Journal of information and communication convergence engineering
http://jicce.org

J. Inf. Commun. Converg. Eng. 17(2): 135-141, Jun. 2019

Regular paper

# Collaborative Filtering Algorithm Based on User–Item Attribute Preference

**JiaQi Ji**[1], **Yeongjee Chung**[2*], ***Member, KIICE***

[1]Department of Information Center, Hebei Normal University for Nationalities, Chengde 067000, China
[2]Department of Computer and Software Engineering, Wonkwang University, Iksan 54538, Korea

## Abstract

Collaborative filtering algorithms often encounter data sparsity issues. To overcome this issue, auxiliary information of relevant items is analyzed and an item attribute matrix is derived. In this study, we combine the user–item attribute preference with the traditional similarity calculation method to develop an improved similarity calculation approach and use weights to control the importance of these two elements. A collaborative filtering algorithm based on user–item attribute preference is proposed. The experimental results show that the performance of the recommender system is the most optimal when the weight of traditional similarity is equal to that of user–item attribute preference similarity. Although the rating-matrix is sparse, better recommendation results can be obtained by adding a suitable proportion of user–item attribute preference similarity. Moreover, the mean absolute error of the proposed approach is less than that of two traditional collaborative filtering algorithms.

**Index Terms**: Collaborative filtering, Recommender system, Attribute preference, Data sparsity

## I. INTRODUCTION

In the data explosion era, the recommender system (RS) is becoming one of the most effective tools to reduce information overload [1]. An RS can provide personalized recommendations to users depending on their past behaviors. Three major categories of RSs include the content-based RSs [2], collaborative filtering (CF) RSs [3], and hybrid RSs [4]. Among these, CF is the most popular approach to build an RS and it is widely adopted in data industries. The CF technique assumes that the users who assigned similar ratings to the same items tend to have similar preferences. Two main approaches to CF exist: item-based CF that associates the item with nearest neighbors [5], and user-based CF that associates a set of nearest neighbors with each user [6].

However, the RS frequently encounters data sparsity problems in real-world applications. In most applications, the number of users and items is increasing drastically. However, the items rated by users are few and relatively scattered; therefore, the user–item rating matrix is very sparse [7]. If the original data accumulated by the RS is minimal, it is difficult to estimate the similarities among the users. Therefore, the recommendation accuracy of the RS is poor, especially during the early development of the RS.

Clustering of recommenders has been investigated in RS as an unsupervised learning method [8]. In general, the idea of dividing big communities of users into smaller sets (clusters) has seemed to offer advantages, such as scalability, which as a result improves the response time, due to the smaller set of data that algorithms operate on. However, the loss of prediction accuracy is not compensated at a sufficient level to render the use of clusters an attractive solution.

In recent years, researchers have conducted several studies to resolve the issues of data sparsity in the user–item rating

**Received** 15 June 2019, **Revised** 21 June 2019, **Accepted** 21 June 2019
***Corresponding Author** Yeongjee Chung (E-mail: yeongjee@gmail.com, Tel: +82-63-850-6887)
Dept. of Computer and Software Engineering, Wonkwang University, Iksan 54538, Korea.

**Open Access** **https://doi.org/10.6109/jicce.2019.17.2.135**    print ISSN: 2234-8255    online ISSN: 2234-8883

matrix. The most effective solution is to introduce some auxiliary information (i.e., tags, review, and item description). To provide a recommendation of news (items) to a specific reader (user), we do not only calculate the similarity among the readers but also obtain all the news the reader has read earlier, analyze the contents of this news, and combine this together for a recommendation. In [9], researchers calculated the similarities among the users, used user-based clustering to group the users, and generated recommendations for each group.

Yu et al. [10] proposed an algorithm of cross-domain CF using attribute construction and locally weighted linear regression. They constructed attributes in different domains and used these attributes to represent different auxiliary domains. Their proposed algorithm has outperformed state-of-the-art algorithms at various sparsity levels.

In this study, the attributes of the item were analyzed, and the item-attribute matrix was introduced to alleviate the sparsity problem of the rating matrix. We combined the traditional similarity calculation and user–item attribute preference similarity calculation methods for performing the similarity calculation and proposed the user–item attribute preference collaborative filtering algorithm (UIAP-CF).

