a model [5]. On the other hand, memory-based CF does not build a specific model, but directly computes the similarity between users or items using the entire rating matrix or its samples. Hence, memory-based CF is easy to implement and effective to manage. However, it has also some drawbacks such as dependence on human ratings, performance decrement when data are sparse, and disability of recommendation for new users (i.e., cold-start users) and items [5].

Memory-based CF approaches are again classified into user-based CF and item-based CF. The main ideas behind the user-based CF and item-based CF approaches are to find the user similarity and the item similarity, respectively, according to the ratings (or preferences). After finding similar users, called *neighbors*, user-based CF recommends the top-N most preferable items that an active user has not accessed yet. User-based CF has limitations related to scalability, especially when the number of users is much larger than the number of items. Item-based CF was proposed to mitigate this scalability problem, but cannot still entirely solve the problem when the numbers of users and items are large. Despite such limitations, CF has been employed as one of the most representative recommender systems leveraged in online commerce.

In addition, there have been many studies on the design of CF algorithms in terms of reducing the mean absolute error (MAE) or root mean squared error (RMSE) of rating prediction [8]. However, recommender systems designed in the sense of minimizing the MAE or RMSE do not inherently improve recommendation accuracy. We assume that there are two recommender systems having the same MAE or RMSE of the rating prediction. We note that they may differ from each other in terms of user experience (UX) since there is a possibility that one recommender system recommends an item whereas the other does not. For example, suppose that the real preference of a user on an item is 4.2 and two recommender systems predict the preference as 3.8 and 4.6, respectively. Then, when items having the predicted preference of more than 4.0 are assumed to be recommended, the MAEs of two recommender systems are the same but only the latter one will recommend the item. In order to redeem the above case, some performance metrics related to UX such as *precision*, *recall*, and F_1 score have been widely used in the literature.

On the other hand, several companies, e.g., Pandora Internet Radio, Netflix, and Artsy, have developed their own clustering-based recommendation methods, called Music Genome Project, Micro-Genres of Movies, and Art Genome Project, respectively. These clustering-based recommendation methods have successfully led to satisfactory performance, but the processing cost for clustering is very expensive. For example, it is widely known that each song tends to be analyzed by a musician through a process that takes usually 20 to 30 minutes per song in the case of Music Genome Project.

Unlike the aforementioned clustering-based recommendation methods that take long processing time to recommend items, we aim to design a simple but novel *clustering-based* CF (CBCF) method only with ratings given by users, which is thus easy to implement. That is, we design such a simple clustering-based approach with no further prior information while improving the recommendation accuracy. To this end, in this paper, we introduce the CBCF method using an incentivized/penalized user (IPU) model in improving the performance of recommender systems in terms of *precision*, *recall*, and F_1 score. More specifically, we present the CBCF method by carefully exploiting different preferences among users along with clustering. Our proposed method is built upon a predicted rating matrix-based clustering that can drastically reduce the processing overhead of clustering. In our CBCF method, we aim to select items to be recommended for users along with clustering. To this end, users are divided into several clusters based on the actual rating data and Pearson correlation coefficient. Then, items are regarded as more important or less important depending on the clusters that the users belong to. Afterwards, we give each item an *incentive/penalty* according to the preference tendency by users within the same cluster. The main contributions of our work are summarized as follows.

- An easy-to-implement CBCF method using the IPU model is proposed to further enhance the performance related to UX.
- To design our CBCF method, we first formulate a constrained optimization problem, in which we aim to maximize the *recall* (or equivalently F_1 score) for a given *precision*.
- We numerically find the amount of incentive/penalty that is to be given to each item according to the preference tendency by users within the same cluster.
- We evaluate the performance of the proposed method via extensive experiments and demonstrate that F_1 score of the CBCF method using the IPU model is improved compared with the baseline CF method without clustering, while *recall* for given (fixed) *precision* can be significantly improved by up to about 50%.

The remainder of this paper is organized as follows. Related work to our contributions is presented in Section II. Some backgrounds are presented in Section III. The overview of our proposed CBCF using the IPU model and the problem definition are described in Section IV. The implementation details of our CBCF method are shown in Section V. The datasets are described in Section VI, and the performance is analyzed via experiments in Section VII. Finally, we summarize our paper with some concluding remarks in Section VIII.

II. RELATED WORK

The method that we propose in this paper is related to four broader areas of research, namely CF approaches in recommender systems, various clustering methods, clustering-based recommender systems, and several studies on the recommender systems that analyzed the performance metrics such as *precision* and *recall*.

A. CF-AIDED RECOMMENDER SYSTEMS

CF is one of the most popular techniques used by recommender systems, but has some shortcomings vulnerable to data sparsity and cold-start problems [9]. If the data sparsity problem occurs with insufficient information about the ratings of users on items, then the values of predicted preference become inaccurate. Moreover, new users or items cannot be easily embedded in the CF process based on the rating information. There have been a plenty of challenges tackling these two problems [10], [11]. On the other hand, some of studies focused on how to improve prediction accuracy of CF-aided recommender systems [8], [12], [13]. In [12], [13], new similarity models were presented by using proximity impact popularity and Jaccard similarity measures, respectively. In [8], a typicality-based CF method, termed TyCo, was shown by taking into account typicality degrees. Recently, serendipitous CF-aided recommender systems received an attention, where surprising and interesting items are recommended to users [14]–[16].

B. CLUSTERING METHODS

Clustering has been widely used in diverse data mining applications: clustering algorithms such as k-Means and density-based spatial clustering of applications with noise (DBSCAN) were implemented in [17] to monitor game stickiness; a novel objective function based on the entropy was proposed in [18] to cluster different types of images; a cluster validity index based on a one-class classification method was presented in [19] by calculating a boundary radius of each cluster using kernel functions; a modified version of mean shift clustering for one-dimensional data was proposed in [20] to meet the real-time requirements in parallel processing systems; and a new criterion, called the cluster similar coefficient (CSC), was introduced in [21] to determine the suitable number of clusters, to analyze the non-fuzzy and fuzzy clusters, and to build clusters with a given CSC.

C. CLUSTERING-BASED RECOMMENDER SYSTEMS

There has been diverse research to enhance recommendation accuracy by means of clustering methods [22]–[25]. In [22], CF and content-based filtering methods were conducted by finding similar users and items, respectively, via clustering, and then personalized recommendation to the target user was made. As a result, improved performance on the *precision*, *recall*, and F_1 score was shown. Similarly as in [22], communities (or groups) were discovered in [23] before the application of matrix factorization to each community. In [24], social activeness and dynamic interest features were exploited to find similar communities by item grouping, where items are clustered into several groups using cosine similarity. As a result of grouping, the K most similar users based on the similarity measure were selected for recommendation. The performance of user-based CF with several clustering algorithms including K-Means, self-organizing maps (SOM), and *fuzzy* C*-Means (FCM)* clustering methods was shown in [25]. It was shown that user-based CF based on the FCM has the best performance in comparison with K-Means and SOM clustering methods. Moreover, several clustering approaches were studied in CF-aided recommender systems: heterogeneous evolutionary clustering was presented in [26] by dividing individuals with similar state values into the same cluster according to stable states; another dynamic evolutionary clustering was shown in [27] by computing user attribute distances; and more recently, dynamic evolutionary clustering based on time weight and latent attributes was proposed in [28].

D. PERFORMANCE ANALYSIS IN TERMS OF PRECISION AND RECALL

Performance metrics related to UX such as *precision*, *recall*, and F_1 score have been widely adopted for evaluating the accuracy of recommender systems [29]–[32]. In [30], time domain was exploited in designing CF algorithms by analyzing the inter-event time distribution of human behaviors when similarities between users or items are calculated. In addition, performance on the accuracy of other various recommender systems was analyzed in [29], [31], [32] with respect to *precision* and *recall*.

