Jurnal Ilmiah KURSOR
Menuju Solusi Teknologi Informasi
Vol. 10, No. 1, Juli 2019

ISSN 0216 – 0544
e-ISSN 2301– 6914

IMPACT OF IMPUTATION ON CLUSTER-BASED COLLABORATIVE FILTERING APPROACH FOR RECOMMENDATION SYSTEM

[a]Noor Ifada, [b]Susi Susanti, [c]Mula'ab

[a,b,c]Informatics Department, University of Trunojoyo Madura, Bangkalan, Indonesia 69162
E-mail: [a]noor.ifada@trunojoyo.ac.id, [b]susisusanti3012@gmail.com, [c]mulaab@trunojoyo.ac.id

Abstract

The Collaborative Filtering (CF) widely used in Recommendation System commonly suffers the sparsity issue since the unobserved rating entries usually over dominance the observed ones. A clustering technique is an alternative solution that can solve the problem. However, no in-depth work has investigated how the missing entries should be mitigated and how the cluster-based approach can be implemented. In this study, we show how the imputed cluster-based approach deals with the missing entries, improving the recommendation quality. The framework of our method consists of four main stages: rating imputation to replace the missing entries, K-means clustering to group users or items based on the imputed rating data, CF-based prediction model, and generating the list of top-N recommendation. This paper uses three variations of rating imputation techniques, i.e., null, mean, and mode. The cluster-based approach is employed by using the K-Means as the clustering technique, and either the user-based or the items-based model as the CF approach. Experiment results show that the null imputation technique performs the best compared to the mean and mean techniques when dealing with the missing entries. This finding indicates that the implementation of the clustering technique is sufficient for solving the sparsity issue such that imputing the missing entries is not necessary. We also show that our imputed cluster-based CF methods always outperform the traditional CF methods. The results confirm that the implementation of a cluster-based approach can improve the recommendation quality of traditional CF methods.

Keywords*: clustering, collaborative filtering, imputation, sparsity.*

INTRODUCTION

Recommendation Systems (RS) help users to tackle the problem of having to find items that suit their preference from the overwhelming amount of available items. RS can generate a set of personalized lists of item recommendations that might be of interest its users by learning through their previous rating activities [1, 2].

Collaborative Filtering (CF) approach is widely used in recommendation systems [1, 3]. The memory-based CF approach employs the users' or items' similarities to generate the list of recommendations to a target user, and therefore, it can be categorized as the user-based and item-based models [2]. In the user-based model, the list of recommendations is generated based on the users' similarities. Meanwhile, the item-based model generates the list based on the items' similarities that the user liked in the past.

The traditional CF is also known to typically suffer from a sparsity issue that impacts the recommendation performance [2, 4-6]. The issue occurs since, commonly, the unobserved entries over dominance the rating data. A clustering technique is a practical solution that can solve the problem, in which it creates groups of users [6-11] or items [12]. However, no in-depth work has been done that investigates how the missing entries should be mitigated using the imputation technique [13-15], and how the cluster-based approach can be implemented to the user-based and item-based CF approach comprehensively.

In this paper, we conduct an in-depth study on implementing the imputed cluster-based CF approach to improve the quality of recommendations of the traditional CF approach. Our work is focusing on addressing the sparsity challenge, i.e., by implementing the rating imputation technique and cluster-based approach that improves the quality of recommendations of the traditional CF approach. We show how the three variations of imputation techniques deal with the missing rating entries problem and how then the K-means clustering technique enhances the performance of the CF traditional methods.

Experimental results on a real-world rating dataset show that out of the three imputation techniques, the *null* imputation best deals with the missing entries. This finding indicates that the implementation of the clustering technique is sufficient for solving the sparsity issue such that imputing the missing entries is not necessary. We also show that the imputed cluster-based CF methods always outperform their traditional counterparts. These outcomes confirm that the cluster-based approach can improve the recommendation quality of RS.

The summary of our contributions is as follows: (1) the implementation of three rating imputation techniques to deal with the missing rating entries, and (2) the imputed cluster-based CF methods that improve the recommendation quality of the traditional CF approach by implementing the K-Means clustering technique on two CF-based models.

IMPUTED CLUSTER-BASED CF METHOD

The focus of our imputed cluster-based CF method is to improve the quality of recommendations of the traditional CF method by dealing with the sparsity issue commonly occurs in CF. In this paper, we use the rating data as the input of the method. The data consists of observed entries that form the binary correlations between users and items. Each rating score represents the user's level of preference for items.

Let $U = \{u_1, u_2, \dots, u_m\}$ and $I = \{i_1, i_2, \dots, i_n\}$ be the set of *m* users and *n* items. The correlation within rating data can be modeled as a rating matrix of $R \in \mathbb{R}^{m \times n}$ where r_{ui} represents the rating of item *i* given by user *u*. While I_u denotes the set of items that the user u have rated, whereas U_i denotes the set of users who have rated movie *i*.