## II. RELATED WORK

CF techniques play a significant role in recommender systems and are mainly classified into user-based CF and item-based CF methods. A critical step in the user-based CF algorithm is the similarity measure based on past user behaviors, because users who demonstrate similar past behaviors exhibit similar preferences on items. The rating matrix R can be used to calculate the similarity among users. Each row of R represents the user ratings on different items, and each column of R denotes an item rating provided by different users. Traditional similarity measures, such as cosine similarity (COS) and Pearson correlation coefficient (PCC), have been applied in CF for decades.

The similarity can be calculated using the PCC formula for covariance of the preferences (ratings) of two users divided by their standard deviations based on co-related items [11]. The formula for PCC is as follows:

$$PCC(u,v)=\frac{\sum_{i\in I}(r_{u,i}-\overline{r}_u)\cdot(r_{v,i}-\overline{r}_v)}{\sqrt{\sum_{i\in I}(r_{u,i}-\overline{r}_u)^2}\sqrt{\sum_{i\in I}(r_{v,i}-\overline{r}_v)^2}}. \quad (1)$$

where $I$ is the set of co-related items for users $u$ and $v$, $r_{u,i}$ is the rating of item $i$ by user $u$, and $\overline{r}_u$ is the average rating of user $u$ for all the correlated items.

By calculating the value of the cosine angle between two vectors of ratings, we can compute COS that shows the similarity between the two users [11].

$$COS(u,v)=\frac{\sum_{i\in I}r_{u,i}\cdot r_{v,i}}{\sqrt{\sum_{i\in I}r_{u,i}^2}\sqrt{\sum_{i\in I}r_{v,i}^2}}. \quad (2)$$

Without loss of generality, we call these two similarity calculation methods traditional similarity calculation approaches (or traditional similarity measures) in this paper. Because of inherent limitations in traditional similarity measures [12-16], researchers have proposed many improved or novel similarity models, which will be discussed in Section III. A.

## III. PROPOSED ALGORITHM

### *A. Drawbacks of Traditional User Similarity Measures*

The key feature of the user-based CF algorithm is the user similarity calculation, and the calculation results exhibit a profound influence on the accuracy of the RS. The user similarity is evaluated based only on users' ratings of items in the traditional user-based CF. It ignores potential associated information between the users. An example is provided to illustrate this problem.

In Table 1, $u_1$, $u_2$, and $u_3$ denote users; $i_1$, $i_2$, $i_3$, $i_4$, $i_5$, $i_6$, and $i_7$ represent movies. If $u_1$ rated $i_1$ (the number of ratings does not matter), the value is 1, otherwise it is 0.

Table 2 is a movie genre matrix, in which 1 indicates that the movie is the corresponding type. For instance, the type of movie $i_1$ corresponds to "Action," "Romance," and "Drama."

**Table 1.** User-rating matrix

| | $i_1$ | $i_1$ | $i_3$ | $i_4$ | $i_5$ | $i_6$ | $i_7$ |
|---|---|---|---|---|---|---|---|
| $u_1$ | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| $u_2$ | 1 | 0 | 0 | 1 | 1 | 0 | 0 |
| $u_3$ | 1 | 0 | 0 | 0 | 0 | 1 | 1 |

**Table 2.** Movie genre matrix

| | Action | Romance | Drama | Sci-Fi | Mystery | Horror | Documentary | Animation | War |
|---|---|---|---|---|---|---|---|---|---|
| $i_1$ | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| $i_2$ | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| $i_3$ | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| $i_4$ | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| $i_5$ | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| $i_6$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| $i_7$ | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |

According to the traditional user-based CF algorithm, the similarity (COS) between $u_1$ and $u_2$ is 1/3, and COS between $u_1$ and $u_3$ is 1/3 as well. It can be observed from the calculation results that the similarity between $u_1$ and $u_2$ is the same as that between $u_1$ and $u_3$. However, if we analyze the type of movies that the users prefer, we can observe that $u_1$ prefers $i_1$, $i_2$, and $i_3$. Next, $i_1$, $i_2$, and $i_3$ belong to the category of "Action." Therefore, we can determine that user $u_1$ prefers "Action" movies. By analogy, we can infer that user $u_2$ prefers "Action," and $u_3$ prefers "Animation" or "War." In this case, the calculated similarity between $u_1$ and $u_2$ is greater than that between $u_1$ and $u_3$.

From the prior analysis, we can conclude that the similarity between users is calculated based on not only the user-rating matrix but also the user-attribute similarity. To overcome this drawback, we proposed a novel algorithm called the UIAP-CF, which is based on collaborative filtering. In the subsequent section, we will introduce this algorithm in detail.