III. BACKGROUNDS

In this section, we summarize both preference prediction based on several CF algorithms and two clustering algorithms.

A. PREFERENCE PREDICTION METHODS

Preference prediction methods using CF are divided into memory-based and model-based approaches. Memory-based approaches directly utilize volumes of historical data to predict a rating on a target item and provide recommendations for active users. Whenever a recommendation task is performed, the memory-based approaches need to load all the data into the memory and implement specific algorithms on the data. On the other hand, model-based approaches leverage certain data mining methods to establish a prediction model based on the known data. Once a model is obtained, it does not need the raw data any more in the recommendation process [33].

In our work, we adopt memory-based approaches for our CBCF method. Although model-based approaches offer the benefits of prediction speed and scalability, they have some practical challenges such as inflexibility and quality of predictions. More specifically, building a model is often a time- and resource-consuming process; and the quality of predictions depends heavily on the way that a model is built.

1) USER/ITEM-BASED CF

There are two major memory-based CF algorithms, i.e., *user*-based and *item*-based algorithms. In user/item-based CF, we make a prediction for an active user, u, on a certain item i after finding similar users/items, respectively. Generally, in *user*-based CF, a correlation-based similarity is used for computing a user similarity and then a weighted sum of other users' ratings are used for making a prediction. In *item*-based CF, a cosine-based similarity and a simple weighted average can also be used for computing an item similarity and making a prediction, respectively. For more detailed process of both CF algorithms, we refer to [5].

B. CLUSTERING

Among various clustering methods such as SOM, K-Means, FCM, and spectral clusterings, we select spectral clustering and FCM, which have been widely known to ensure satisfactory performance. We briefly explain these two algorithms as follows.

Spectral clustering is based on the spectrum of an affinity matrix. In the affinity matrix, an affinity value between two objects (i.e., items) increases or decreases when the similarity between two objects is high or small, respectively. The Gaussian similarity function for quantifying the similarity between two objects is widely used to construct the affinity matrix.[1] After obtaining the affinity matrix, we find the corresponding eigenvectors/eigenvalues to group objects into several clusters. Finally, *spectral* clustering divides objects based on the eigenvectors/eigenvalues. There are various strategies for object division (refer to [34] for the details). While *spectral* clustering is simple to implement by a standard linear algebra software tool, it is known to significantly outperform traditional clustering algorithms such as K-Means clustering [34].

FCM clustering [35] allows each object to be the member of all clusters with different degrees of fuzzy membership by employing a coefficient w_{ij}^m that links an object x_i to a cluster c_j, where m is the hyper-parameter that controls how fuzzy the cluster will be. The higher m is, the fuzzier the cluster will be. FCM clustering first initializes coefficients of each point at random given a number of clusters. Then, the following two steps are repeated until the coefficients' change between two iterations is less than a given sensitivity threshold: 1) Computing the centroid for each cluster and 2) Recomputing coefficients of being in the clusters for each point.

IV. PROBLEM FORMULATION

In this section, we define and formulate our problem with a motivating example.

[1]The Gaussian similarity function is given by $s(x_i, x_j) = e^{\frac{-\|x_i-x_j\|^2}{2\sigma^2}}$, where σ controls the width of the neighborhoods [34].

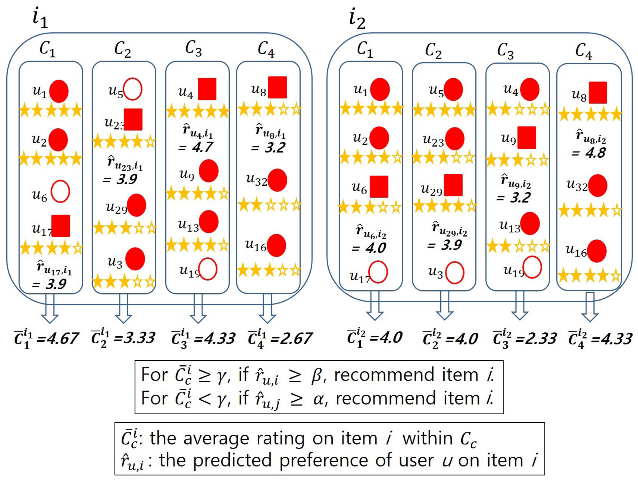


FIGURE 1. **An example of the proposed CBCF method with the IPU model, where two items and four clusters are assumed. Here, colored square items and colored circular items represent test data and training data, respectively.**

A. PROBLEM DEFINITION

The contribution of our work is to make a proper decision with which items should be recommended or not under the same MAE or RMSE in terms of improving UX (i.e., *recall* (or equivalently F_1 score) for a given *precision*). For example, suppose that there are two items with the same predicted preference value given by 3.9. If a recommender system only suggests items whose predicted preference is over 4.0, then above two items will be dropped by the system. However, there may be some users who are satisfied with the items, and thus UX will decrease in this case. In order to enhance the UX, we give each item an *incentive* or *penalty* according to the preference tendency by users. To this end, we *cluster* users into some groups and make a decision on which items are given the *incentive/penalty* based on a group that users belong to.

Fig. 1 shows an example of our proposed CBCF method with the IPU model, where two items and four clusters are assumed. Users are assumed to be grouped into four clusters, i.e., C_1, C_2, C_3, and C_4. From the figure, it can be seen that four users u_1, u_2, u_6, and u_{17} belong to cluster C_1. Here, colored square items and colored circular items represent test data and training data, respectively. We first denote $\hat{r}_{u,i}$ and $r_{u,i}$ as the *predicted* preference and *real* preference, respectively, of user u on item i, where memory-based and model-based CF approaches can be employed for rating prediction (refer to Section V for more details). Then as illustrated in Fig. 1, we have the *real* preference $r_{u_{17},i_1} = 4.0$ and its *predicted* preference $\hat{r}_{u_{17},i_1} = 3.9$. Items that each user u already rated along with the real preference are colored with red, whereas the others are not. For example, in cluster C_1, i_1 was rated as 5.0, 5.0, and 4.0 stars by users u_1, u_2, and u_{17}, respectively, thus resulting in $r_{u_1,i_1} = 5.0$, $r_{u_2,i_1} = 5.0$, and $r_{u_{17},i_1} = 4.0$. In the same cluster, users u_1, u_2, and u_6 rated i_2 as 5, 4, and 3 stars, respectively, resulting in $r_{u_1,i_2} = 5.0$,

Algorithm 1 Proposed CBCF Using the IPU Model

```
if C̄_c^i ≥ γ then
    if r̂_{u,i} ≥ β then
        Recommend item i to user u
    else Drop item i;
else
    if r̂_{u,i} ≥ α then
        Recommend item i to user u
    else Drop item i;
end
```

$r_{u_2,i_2} = 4.0$, and $r_{u_6,i_2} = 3.0$. Let us now denote $\bar{C}_c^i$ as the average preference on item i of users within cluster C_c. More specifically, $\bar{C}_c^i$ can be expressed as

$$\bar{C}_c^i = \frac{\sum_{u \in U_{i,c}} r_{u,i}}{|U_{i,c}|}, \tag{1}$$

where $U_{i,c}$ is the set of users who rated item i within cluster C_c and $|\cdot|$ is the cardinality of a set. Then, as shown in Fig. 1, the average preference $\bar{C}_1^{i_1}$ of item i_1 rated by users within C_1 is given by 4.67. Similarly, $\bar{C}_2^{i_1}$ is given by 3.33.