Fig. 1 presents a toy example of rating matrix $R \in \mathbb{R}^{3\times6}$ where $U = \{u_1, u_2, u_3\}$ and $I = \{i_1, i_2, i_3, i_4, i_5, i_6\}$. Therefore, $I_1 = \{1, 2, 4, 5\}$, $I_2 = \{3, 5, 6\}$, and $I_3 = \{4, 6\}$. Meanwhile, $U_1 = \{1\}$, $U_2 = \{1\}$, $U_3 = \{2\}$, $U_4 = \{1, 2\}$, $U_5 = \{1, 2\}$, and $U_6 = \{2, 3\}$.

		Item					
		i_1	i_2	i_3	i_4	i_5	i_6
User	u_1	1	**3**	**0**	**2**	**3**	**0**
	u_2	0	**0**	**3**	**0**	**1**	**2**
	u_3	0	**0**	**0**	**4**	**0**	**4**

Fig. 1. A toy example of rating matrix $R \in \mathbb{R}^{3\times6}$

The framework of our method consists of four main stages (see Fig. 2), i.e., implementation of an imputation technique to replace the missing rating entries, implementation of K-means clustering to group users or items based on the imputed rating data, implementation of CF-based prediction model, and generating the list of top-N recommendation.

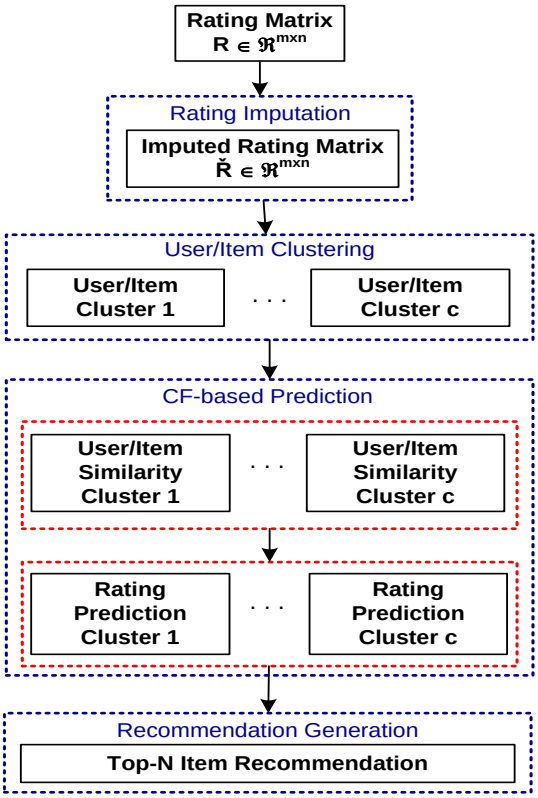

Fig. 2. The framework of our imputed cluster-based CF method

Rating Imputation

The rating imputation is the stage where we deal with the missing rating entries by replacing them with certain values [13-15].

This paper implements three imputation techniques, i.e., *null*, *mean*, and *mode* [16]. Those techniques respectively replace the missing entries by the zeroes, mean, and mode of rating values of each user or item. The rating imputation algorithm is presented in Fig. 43. At this stage, we now have the imputed rating matrix $\hat{R} \in \mathbb{R}^{m \times n}$ that will be used in the clustering stage.

Algorithm: Rating Imputation

Input: Rating Matrix $R \in \mathbb{R}^{m \times n}$,

Output: Imputed Rating Matrix $\hat{R} \in \mathbb{R}^{m \times n}$

Process:

1. For each user or item, calculate the mean and mode
2. Case:
   a. *Null* imputation: replace the missing rating entries with zero values
   b. *Mean* imputation: replace the missing rating entries with the mean of rating values of each user or item
   c. *Mode* imputation: replace the missing rating entries with the mode of rating values of each user or item

Fig. 3. Rating imputation algorithm

Clustering using K-Means

The clustering is the stage where we group the users or items such that the considered related users or items are unified in the same cluster.

This paper implements the well-known K-Means clustering technique to group users or items based on the rating data [8]. The algorithm is shown in Fig. 4.

Algorithm: K-Means Clustering

Input: Rating matrix $\hat{R}$, cluster size C

Output: C Clusters

Process:

1. Initialize C centroids randomly
2. Assign each user or item point to its closest centroid, based on the Euclidean distance
3. Update the centroids by taking the man of all users or items assigned to the centroid's cluster
4. Repeat steps 2 and 3 until no user or item points change clusters

Fig. 4. K-Means clustering algorithm

CF-based Prediction

As previously mentioned in the introduction section, the main step of the CF prediction approach is calculating the users' or items' similarities. Unlike the traditional approach that conducts the calculation for all users or items, our cluster-based CF methods calculate the similarities per user's or item's cluster only. In this paper, we implement two models

of the CF-based approach, i.e., user-based and item-based.