### *B. UIAP-CF*

The user-attribute similarity can be calculated according to the types of movies preferred by the user. To calculate the user-attribute similarity, we must construct the user-attribute vector. Therefore, we first introduced the method to construct the user-attribute vector, and then the calculation method of user-attribute similarity is provided. Without loss of generality, the movies were considered as items.

#### *1) Item-Attribute Vector*

Each item included one or more attributes. For a movie, we constructed the movie-attribute vector based on whether it belonged to a certain type. The movie genre can be regarded as item-attributes. Therefore, (3) was used to describe the item-attribute vector, where $f_i(m)$ is 1 or 0, which implies that the item possesses/does not possess a particular attribute. The number of attributes is denoted as $k$. For instance, from Table 2, we observe that the number of attributes $k$ = 9, and item $i_1$ can be denoted as $F(i_1)$ = {1, 1, 1, 0, 0, 0, 0, 0, 0}.

$$FM(m) = \{f_1(m), f_2(m), ..., f_k(m)\}. \tag{3}$$

#### *2) User-Attribute Vector*

*FU(u)* is used to describe the user-attribute vector, which is defined in (4), where $c_i(u)$ is equal to the number of attributes $i$ in the item (movie) preferred by user $u$ divided by the total number of types contained in the user's preferred movie for normalization.

$$FU(u) = \{c_1(u), c_2(u), ..., c_k(u)\}. \tag{4}$$

$$c_i(u) = \frac{\sum_{m \in S(u)} f_i(m)}{d}. \tag{5}$$

For user $u$, we constructed $S(u)$, which included the set of items (movies) rated by $u$. For example, user $u_1$ rated $i_1$, $i_2$, and $i_3$. Therefore, $S(u_1) = \{i_1, i_2, i_3\}$.

We provide an example to further illustrate the aforementioned formula. User $u_1$ rated items $i_1$, $i_2$, and $i_3$. Therefore, the total number of item-attributes for $i_1$, $i_2$, and $i_3$ is estimated as $d$ = 7, where the number of "Action" movies is 3, "Romance" is 1, "Drama" is 1, "Sci-Fi" is 1, and "Mystery" is 1. Next, we calculate $c_1(u_1)$ = 3/7, $c_2(u_1)$ = 1/7, $c_3(u_1)$ = 1/7, $c_4(u_1)$ = 1/7, $c_5(u_1)$ = 1/7, $c_6(u_1)$ = 0/7, $c_7(u_1)$ = 0/7, $c_8(u_1)$ = 0/7, and $c_9(u_1)$ = 0/7, and therefore, $FU(u_1)$ = {3/7, 1/7, 1/7, 1/7, 1/7, 0, 0, 0, 0}.

The above process is repeated to obtain all the user-attribute vectors.

#### *3) User-Attribute Similarity*

When the user-attribute vector was obtained, we calculated the user-attribute similarity using (6).

$$\begin{aligned} Sim\_UVector(u,v) &= \frac{FU(u) \cdot FU(v)}{||FU(u)|| \times ||FU(v)||} \\ &= \frac{\sum_{i=1}^{k} c_i(u) \times c_i(v)}{\sqrt{\sum_{i=1}^{k} c_i(u)^2} \times \sqrt{\sum_{i=1}^{k} c_i(v)^2}} \end{aligned} \tag{6}$$

The similarity was calculated between $u$ and $v$ using COS by (6). Instead, we can use the Pearson correlation coefficient, Jaccard similarity, or other similarity measures.

#### *4) User Similarity*

The optimized user similarity was calculated not only using (1) and (2) but also using the user-attribute similarity (6). Additionally, we believe that the optimization method can be applied to user similarity measures by adding $\lambda$, a weight adjustment parameter, especially when it consists of several parts. Therefore, the formula is redefined as follows:

$$\begin{aligned} Sim(u,v) &= \lambda \times Sim\_UVector(u,v) \\ &+ (1-\lambda) \times Sim\_Traditional(u,v) \end{aligned} \tag{7}$$

where *Sim_Traditional*$(u, v)$ is calculated by (1) or (2). *Sim_UVector* $(u, v)$ is calculated by the user-attribute vector using (6). $\lambda$ is the weight adjustment parameter with a range of [0,1]. When $\lambda$ equals 1, it implies that only user-attribute similarity is used; when $\lambda$ equals 0, it implies that only the rating matrix is used for calculations; when $\lambda$ is between 0 and 1, the results of the final similarity is obtained by combining the