Based on the values of $\bar{C}_c^i$ in each cluster, we decide which items should be recommended or not for user u according to the following recommendation strategy using the IPU model. When the value of $\bar{C}_c^i$ is sufficiently large, i.e., $\bar{C}_c^i \geq \gamma$, item i is given an incentive, where $\gamma > 0$ indicates a system parameter that is to be optimized later. Otherwise (i.e., if $\bar{C}_c^i < \gamma$), item i having small $\bar{C}_c^i$ gets a penalty. System parameters α and β are used as thresholds for giving a penalty and an incentive, respectively, in our method and are set to certain positive values, where $\alpha \geq \beta$. For example, suppose that $\alpha = 4.5$, $\beta = 3.5$, and $\gamma = 3.0$. Then, in Fig. 1, i_1 will be recommended to u_{19} but i_2 will not be recommended to u_{19} if the predicted preferences of i_1 and i_2 (i.e., $\hat{r}_{u_{19},i_1}$ and $\hat{r}_{u_{19},i_2}$) are 3.8 and 4.2, respectively. This is because $\bar{C}_3^{i_1}$ $(= 4.33)$ is larger than γ $(= 3.0)$ and $\hat{r}_{u_{19},i_1}$ $(= 3.8)$ is also larger than β $(= 3.5)$. In the case of i_2, however, u_{19} does not receive recommendation since $\bar{C}_3^{i_2}$ $(= 2.33)$ is smaller than γ as well as $\hat{r}_{u_{19},i_2} < \alpha$. In short, a decision on recommendation can be changed depending on the preference tendency of each user obtained from clustering.

Algorithm 1 describes our CBCF method using the IPU model. From Algorithm 1, it is observed that items rated over β are just recommended when $\bar{C}_c^i \geq \gamma$. If $\bar{C}_c^i < \gamma$, then only items whose predicted preference is larger than α are recommended.

As mentioned before, we use the *precision*, *recall*, and F_1 score for performance evaluation. These three performance metrics can be expressed as functions of true positive (*tp*), true negative (*tn*), false positive (*fp*), and false negative (*fn*). Assume that we predict a *condition* as true. If the *condition* is actually true (or false), then it is *tp* (or *fp*). If a *condition* is predicted as false and the *condition* is actually true (or false), then it is *fn* (or *tn*).

For given user u and item i, the terms *tp*, *tn*, *fp*, and *fn* are dependent on α, β, and γ, and thus are given by

$$\begin{aligned}
&f_{tp}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}}) \\
&\quad = I_{[\gamma,\infty)}(\bar{C}_c^{u,i}) \cdot I_{[\beta,\infty)}(\hat{r}_{u,i}) \cdot I_{[\delta_{\text{pref}},\infty)}(r_{u,i}) \\
&\qquad + I_{(0,\gamma)}(\bar{C}_c^{u,i}) \cdot I_{[\alpha,\infty)}(\hat{r}_{u,i}) \cdot I_{[\delta_{\text{pref}},\infty)}(r_{u,i}), \\
&f_{fp}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}}) \\
&\quad = I_{[\gamma,\infty)}(\bar{C}_c^{u,i}) \cdot I_{[\beta,\infty)}(\hat{r}_{u,i}) \cdot I_{(0,\delta_{\text{pref}})}(r_{u,i}) \\
&\qquad + I_{(0,\gamma)}(\bar{C}_c^{u,i}) \cdot I_{[\alpha,\infty)}(\hat{r}_{u,i}) \cdot I_{(0,\delta_{\text{pref}})}(r_{u,i}), \\
&f_{fn}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}}) \\
&\quad = I_{[\gamma,\infty)}(\bar{C}_c^{u,i}) \cdot I_{(0,\beta)}(\hat{r}_{u,i}) \cdot I_{[\delta_{\text{pref}},\infty)}(r_{u,i}) \\
&\qquad + I_{(0,\gamma)}(\bar{C}_c^{u,i}) \cdot I_{(0,\alpha)}(\hat{r}_{u,i}) \cdot I_{[\delta_{\text{pref}},\infty)}(r_{u,i}), \\
&f_{tn}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}}) \\
&\quad = I_{[\gamma,\infty)}(\bar{C}_c^{u,i}) \cdot I_{(0,\beta)}(\hat{r}_{u,i}) \cdot I_{(0,\delta_{\text{pref}})}(r_{u,i}) \\
&\qquad + I_{(0,\gamma)}(\bar{C}_c^{u,i}) \cdot I_{(0,\alpha)}(\hat{r}_{u,i}) \cdot I_{(0,\delta_{\text{pref}})}(r_{u,i}),
\end{aligned} \tag{2}$$

respectively, where $I_A(x)$ is the indicator function of set A and δ_{pref} is a threshold value for determining whether a user really satisfies with the corresponding item.[2] Then, it follows that $f_{tp}^{u,i} = 1$ if $\bar{C}_c^{u,i} \geq \gamma$, $\hat{r}_{u,i} \geq \beta$, and $r_{u,i} \geq \delta_{\text{pref}}$; $f_{tp}^{u,i} = 1$ if $\bar{C}_c^{u,i} < \gamma$, $\hat{r}_{u,i} \geq \alpha$, and $r_{u,i} \geq \delta_{\text{pref}}$; and $f_{tp}^{u,i} = 0$ otherwise. In a similar fashion, $f_{fp}^{u,i} = 1$ if $\bar{C}_c^{u,i} \geq \gamma$, $\hat{r}_{u,i} \geq \beta$, and $r_{u,i} < \delta_{\text{pref}}$; $f_{fp}^{u,i} = 1$ if $\bar{C}_c^{u,i} < \gamma$, $\hat{r}_{u,i} \geq \alpha$, and $r_{u,i} < \delta_{\text{pref}}$; and $f_{fp}^{u,i} = 0$ otherwise. Moreover, $f_{fn}^{u,i} = 1$ if $\bar{C}_c^{u,i} \geq \gamma$, $\hat{r}_{u,i} < \beta$, and $r_{u,i} \geq \delta_{\text{pref}}$; $f_{fn}^{u,i} = 1$ if $\bar{C}_c^{u,i} < \gamma$, $\hat{r}_{u,i} < \alpha$, and $r_{u,i} \geq \delta_{\text{pref}}$; and $f_{fn}^{u,i} = 0$ otherwise. Finally, $f_{tn}^{u,i}$ is also counted similarly as above, but it is not used for computing the *precision*, *recall*, and F_1 score.

Based on (2), the *precision* and *recall* are given by[3]

$$\begin{aligned}
&\mathit{precision}(\alpha,\beta,\gamma,\delta_{\text{pref}}) \\
&= \frac{\sum_{(u,i)\in \mathrm{T}} f_{tp}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}})}{\sum_{(u,i)\in \mathrm{T}} f_{tp}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}}) + \sum_{(u,i)\in \mathrm{T}} f_{fp}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}})} \\
&\mathit{recall}(\alpha,\beta,\gamma,\delta_{\text{pref}}) \\
&= \frac{\sum_{(u,i)\in \mathrm{T}} f_{tp}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}})}{\sum_{(u,i)\in \mathrm{T}} f_{tp}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}}) + \sum_{(u,i)\in \mathrm{T}} f_{fn}^{u,i}(\alpha,\beta,\gamma,\delta_{\text{pref}})},
\end{aligned} \tag{3}$$

where T represents the set of test data used for measuring *precision* and *recall*. Due to the fact that the F_1 score is the harmonic mean of *precision* and *recall*, it is defined as

$$F_1(\alpha,\beta,\gamma,\delta_{\text{pref}}) = \frac{2\mathit{precision} \times \mathit{recall}}{\mathit{precision} + \mathit{recall}}. \tag{4}$$

Let us recall the example in Fig. 1, where $\alpha = 4.5$, $\beta = 3.5$, and $\gamma = 3.0$. Square items representing the test data

[2] Note that δ_{pref} is generally set to 4.0 (or 8.0) in case of a five-point scale (or a ten-point scale).

[3] To simplify notations, $\mathit{precision}(\alpha,\beta,\gamma,\delta_{\text{pref}})$ and $\mathit{precision}(\alpha,\beta,\gamma,\delta_{\text{pref}})$ will be written as *precision* and *recall*, respectively, if dropping the arguments α, β, γ, and δ_{pref} does not cause any confusion.

TABLE 1. An example of *tp*, *fn*, *fp*, and *tn* when $\gamma = 0$ and $\gamma = 3$.

		item i_1	item i_2
$\gamma = 0$ (baseline)	Recommended items	$u_4 \Rightarrow tp$	$u_6 \Rightarrow fp$ $u_8 \Rightarrow tp$
	Non-recommended items	$u_{17} \Rightarrow fn$ $u_{23} \Rightarrow fn$ $u_8 \Rightarrow tn$	$u_9 \Rightarrow tn$ $u_{29} \Rightarrow fn$
$\gamma = 3.0$ (proposed)	Recommended items	$u_{17} \Rightarrow tp$ $u_4 \Rightarrow tp$	$u_6 \Rightarrow fp$ $u_{29} \Rightarrow tp$ $u_8 \Rightarrow tp$
	Non-recommended items	$u_{23} \Rightarrow fn$ $u_8 \Rightarrow tn$	$u_9 \Rightarrow tn$

are used for performance analysis. Suppose that items rated over 4 stars are satisfactory for users, i.e., $\delta_{\text{pref}} = 4.0$, which is a typical assumption in recommender systems [36]. Then, user u_{17} should receive recommendation for item i_1, whereas user u_8 should not. Users u_{29} and u_8 are actually satisfied with item i_2. Based on the test dataset in Fig. 1, the terms *tp*, *tn*, *fp*, and *fn* are summarized in Table 1. For comparison, let us consider a baseline scenario where clustering is not exploited. To this end, we assume $\gamma = 0$ and modify the recommendation strategy so that item i is recommended only if the predicted preference $\hat{r}_u^i$ is no less than 4.0. In this case, the four terms *tp*, *fn*, *fp*, and *tn* are also depicted in Table 1. Using the result of Table 1, we are ready to compute the *precision* and *recall* for the two cases, i.e., $\gamma = 0$ and $\gamma = 3.0$, as follows.

- $\gamma = 0$ (baseline): From Table 1, it follows that $tp = 2$, $fp = 1$, and $fn = 3$. Thus, using (2), we have $precision = 2/3$ and $recall = 2/5$.
- $\gamma = 3.0$ (proposed): Suppose that $\alpha = 4.5$ and $\beta = 3.5$. From Table 1 and (2), it follows that $tp = 4$, $fp = 1$, and $fn = 1$. Hence, we have $precision = 4/5$ and $recall = 4/5$.

Consequently, performance on the *precision* and *recall* can be improved by properly adjusting the system parameters α, β, and γ under our IPU model when items are grouped into multiple clusters.

B. FORMULATION

It is worth noting that the *precision*, *recall*, and F_1 score vary significantly according to the change of α, β, and γ. For this reason, we aim at finding the optimal α, β, and γ such that the F_1 score (or *recall*) is maximized. We thus formulate a new constrained optimization problem as follows[4]:

$$\begin{aligned} \underset{\alpha,\beta,\gamma}{\text{maximize}} \quad & F_1(\alpha, \beta, \gamma) \text{ or } recall(\alpha, \beta, \gamma) \\ \text{subject to} \quad & precision(\alpha, \beta, \gamma) \geq \delta_{\text{precision}} \\ & \alpha \geq \beta, \end{aligned} \tag{5}$$

[4]Since the parameter δ_{pref} is generally set to a certain value, δ_{pref} will be dropped from the argument of each function to simplify notations if dropping it does not cause any confusion.

where $\delta_{\text{precision}}$ is a pre-defined threshold value for *precision* and is set to a certain value appropriately according to various types of recommender systems. Equation (5) can be also easily modified for different purposes. For example, we can find the optimal α, β, and γ such that $precision(\alpha, \beta, \gamma)$ is maximized under $recall(\alpha, \beta, \gamma) \geq \delta_{\text{recall}}$ or $recall(\alpha, \beta, \gamma)$ is maximized under $precision(\alpha, \beta, \gamma) \geq \delta_{\text{precision}}$, where δ_{recall} is a pre-defined threshold value for *recall*. Hence, the *precision*, *recall*, and F_1 score can be improved by not only clustering items but also optimally finding parameters α, β, and γ in our CBCF method using the IPU model.

V. PROPOSED METHOD

The CBCF method recommends desirable items according to the result of item clustering and the preference tendency of each user using our IPU model.

The main contribution of our CBCF method using the IPU model is to give either an incentive or a penalty to each item based on $\bar{C}_c^i$ (the average preference on item i of users within cluster C_c), which depends on the result of clustering. As mentioned before, since there are empty elements in the rating matrix $\mathbf{R}_{CBCF}$ that users have not rated or accessed yet, the Euclidian distance between user vectors (i.e., row vectors in $\mathbf{R}_{CBCF}$) cannot be accurately calculated. Hence, we use the Pearson correlation coefficient (PCC) in our work. PCC computes the correlation between two users' common ratings to measure their similarity, and thus needs two common ratings at least. PCC between two users, u_1 and u_2, is calculated as

$$s(u_1, u_2) = \frac{\sum_{i \in I_{u_1} \cap I_{u_2}} (r_{u_1,i} - \bar{r}_{u_1}) \cdot (r_{u_2,i} - \bar{r}_{u_2})}{\sqrt{\sum_{i \in I_{u_1} \cap I_{u_2}} (r_{u_1,i} - \bar{r}_{u_1})^2} \cdot \sqrt{\sum_{i \in I_{u_1} \cap I_{u_2}} (r_{u_2,i} - \bar{r}_{u_2})^2}}, \tag{6}$$

where I_{u_1} and I_{u_2} are the item sets rated by u_1 and u_2, respectively, and $\bar{r}_{u_1}$ and $\bar{r}_{u_2}$ are the mean values of their ratings over the item set $I_{u_1} \cap I_{u_2}$ that two users have commonly rated, respectively. Here, $s(u_1, u_2)$ ranges from -1 to 1. A correlation coefficient close to -1 indicates a negative linear relationship, and $s(u_1, u_2)$ of 1 indicates a perfect positive linear relationship.

Let us turn our attention to the description of our CBCF method in Algorithm 2. First, the set of clusters, C, is obtained by the result of clustering where c groups are generated, and an $n \times m$ rating matrix $\mathbf{R}_{CBCF}$ is initialized (refer to lines 1–2 in Algorithm 2). In the next step, we use a preference prediction method based on memory-based approaches along with $\mathbf{R}_{CBCF}$ and the resulting output is stored in $\hat{R}$ (refer to line 3). More specifically, user/item-based CF algorithms are used to evaluate the performance of our proposed CBCF method. The threshold values α, β, and γ can be determined by solving the optimization problem in (5) via exhaustive search. In the *for* loop, the set I_u is the items of missing ratings in the test set for each user u and the predicted ratings in I_u are assigned to $\hat{r}_{u,I_u}$, where $|I_u|$ denotes the cardinality of the set I_u. Now, we decide which items are recom-