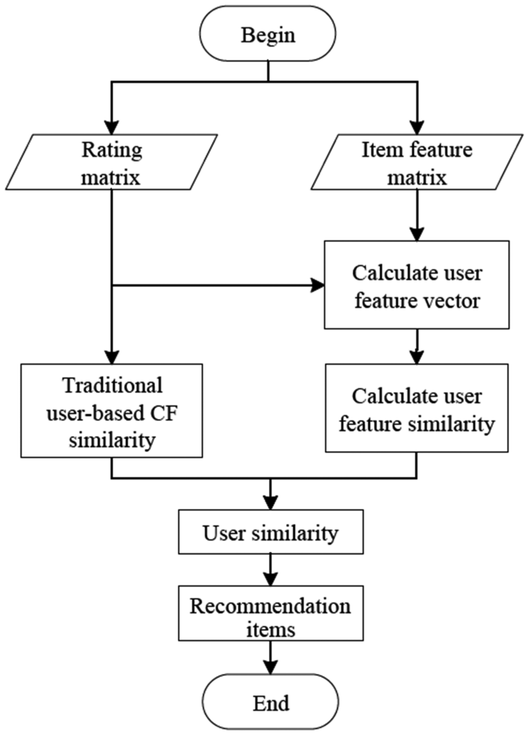


**Fig. 1.** User-based CF flowchart.

user-attribute similarity and the similarity achieved using the traditional CF recommender algorithm.

Therefore, a calculation method of user similarity is finally achieved with steps 1) to 4) to alleviate the drawbacks of traditional user similarity. Fig. 1 shows the flowchart of the improved user-based CF algorithm.

From Fig. 1, we can observe that UIAP-CF first calculates the user-attribute similarity and traditional user-based CF similarity, combines the two calculation results to provide the final user similarity, and finally recommends *n* items that the user may prefer the most.

### C. Generation of Recommendation

According to the similarity between the target user and the nearest neighbors, the classical nearest neighbor prediction was used to calculate the unrated items, as shown in (8).

$$P_{u,i} = \overline{R}_u + \frac{\sum_{v \in knn} sim\_user(u,v)(R_{v,i} - \overline{R}_v)}{\sum_{v \in knn} sim\_user(u,v)}. \quad (8)$$

where $\overline{R}_u$ is the average rating value rated by *u*, and *knn* is the collection of *k* nearest neighbors of *u*.

User-based CF also calculates similarities between users to find the nearest neighbors of the user according to user–item attribute preference. The similarity measure weighted by the nearest neighbors for each target item that takes the proportion of co-related ratings into account. We believe that the prediction of ratings on unrated items is usually based on the ratings of the nearest neighbors in a user-based CF system.

### D. Clustering of Recommenders

The main idea behind clustering is to permanently partition choices into smaller sets to simplify a future choice for a neighbor. This is an idea adopted in social networking, where information can be overwhelming. Although an algorithm for creating a uniform cluster of items works by minimizing the variance between items in the same cluster, its applicability is limited to the special algorithm of item-based CF, in which correlation is performed on items rather than users.

## IV. EXPERIMENTS

### A. Dataset

In the experiment, the MovieLens 1M dataset provided by GroupLens Research were used. The rating range was 1–5, in which a higher value indicates the level of preference of the user. The movie attributes in MovieLens included the movie number, movie name, movie release date, and movie genre. We considered 18 different types of movies, and each movie belonged to multiple genres. This dataset contained data on 6010 users, 3952 movies, and 1,000,209 ratings. It was obvious that the dataset was very sparse (1 – 1000209/(6010 * 3952) = 95.75%).

The dataset was divided into a training set, validation set, and test set. 80% of the data was used as the training set, 10% of the data was used as the validate set, and 10% of the data was used as the test set. The model was trained by the training set, the parameters were determined by the validation set, and the performance of algorithm was evaluated by the test set. We used the 5-fold cross validation method to obtain the final experiment results.