Algorithm 2 CBCF Using the IPU Model

1 Clusters $C \in \{C_1, \cdots, C_c\}$;
2 Initialize the $n \times m$ rating matrix $\mathbf{R}_{CBCF}$;
3 $\hat{R} \leftarrow$ a function of rating prediction with $\mathbf{R}_{CBCF}$;
4 Initialize the threshold values α, β, and γ;
5 **for** $u \leftarrow 1$ **to** n **do**
6     $I_u \leftarrow$ items of missing ratings in the test set for user u;
7     $\hat{r}_{u,I_u} \leftarrow$ predicted rating values of I_u;
8   **for** $i \leftarrow 1$ **to** $|I_u|$ **do**
9         $C_{tmp} \leftarrow$ a cluster to which user u belongs;
10         $\bar{C}^i_{tmp} \leftarrow$ average rating on item i in C_{tmp};
11   **if** $\hat{r}_{u,i} \geq \alpha$ **then**
12   Recommend item i to user u
13   **else if** $\hat{r}_{u,i} \geq \beta$ && $\bar{C}^i_{tmp} \geq \gamma$ **then**
14   Recommend item i to user u
15   **else** Drop item i;
16   **end**
17 **end**

mended or dropped for given α, β, and γ. When $\hat{r}_{u,i} \geq \alpha$, the item i is recommended to user u regardless of the value of γ as mentioned in Algorithm 1 (refer to lines 11–12 in Algorithm 2). However, when $\hat{r}_{u,i} < \alpha$, we have to check the value of threshold γ, which is to be compared with the average preference on a certain item of users in a cluster, denoted by $\bar{C}^i_{tmp}$. When $\bar{C}^i_{tmp} < \gamma$, the item i will not be recommended even if $\beta \leq \hat{r}_{u,i} < \alpha$. This is because we give a penalty to the item i for $\bar{C}^i_{tmp} < \gamma$. On the other hand, when $\hat{r}_{u,i} > \beta$ and $\bar{C}^i_{tmp} \geq \gamma$, the item i will be recommended to user u (refer to lines 13–14). The item i will be always dropped when $\hat{r}_{u,i} < \beta$ (refer to line 15).

Finally, we find α, β, and γ fulfilling (5). Algorithm 2 is performed iteratively while varying the values of α, β, and γ. That is, lines 4–17 in Algorithm 2 are iteratively executed by numerically optimizing α, β, and γ according to (5).

The CBCF method using the IPU model is summarized as follows:

- Suppose that the CBCF method decides whether a certain item (i) is recommended to an active user (u) or not under the IPC model based on clustering.
- If the predicted preference is sufficiently large (i.e., $\hat{r}_{u,i} \geq \alpha$), then the item i is recommended to the user u.
- If the predicted preference is not sufficiently large but the two conditions, i.e., $\hat{r}_{u,i} \geq \beta$ and $\bar{C}^i_c \geq \gamma$, are met, then the item i is recommended to the user u, where $\bar{C}^i_c$ is the average preference on item i of users within cluster C_c.

VI. DATASET AND DATABASE STRUCTURE

In this section, we describe our dataset and database (DB) structure. CBCF is utilized for non-cold-start users, but it will be empirically shown in Section VII how it is robust to more difficult situations including cold-start users.[5] We use the MovieLens 100K dataset[6] with the following attributes:

- 100K dataset have 100,000 anonymous ratings
- Ratings are made on a 5-star scale
- There are 943 users in 100K dataset
- There are 1,682 movies in 100K dataset
- Each user has at least 20 ratings.

Note that the sparsity (i.e., the ratio of the number of missing cells in a rating matrix to the total number of cells) of the rating matrix obtained from the MovieLens 100K dataset is 93.7%, which is high and often causes performance degradation. One popular solution to the data sparsity problem is the use of data imputation [38]–[40], which includes the zero injection method [38] in which zeros are given to some missing cells in a rating matrix and two matrix factorization-based methods [39], [40] that assign zeros or twos to all missing cells in a rating matrix. Even if such data imputation techniques are known to significantly improve the prediction accuracy, we do not employ them in our experiments since solving the data sparsity problem is not our primary focus. The DB structure for CBCF is described as follows.

Assume that there are a set of users, U, and a set of items, I, in a recommender system as follows:

$$U \triangleq \{u_1, u_2, \cdots, u_n\},$$
$$I \triangleq \{i_1, i_2, \cdots, i_m\}, \tag{7}$$

where n and m represent the number of users and the number of items, respectively. Then, in the CBCF process, the rating matrix $\mathbf{R}_{CBCF}$ is defined as

$$\mathbf{R}_{CBCF} = \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} & \dots & r_{1,m} \\ r_{2,1} & r_{2,2} & r_{2,3} & \dots & r_{2,m} \\ r_{3,1} & r_{3,2} & r_{3,3} & \dots & r_{3,m} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ r_{n,1} & r_{n,2} & r_{n,3} & \dots & r_{n,m} \end{pmatrix}, \tag{8}$$

where $r_{u,i}$ is the rating of user u on item i for $u \in \{1, \cdots, n\}$ and $i \in \{1, \cdots, m\}$. Note that $\mathbf{R}_{CBCF}$ can be either the users' explicit ratings or the users' implicit preferences. If user u has not rated or accessed item i yet, then $r_{u,i}$ remains empty.

The user set U is grouped into several clusters and a *user cluster* is a set of similar users in the rating matrix $\mathbf{R}_{CBCF}$. In order to cluster U, we define n user vectors, each of which consists of m elements, which are given by

$$\mathbf{U}_b = [r_{b,1}, r_{b,2}, \cdots, r_{b,m}] \tag{9}$$

for $b \in \{1, \cdots, n\}$. Suppose that n user vectors are clustered into c user groups,[7] where the set of clusters, C, is denoted by

$$C = \{C_1, C_2, \cdots, C_c\}. \tag{10}$$

[5]In this paper, a cold-start user is defined as the user who does not have enough rating information. More than 20 ratings for each user are usually known as enough information [37].

[6]http://grouplens.org/datasets/movielens/.

[7]For clustering, it is of importance how to determine the number of clusters. This is heavily dependent on the characteristics of recommender systems and thus is beyond the scope of this paper.

TABLE 2. DB structure of CBCF.

User ID	Item ID	Ratings ($\mathbf{R}_{CBCF}$)
u_1	i_1	$r_{1,1}$
u_1	i_2	$r_{1,2}$
u_1	i_8	$r_{1,8}$
$\vdots$	$\vdots$	$\vdots$
u_n	i_{m-4}	$r_{n,m-4}$
u_n	i_m	$r_{n,m}$

In this case, one can say that the users within a cluster are relatively closer than other users not in the same cluster from the viewpoint of users' preferences. For example, assume that there are four user vectors given by $\mathbf{U}_1 = [2, 0, 1, 0]$, $\mathbf{U}_2 = [0, 4, 0, 2]$, $\mathbf{U}_3 = [3, 0, 2, 0]$, and $\mathbf{U}_4 = [0, 3, 0, 2]$. Let us divide the four vectors into two clusters. Then, $\mathbf{U}_1$ and $\mathbf{U}_3$ will be grouped into one cluster and are considered as similar users by the users' ratings because the Euclidian distance between $(\mathbf{U}_1, \mathbf{U}_3)$ is closer than that made from other combinations including $(\mathbf{U}_1, \mathbf{U}_2)$, $(\mathbf{U}_1, \mathbf{U}_4)$, $(\mathbf{U}_3, \mathbf{U}_2)$, and $(\mathbf{U}_3, \mathbf{U}_4)$.

The DB structure for CBCF is shown in Table 2. The DB consists of the following three fields: user ID, item ID, and ratings. For example, if item i_1 was enjoyed by user u_1 and was rated as 4.0, then a new tuple '$u_1|i_1|4.0$' will be inserted into the DB.

VII. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of our proposed CBCF method using the IPU model in terms of *precision*, *recall*, and F_1 score. In our experiments, unless otherwise stated, item-based CF is adopted in our proposed method since it shows better performance on the accuracy of recommendation for memory-based CF, which will be verified later in this section. We use Apache Mahout[8] whose goal is to build an environment for performing downstream machine learning tasks such as CF, clustering, and classification. It is assumed that the recommendation result is true when the following conditions are met:

- The real rating of an item recommended to a user is 4.0 or 5.0.
- The real rating of an item not recommended to a user is less than 4.0.