### B. Parameter Choices

#### 1) Choice 1: λ

$\lambda$ is the parameter used to adjust the proportion between the traditional user similarity *Sim_Traditional*(*u*, *v*) and user–item preference similarity *Sim_UVector*(*u, v*). *Sim*(*u, v*) is calculated by the user-attribute vector with the weight adjustment parameter $\lambda$, which is in the range of [0,1]. We used the mean absolute error (MAE) defined in (9) to evaluate the performance of RS. In this formula, *N* is the number of ratings, $r_i$ is a rating in the training set, and $\hat{r}_i$ is the prediction rating response to $r_i$. With a lower value of MAE, the performance is better.

$$MAE = \frac{\sum_{i=1}^{N} | r_i - \hat{r}_i |}{N}. \quad (9)$$

Obviously, it was equivalent to using only the traditional similarity calculation method, when $\lambda$ was 0. In contrast, only the user–item preference similarity calculation method was used, when $\lambda$ was 1. We tuned the value of $\lambda$ from 0 to 1, and the MAE results are shown in Fig. 2.

From Fig. 2, we can conclude that the MAE first decreases and then increases with the increase of $\lambda$. The performance obtained using only the user–item preference similarity calculation method ($\lambda$ = 0) is observed to be worse than the performance obtained using only the traditional similarity calculation method ($\lambda$ = 1). The performance of RS is the most optimal when $\lambda$ = 0.5.

Moreover, the influence of the distinct size of the training set on $\lambda$ was also considered. In the previous experiment, the proportion of the training set was equal to 0.8 (T = 0.8). A smaller value of T denotes a sparser dataset. Thus, we varied T from 0.8 to 0.6 to validate the effect of data sparsity on $\lambda$. From Fig. 3, we can observe that a similar trend was achieved in previous experiments. With the increase of $\lambda$, the performance of RS increases first and then decreases. When T = 0.8 and $\lambda$ = 0, the MAE is 0.89. However, when T = 0.6 and $\lambda$ = 0.7, the MAE is 0.771. Therefore, it is evident that even if the dataset becomes sparse, the MAE does not increase. Instead, the MAE decreases whent the user–item preference similarity is introduced.

### 2) Choice 2: Number of Neighbors

The number of nearest neighbors also demonstrates a considerable influence on the recommendation quality. If the set of neighbors is larger, it does not only affect the recommendation quality but also increases the amount of calculation and reduces the recommendation efficiency. In the subsequent experiment, we set $\lambda$ = 0.5, T = 0.8, and varied the number of neighbors from 5 to 60. From Fig. 4, we can observe that when the number of nearest neighbors is less than 30, the performance of the RS increases with an increase in the number of neighbors. However, when the number of nearest neighbors is greater than 30, the MAE remains almost unchanged. This shows that the performance of the RS is the most optimal when the number of nearest neighbors is 30.

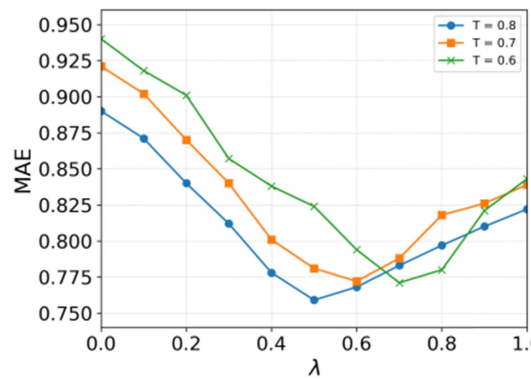


**Fig. 3.** Effect of data sparsity on $\lambda$.

## C. Comparison with Baselines

To establish the effectiveness of our proposed algorithm, we compared it with the traditional user-based CF algorithm. In the subsequent experiment, we set $\lambda$ = 0.5, T = 0.8, and the number of neighbors to 30. For traditional user-based CF, we used the cosine and Pearson similarity calculation methods.

Fig. 5 shows that the proposed UIAP-CF performs considerably better than other methods. This also establishes that the user–item preference similarity calculation method introduced in this study is effective, as it can mitigate the sparse-