In our experiments, the number of clusters for both *spectral* and FCM clustering algorithms is set to $c = 10$; the fuzzy degree m of FCM clustering is set to 2 according to [41]; and the convergence threshold of FCM clustering is set to 10^{-4}. In the FCM clustering, an object is assigned to such a cluster that has the highest coefficient. In our subsequent experiments, we adopt *spectral* clustering by default unless otherwise stated. Fig. 2 compares the inter-cluster Euclidean distances with the intra-cluster Euclidean distances in order to show the validity of clustering. The values of PCC range between -1.0 and 1.0, where 1.0 and -1.0 imply that two objects (e.g., users) have the highest positive and negative correlations, respectively. Hence, since most clustering algorithms do not employ any negative correlation, the value of PCC between two users u_1 and u_2, namely $s(u_1, u_2)$, is shifted as follows:

$$s(u_1, u_2) \leftarrow 1 - s(u_1, u_2) \text{ for } s(u_1, u_2) \in [0, 1]$$
$$s(u_1, u_2) \leftarrow -(s(u_1, u_2) - 1) \text{ for } s(u_1, u_2) \in [-1, 0). \quad (11)$$

Then, a value close to 0 indicates a highly positive correlation while a value close to 2 corresponds to a highly negative correlation. As shown in Fig. 2, it is observed that the intra-cluster distance is smaller than the inter-cluster distances from the perspective of cluster 0. It thus reveals that our PCC-based clustering works appropriately.

Fig. 3 shows the effect of α and β, which correspond to thresholds for giving a penalty and an incentive, respectively, on the F_1 score when another threshold γ is set to 3.4. We note that the proposed CBCF method using the IPU model has the maximum F_1 score ($= 0.7451$) when $\alpha = 3.7$ and $\beta = 2.9$. It is observed that the F_1 score decreases as α and β increase since the decreasing rate of *recall* is larger than the increasing rate of *precision* with increasing α and β. More specifically,

[8]http://mahout.apache.org/.

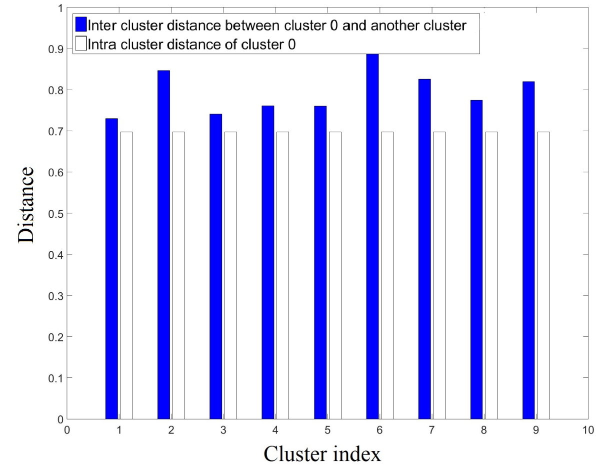


FIGURE 2. Comparison of the inter-cluster and intra-cluster Euclidean distances.

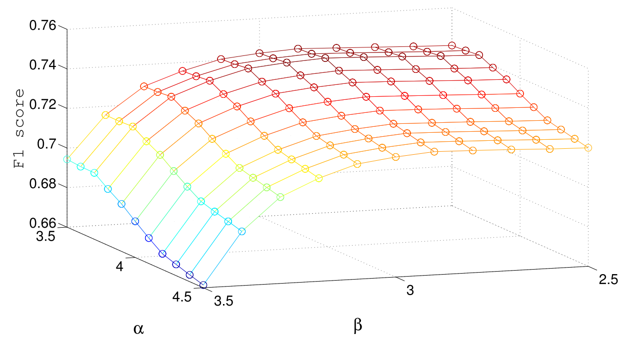


FIGURE 3. F_1 score over α and β when $\gamma = 3.4$.

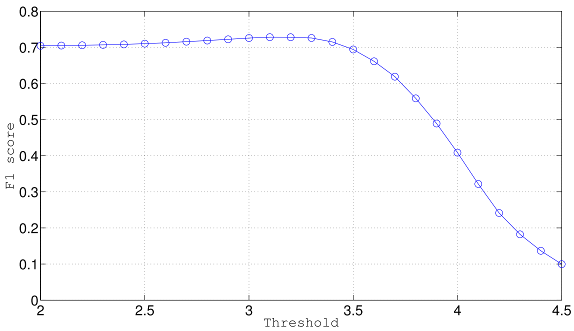


FIGURE 4. F_1 **score over the recommendation threshold when item-based CF is adopted.**

TABLE 3. **The maximum *recall* and** F_1 **score for given *precision*.**

precision	Baseline method		Proposed method	
	recall	F_1 score	*recall*	F_1 score
0.7449	0.2815	0.4085	0.4343	0.5487
0.7201	0.4565	0.5588	0.5706	0.6367
0.7074	0.5499	0.6188	0.6842	0.6956
0.6519	0.7914	0.7149	0.825	0.7283
0.6036	0.9177	0.7282	0.9402	0.7352

if both α and β are large, then *precision* and *recall* tend to increase and decrease, respectively, since fewer items are recommended. However, due to the fact that the decrement of *recall* is faster than the increment of *precision*, the F_1 score gets reduced accordingly. For example, in Fig. 3, it is seen that *precision* $= 0.6595$ and *recall* $= 0.8564$ when $\alpha = 3.7$, $\beta = 2.9$, and $\gamma = 3.4$, while *precision* $= 0.6853$ and *recall* $= 0.076$ when $\alpha = 4.4$, $\beta = 4.4$, and $\gamma = 3.4$.

Fig. 4 shows the F_1 score over the recommendation threshold when the baseline item-based CF method without clustering (i.e., $\gamma = 0$) is adopted. In this baseline approach, if the predicted rating of a certain item is larger than the recommendation threshold, then the corresponding item is recommended to a user. If the real rating is over 4.0, then the recommendation is regarded as valid. As shown in this figure, the maximum of F_1 score is 0.7282 when the threshold value is given by 3.1. It is shown that the overall tendency is similar to that in Fig. 3, but the F_1 score of the proposed method is increased by about 3% compared to this baseline approach employing item-based CF.

Table 3 shows the *recall* and F_1 score for given *precision* when the proposed CBCF using the PIU model and the baseline method without clustering are used. In the baseline item-based CF method, when the recommendation threshold is set to 4.0, the value of *precision* is 0.7449 and the corresponding maximum *recall* is 0.2815. On the other hand, in the proposed method, when $\alpha = 3.9$, $\beta = 2.1$, and $\gamma = 4.2$, the maximum value of *recall* is 0.4343. This improvement is nearly 50%. That is, the proposed method has a remarkably higher *recall* value compared to the baseline under the same *precision* as depicted in Table 3. From Figs. 3 and 4, and Table 3, it is shown that the proposed method can achieve

TABLE 4. **Performance of the proposed method based on item-based CF, where both *spectral* and FCM clustering algorithms are employed.**

Clustering	γ	α	β	*precision*	*recall*	F_1 score
Spectral	3.4	3.7	2.9	0.6595	0.8564	0.7451
FCM	3.5	3.3	2.5	0.6625	0.8639	**0.7499**

TABLE 5. **Performance of the proposed method based on user-based CF, where both *spectral* and FCM clustering algorithms are employed.**

Clustering	γ	α	β	*precision*	*recall*	F_1 score
Spectral	3.5	3.1	2.7	0.6309	0.8893	0.7382
FCM	3.3	3.7	2.9	0.6448	0.8730	**0.7418**

TABLE 6. **Performance of the proposed and baseline methods for cold-start users.**

Method	*precision*	*recall*	F_1 score
Baseline method	0.7085	0.3552	0.4732
Proposed method	0.6793	0.6934	**0.6863**

a great improvement with respect to *recall* or F_1 score for given *precision*.