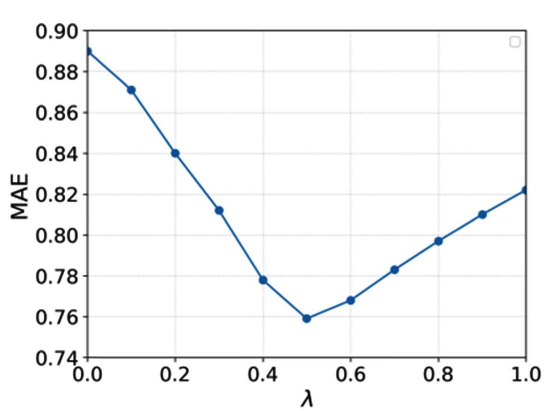


**Fig. 2.** Influence of $\lambda$ on MAE.

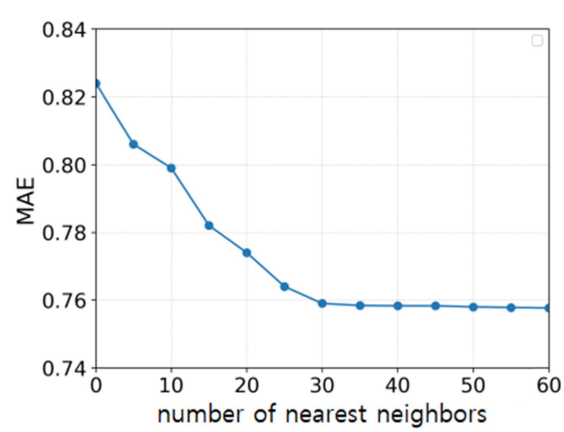


**Fig. 4.** Influence of neighbor set size on MAE.

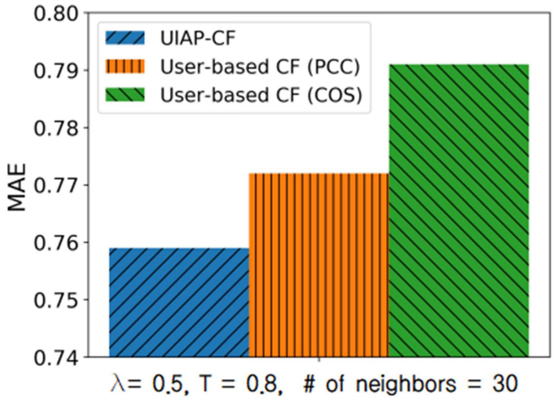


**Fig. 5.** Comparison with baselines.

ness of data and improve the performance of RS.

Apparently, with clustering of recommenders, the information overload is reduced. Such user–item preference similarity as an input to the clustering of recommenders is already used in social networks as core information. It should be noted that clustering, in general, has a drawback in terms of the number of predictions that can be produced for clustered users.

## V. CONCLUSION

This paper describes the drawbacks related to traditional CF algorithms including their data sparsity issues in real-world applications. To overcome these issues, we introduced a user–item preference similarity calculation approach as a CF algorithm based on UIAP. Next, some parameters of the algorithm were determined. Finally, the recommendation performance of UIAP-CF was compared with that of the traditional CF algorithm. The results obtained show that the performance of the RS is the most optimal when the weight coefficient ($\lambda$) is 0.5, i.e., when the weight of user similarity *Sim_Traditional(u,v)* of the traditional CF recommendation algorithm is equal to that of the user–item preference similarity *Sim_UVector(u,v)*. The MAE of the UIAP-CF was observed to be less than that of the traditional CF algorithm (Pearson and COS). Therefore, the UIAP-CF was demonstrated to deliver better recommendation results and alleviate data sparsity.

## ACKNOWLEDGMENTS

This paper was supported by Wonkwang University, Iksan Korea, in 2019.

## REFERENCES

[ 1 ] Y. Liu, J. Nie, L. Xu, Y. Chen, and B. Xu, “Clothing recommendation system based on advanced user-based collaborative filtering algorithm,” in *Lecture Notes in Electrical Engineering*, vol 473. Springer, Singapore, pp. 436-443, 2017. DOI: 10.1007/978-981-10-7521-6_53.

[ 2 ] D. Wang, Y. Liang, D. Xu, X. Feng, and R. Guan, “A Content-Based Recommender System for Computer Science Publications,” *Knowledge-Based Systems,* Elsevier, vol. 157, pp. 1-9, 2018. DOI: /10.1016/j.knosys.2018.05.001.

[ 3 ] M. K. Najafabadi and M. N. R. Mahrin, “A systematic literature review on the state of research and practice of collaborative filtering technique and implicit feedback,” *Artificial Intelligence Review, An International Science and Engineering Journal,* vol. 45, Issue 2, pp. 167-201, 2016. DOI: 10.1007/s10462-015-9443-9.