Generally, a small recommendation threshold value leads a low *precision* and high *recall*, and vice versa. However, as mentioned before, as the threshold value becomes very large, the F_1 score is rapidly decreased because the decreasing rate of *recall* is faster than the increasing rate of *precision*.

Instead of item-based CF, user-based CF can also be employed in our proposed CBCF method. When the parameters α, β, and γ are optimally found via exhaustive search in the sense of maximizing the F_1 score, we evaluate the performance of our proposed CBCF method using the IPU model based on item-based CF and user-based CF methods in Tables 4 and 5, respectively, where both *spectral* and FCM clustering algorithms are employed for non-cold-start users. Based on the results, the following observations are made: i) the proposed method based on item-based CF achieves better performance on the F_1 score than the case of user-based CF and ii) using the proposed method based on FCM clustering is slightly superior to the case of *spectral* clustering.

Moreover, we evaluate the performance of the proposed and baseline methods for more difficult situations having cold-start users whose number of rated items is less than 20, where item-based CF and *spectral* clustering are used. Due to the fact that the MovieLens 100K dataset does not contain records for cold-start users, we modify the experimental setup according to [42]. Specifically, we first select users who have rated between 20-30 items as the testing set, consisting of 290 users, and make the number of rated items of each selected user in the range between 3 and 20 via random masking. The remaining 653 users from the original dataset is used as the training set. The results in Table 6 follow similar trends to those for non-cold-start users while the CBCF method provides gains over the baseline without clustering, where the three threshold values are optimally found in the sense of maximizing the F_1 score.

VIII. CONCLUDING REMARKS

In this paper, we proposed a CBCF method using the IPU model in recommender systems by carefully exploiting different preferences among users along with clustering. Specifically, in the proposed CBCF method, we formulated a constrained optimization problem in terms of maximizing the *recall* (or equivalently F_1 score) for a given *precision*. To this end, clustering was applied so that not only users are divided into several clusters based on the actual rating data and Pearson correlation coefficient but also an incentive/penalty is given to each item according to the preference tendency by users within a same cluster. As a main result, it was demonstrated that the proposed CBCF method using the IPU model brings a remarkable gain in terms of *recall* or F_1 score for a given *precision*.

A possible direction of future research in this area includes the design of a new clustering-based CF method by exploiting the properties of model-based CF approaches (e.g., matrix factorization).

ACKNOWLEDGEMENT

The authors would like to thank Younghyun Kim for providing his initial conception of modeling.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 6, pp. 734–749, Jun. 2005.
[2] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan./Feb. 2003.
[3] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, Aug. 2009.
[4] J. A. Konstan and J. Riedl, "Recommender systems: From algorithms to user experience," *User Model. User-Adapted Interact.*, vol. 22, nos. 1–2, pp. 101–123, Apr. 2012.
[5] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Adv. Artif. Intell.*, vol. 2009, pp. 1–19, Jan. 2009, Art. no. 421425.
[6] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, "Large-scale parallel collaborative filtering for the Netflix prize," in *Proc. Int. Conf. Algorithmic Appl. Manage.*, Shanghai, China, Jun. 2008, pp. 337–348.
[7] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proc. 8th IEEE Int. Conf. Data Mining*, Pisa, Italy, Dec. 2008, pp. 263–272.
[8] Y. Cai, H.-F. Leung, Q. Li, H. Min, J. Tang, and J. Li, "Typicality-based collaborative filtering recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 3, pp. 766–779, Mar. 2014.
[9] G. Guo, J. Zhang, and D. Thalmann, "Merging trust in collaborative filtering to alleviate data sparsity and cold start," *Knowl.-Based Syst.*, vol. 57, pp. 57–68, Feb. 2014.
[10] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, "A collaborative filtering approach to mitigate the new user cold start problem," *Knowl.-Based Syst.*, vol. 26, pp. 225–238, Feb. 2012.
[11] H. Sobhanam and A. K. Mariappan, "A hybrid approach to solve cold start problem in recommender systems using association rules and clustering technique," *Int. J. Comput. Appl.*, vol. 74, no. 4, pp. 17–23, Jul. 2013.
[12] H. Liu, Z. Hu, A. Mian, H. Tian, and X. Zhu, "A new user similarity model to improve the accuracy of collaborative filtering," *Knowl.-Based Syst.*, vol. 56, pp. 156–166, Jan. 2014.
[13] B.-H. Huang and B.-R. Dai, "A weighted distance similarity model to improve the accuracy of collaborative recommender system," in *Proc. 16th IEEE Int. Conf. Mobile Data Manage.*, Pittsburgh, PA, USA, Jun. 2015, pp. 104–109.
[14] Q. Lu, T. Chen, W. Zhang, D. Yang, and Y. Yu, "Serendipitous personalized ranking for top-N recommendation," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol.*, Macau, China, Dec. 2012, pp. 258–265.
[15] K. Oku and F. Hattori, "Fusion-based recommender system for improving serendipity," in *Proc. ACM Workshop Novelty Diversity Rec. Syst. (DiveRS)*, Chicago, IL, USA, Oct. 2011, pp. 19–26.
[16] P. Adamopoulos and A. Tuzhilin, "On unexpectedness in recommender systems: Or how to expect the unexpected," in *Proc. ACM Workshop Novelty Diversity Rec. Syst. (DiveRS)*, Chicago, IL, USA, Oct. 2011, pp. 11–18.
[17] H. Andrat and N. Ansari, "Analyzing game stickiness using clustering techniques," *Adv. Comput. Comput. Sci.*, vol. 554, pp. 645–654, Sep. 2017.
[18] K. Chowdhury, D. Chaudhuri, and A. K. Pal, "A novel objective function based clustering with optimal number of clusters," *Methodologies Application Issues of Contemporary Computing Framework*, pp. 23–32, Sep. 2018.
[19] S.-H. Lee, Y.-S. Jeong, J.-Y. Kim, and M. K. Jeong, "A new clustering validity index for arbitrary shape of clusters," *Pattern Recognit. Lett.*, vol. 112, pp. 263–269, Sep. 2018.
[20] A. Tehreem, S. G. Khawaja, A. M. Khan, M. U. Akram, and S. A. Khan, "Multiprocessor architecture for real-time applications using mean shift clustering," *J. Real-Time Image Process.*, pp. 1–14, Nov. 2017.
[21] T. VoVan and T. N. Trang, "Similar coefficient of cluster for discrete elements," *Sankhya B*, vol. 80, no. 1, pp. 19–36, 2018.
[22] C.-L. Huang, P.-H. Yeh, C.-W. Lin, and D.-C. Wu, "Utilizing user tag-based interests in recommender systems for social resource sharing Websites," *Knowl.-Based Syst.*, vol. 56, pp. 86–96, Jan. 2014.
[23] B. Yin, Y. Yang, and W. Liu, "Exploring social activeness and dynamic interest in community-based recommender system," in *Proc. 23rd Int. Conf. World Wide Web*, Seoul, South Korea, 2014, pp. 771–776.
[24] R. Guerraoui, A.-M. Kermarrec, R. Patra, and M. Taziki, "D2P: Distance-based differential privacy in recommenders," *Proc. VLDB Endowment*, vol. 8, no. 8, pp. 862–873, Apr. 2015.
[25] H. Koohi and K. Kiani, "User based collaborative filtering using fuzzy C-means," *Measurement*, vol. 91, pp. 134–139, Sep. 2016.
[26] J. Chen, H. Wang, and Z. Yan, "Evolutionary heterogeneous clustering for rating prediction based on user collaborative filtering," *Swarm Evol. Comput.*, vol. 38, pp. 35–41, Feb. 2018.
[27] U. Liji, Y. Chai, and J. Chen, "Improved personalized recommendation based on user attributes clustering and score matrix filling," *Comput. Standards Interfaces*, vol. 57, pp. 59–67, Mar. 2018.
[28] J. Chen, L. Wei, Uliji, and L. Zhang, "Dynamic evolutionary clustering approach based on time weight and latent attributes for collaborative filtering recommendation," *Chaos Solitons Fractals*, vol. 114, pp. 8–18, Sep. 2018.
[29] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-n recommender systems," in *Proc. 9th ACM Int. Conf. Web Search Data Mining*, New York, NY, USA, Feb. 2016, pp. 153–162.
[30] C.-X. Jia and R.-R. Liu, "Improve the algorithmic performance of collaborative filtering by using the interevent time distribution of human behaviors," *Phys. A, Stat. Mech. Appl.*, vol. 436, pp. 236–245, Oct. 2015.
[31] Y. Liu, J. Cheng, C. Yan, X. Wu, and F. Chen, "Research on the Matthews correlation coefficients metrics of personalized recommendation algorithm evaluation," *Int. J. Hybrid Inf. Technol.*, vol. 8, no. 1, pp. 163–172, Jan. 2015.
[32] D. Wu, G. Zhang, and J. Lu, "A fuzzy preference tree-based recommender system for personalized business-to-business E-services," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 1, pp. 29–43, Feb. 2015.
[33] Z. Yang, B. Wu, K. Zheng, X. Wang, and L. Lei, "A survey of collaborative filtering-based recommender systems for mobile Internet applications," *IEEE Access*, vol. 4, pp. 3273–3287, 2016.
[34] U. von Luxburg, "A tutorial on spectral clustering," *Statist. Comput.*, vol. 17, no. 4, pp. 395–416, 2007.
[35] J. C. Bezdek, R. Ehrlich, and W. Full, "FCM: The fuzzy *c*-means clustering algorithm," *Comput. Geosci.*, vol. 10, nos. 2–3, pp. 191–203, 1984.
[36] T. Xu, J. Tian, and T. Murata, "Research on personalized recommendation in E-commerce service based on data mining," in *Proc. Int. MultiConf. Eng. Comput. Scientists*, Hong Kong, Mar. 2013, pp. 313–317.
[37] A. L. V. Pereira and E. R. Hruschka, "Simultaneous co-clustering and learning to address the cold start problem in recommender systems," *Knowl.-Based Syst.*, vol. 82, pp. 11–19, Jul. 2015.
[38] W.-S. Hwang, J. Parc, S.-W. Kim, J. Lee, and D. Lee, "'Told you i didn't like it': Exploiting uninteresting items for effective collaborative filtering," in *Proc. IEEE 32nd Int. Conf. Data Eng. (ICDE)*, Helsinki, Finland, May 2016, pp. 349–360.
[39] P. Cremonesi, Y. Koren, and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks," in *Proc. 4th ACM Conf. Recommender Syst.*, Barcelona, Spain, Sep. 2010, pp. 39–46.