[ 4 ] Y. Song, S. Liu, and W. Ji, “Research on personalized hybrid recommendation system,” in *Proceeding of 2017 International Conference on Computer, Information and Telecommunication Systems (CITS)*, Dalian, China, pp. 133-137, 2017. DOI: 10.1109/CITS.2017.8035321.

[ 5 ] A. Bilge and C. Kaleli, “A multi-criteria item-based collaborative filtering framework,” in *Proceeding of 2014 11th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Chonburi, Thailand, pp. 18-22, 2014. DOI: 10.1109/JCSSE.2014.6841835.

[ 6 ] Q. Shambour, “A user-based multi-criteria recommendation approach for personalized recommendations,” *International Journal of Computer Science and Information Security (IJCSIS)*, vol. 14, pp. 657-663. 2016.

[ 7 ] Z. L. Zhao, L. Huang, C. D. Wang, J. H. Lai, and P. S. Yu, “Low-rank and sparse matrix completion for recommendation,” in *Neural Information Processing*, Springer International Publishing, pp. 3-13, 2017. DOI: 10.1007/978-3-319-70139-4_1.

[ 8 ] M. Ester, H. P. Kriegel, J. Sander, and X. Xu, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *Proceeding of Second International Conference on Knowledge Discovery and Data Mining*, pp. 226-231. AAAI Press, Menlo Park, 1996.

[ 9 ] L. Yanxiang, G. Deke, C. Fei, and C. Honghui, “User-based clustering with top-n recommendation on cold-start problem,” in *Proceeding of 2013 Third International Conference on Intelligent System Design and Engineering Applications (ISDEA)*, Hong Kong, China, pp. 1585-1589, 2013. DOI: 10.1109/ISDEA.2012. 381.

[10] X. Yu, J. Y. Lin, F. Jiang, J.W. Du, and J. Z. Han, “A cross-domain collaborative filtering algorithm based on feature construction and locally weighted linear regression,” *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 1425365, 2018. DOI: 10.1155/2018/1425365.

[11] Z. Tan and L. He, “An efficient similarity measure for user-based collaborative filtering recommender systems inspired by the physical resonance principle,” *IEEE Access,* vol. 5, pp. 27211-27228, 2017. DOI: 10.1109/ACCESS.2017.2778424.

[12] L. Candillier, F. Meyer, and F. Fessant, “Designing specific weighted similarity measures to improve collaborative filtering systems,” in *Proeeding of ICDM*, Leipzig, Germany, pp. 242-255, 2008. DOI: 10.1007/978-3-540-70720-2_19.

[13] G. Guo, J. Zhang, and N. Yorke-Smith, “A novel Bayesian similarity measure for recommender systems,” in *Proceeding of IJCAI*, Beijing China, pp. 2619-2625, 2013, [Online] Available: *http://www.ijcai.org/Proceedings/13/Papers/386.pdf.*

[14] G. Guo, J. Zhang, and N. Yorke-Smith, “A novel evidence-based Bayesian similarity measure for recommender systems,” *ACM Trans.*

*Web,* vol. 10, no. 2, pp. 1-30, 2016. DOI: 10.1145/ 2856037.
[15] H. J. Ahn, “A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem,” *Inf. Sci.*, vol. 178, no. 1, pp. 37-51, 2008. DOI: 10.1016/j.ins.2007.07.024.
[16] N. Lathia, S. Hailes, and L. Capra, “The effect of correlation coefficients on communities of recommenders,” in *Proceeding of the 2008 ACM Symposium on Applied Computing - SAC’08,* Fortaleza, Brazil, pp. 2000-2005, 2008. DOI: 10.1145/1363686. 1364172.

**JiaQi Ji**
Dr. Ji received his Ph.D. degree from the Wonkwang University, Iksan, Korea, in 2018, and received his B.S. and M.S. degrees from the Department of Computer Science, Nanchang University, Nanchang, China, in 2007 and 2010, respectively. From 2010 to 2012, he worked as a software engineer in Beijing, China. From 2012 to 2016, he worked as a lecturer at the Department of Information Center, Hebei Normal University for Nationalities, Chengde, China. His current research interests include big data processing.

**Yeongjee Chung**
Dr. Chung received his Ph.D. degree from the Yonsei University, Seoul, Korea, in 1993. He is a professor in the Department of Computer Engineering at the Wonkwang University, Iksan, Korea, from 1995. His primary research interests include the internet of things, big data processing, and mobile computing platforms. He is working on establishing new knowledge for various applications including ubiquitous healthcare and mobile computing with big data processing.