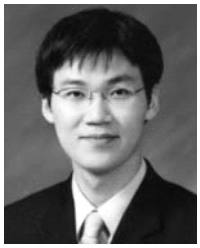
[40] H. Steck, "Training and testing of recommender systems on data missing not at random," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Washington, DC, USA, Jul. 2010, pp. 713–722.

[41] N. R. Pal and J. C. Bezdek, "On cluster validity for the fuzzy *c*-means model," *IEEE Trans. Fuzzy Syst.*, vol. 3, no. 3, pp. 370–379, Aug. 1995.

[42] H. Jazayeriy, S. Mohammadi, and S. Shamshirband, "A fast recommender system for cold user using categorized items," *Math. Comput. Appl.*, vol. 23, no. 1, p. 1, Jan. 2018.

CONG TRAN received the B.S. degree in network and communication and the M.S. degree in computer science from Vietnam National University, Hanoi, Vietnam, in 2009 and 2014, respectively. Since 2016, he has been with the Department of Computer Science and Engineering, Dankook University, Yongin, South Korea, where he is currently pursuing the Ph.D. degree. Since 2019, he has also been with the Department of Computational Science and Engineering, Yonsei University, Seoul, South Korea. His research interests include social network analysis, data mining, and machine learning.

JANG-YOUNG KIM received the B.S. degree in computer science from Yonsei University, Seoul, South Korea, the M.S. degree in computer science and engineering from Pennsylvania State University in University Park, and the Ph.D. degree in computer science and engineering from the University at Buffalo (SUNY). He was a Teaching Assistant with Pennsylvania State University. He was an Assistant Professor in computer science with the University of South Carolina. He is currently an Assistant Professor in computer science with the University of Suwon. His research interests include data-intensive distributed computing, cloud computing, and throughput optimization in high-speed networks.



WON-YONG SHIN (S'02–M'08–SM'16) received the B.S. degree in electrical engineering from Yonsei University, Seoul, South Korea, in 2002, the M.S. and Ph.D. degrees in electrical engineering and computer science from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2004 and 2008, respectively.

In 2009, he joined the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA, USA, as a Postdoctoral Fellow and he was promoted to a Research Associate, in 2011. From 2012 to 2019, he was a Faculty Member of the Department of Computer Science and Engineering, Dankook University, Yongin, South Korea. Since 2019, he has been with the Department of Computational Science and Engineering, Yonsei University, Seoul, South Korea, where he is currently an Associate Professor. His research interests include the areas of information theory, communications, signal processing, mobile computing, big data analytics, and online social networks analysis. From 2014 to 2018, he has served as an Associate Editor for the *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*. He has also served as an Organizing Committee Member of the 2015 IEEE Information Theory Workshop. He has received the Bronze Prize of the Samsung Humantech Paper Contest, in 2008, and the KICS Haedong Young Scholar Award, in 2016.

SANG-WOOK KIM received the B.S. degree in computer engineering from Seoul National University, South Korea, in 1989, and the M.S. and Ph.D. degrees in computer science from the Korea Advanced Institute of Science and Technology (KAIST), South Korea, in 1991 and 1994, respectively. From 1995 to 2003, he was an Associate Professor with the Division of Computer, Information, and Communications Engineering, Kangwon National University, South Korea. In 2003, he joined Hanyang University, Seoul, South Korea, where he is currently a Professor with the Department of Computer Science and Engineering and the Director of the Brain-Korea-21-Plus Research Program. He has been leading the DOYAK Project (formerly known as the National Research Lab Project), since 2015. From 2009 to 2010, he visited the Computer Science Department, Carnegie Mellon University, as a Visiting Research Professor. From 1999 to 2000, he was with the IBM T. J. Watson Research Center, USA, where he held a postdoctoral position. He has authored over 110 papers in refereed international journals and conference proceedings. His research interests include databases, data mining, multimedia information retrieval, social network analysis, recommendation, and web data analysis. He is a member of the ACM. He also visited the Computer Science Department, Stanford University, in 1991, as a Visiting Researcher. He was the Program Co-Chair of the International Conference on Emerging Databases, in 2013, the General Co-Chair of the Sixth International Conference on Computational Collective Intelligence Technologies and Applications, in 2014, the Program Co-Chair of the ACM International Conference on Ubiquitous Information Management and Communications (ACM IMCOM), in 2014, and the Track Chair of the Social Network and Media Analysis Track in the ACM International Symposium on Applied Computing (ACM SAC), in 2017. He has also served on Program Committees of over 80 international conferences including the IEEE ICDE, VLDB, WWW, and ACM CIKM. He has received the Best Poster Presentation Award from the ACM International Conference on Information and Knowledge Management (ACM CIKM), in 2013, and the Best Paper Award from the 29th ACM International Symposium on Applied Computing (ACM SAC), in 2014. He is currently an Associate Editor of two international journals including *Information Sciences* and *Computer Science & Information Systems (ComSIS)*. He was a recipient of the Outstanding Service Award from ACM SIGAPP and the Outstanding Contributions Award from the Database Society of Korea, in 2014. He has received the Presidential Award of Korea, in 2017, for his academic achievement.