User-based Model

The user-based model calculates the rating predictions based on the users' similarities. In this paper, we use the Pearson similarity function to best estimate the users' rating similarities [2, 3]. Equation (1) formulates the similarity calculation between user u and v grouped in the same cluster:

$$SimU(u,v) = \frac{\sum_{k \in I_u \cap I_v} (\hat{r}_{uk} - \hat{\mu}_u) \cdot (\hat{r}_{vk} - \hat{\mu}_v)}{\sqrt{\sum_{k \in I_u \cap I_v} (\hat{r}_{uk} - \hat{\mu}_u)^2} \cdot \sqrt{\sum_{k \in I_u \cap I_v} (\hat{r}_{vk} - \hat{\mu}_v)^2}} \quad (1)$$

where $\hat{r}_{uk}$ and $\hat{r}_{vk}$ are the rating of movie k by user u and v. While $\hat{\mu}_u$ and $\hat{\mu}_v$ are the average rating of user u and v.

To calculate the rating prediction, the top-D nearest neighbors of user u regarding movie i, $S_u(i)$, is formed beforehand. Equation (2) formulates the rating prediction calculation of user u to movie i based on the cluster user-based model:

$$PU_{ui} = \mu_u + \frac{\sum_{v \in S_u(i)} \hat{r}_{vi} \cdot SimU(u,v)}{\sum_{v \in S_u(i)} |SimU(u,v)|} \quad (1)$$

where $|S_u(i)| \leq D$.

Item-based Model

The item-based model calculates the rating predictions based on the items' similarities that the user liked in the past. In this paper, we use the Adjusted Cosine similarity function to best estimate the items' similarities [2, 3]. Equation (3) formulates the similarity calculation between movie i and j grouped in the same cluster:

$$SimI(i,j) = \frac{\sum_{u \in U_i \cap U_j} (\hat{r}_{ui} - \hat{\mu}_u) \cdot (\hat{r}_{uj} - \hat{\mu}_u)}{\sqrt{\sum_{u \in U_i \cap U_j} (\hat{r}_{ui} - \hat{\mu}_u)^2} \cdot \sqrt{\sum_{u \in U_i \cap U_j} (\hat{r}_{uj} - \hat{\mu}_u)^2}} \quad (2)$$

where $\hat{r}_{ui}$ and $\hat{r}_{uj}$ are the rating given by user u to movie i and j. While $\hat{\mu}_u$ is the average rating of user u.

Based on the items' similarities, we form the top-D nearest neighbors of movie i of user u, $T_i(u)$. The rating prediction calculation of user u to movie i based on the cluster item-based model is formulated in Equation (4):

$$PI_{ui} = \frac{\sum_{j \in T_i(u)} \hat{r}_{uj} \cdot SimI(i,j)}{\sum_{j \in T_i(u)} |SimI(i,j)|} \quad (3)$$

where $|T_i(u)| \leq D$.

Recommendation Generation

The list of recommendations for each target user u is generated based on the rating predictions. In this case, the prediction scores are ranked in descending order such that the items with the top-N scores are recorded as the top-N list of recommendations, $Top(N)$.

Note that for ease of explanation, we label the cluster-based CF methods developed in this paper as *UCCF* and *ICCF*. *UCCF* is our cluster-based CF method that implements a combination of the user clustering algorithm and the user-based model to calculate the rating prediction and generate the list of recommendations. Whereas, *ICCF* is our cluster-based CF method that implements a combination of the item clustering algorithm and item-based model.

Based on the three imputation techniques implemented to deal with the missing rating entries, we vary the methods as *UCCF-Null*, *UCCF-Mean*, *UCCF-Mode*, *ICCF-Null*, *ICCF-Mean*, and *ICCF-Mode*.

RESULT AND DISCUSSION

In this section, we conduct and present the results of a series of experiments that evaluate the proposed imputed cluster-based CF methods. The research questions to address are: 1) Which imputation technique produces the best results? 2) How do the parameters of the imputed cluster-based method influence the recommendation performance? 3) Does the imputed cluster-based CF methods outperform the traditional CF methods?

Experiment Setup

We use the real-world MovieLens dataset retrieved from the GroupLens corpus (http://files.grouplens.org/datasets/movielens/ml-100k.zip) for the experiments. Table 1 details the description of the dataset. This paper uses the $u.data$ data – contains 943 users, 1682 movies/items, and 100K rating – to build the rating matrix $R \in \mathbb{R}^{943 \times 1682}$. The data density is 6.3047%, which means that it is very sparse due to the missing entries overdominance. The rating value is of five levels of preferences, i.e., 1 to 5, indicating the lowest to the highest level of fondness.

We evaluate the performances of methods by using the 5-fold cross-validation procedure,

such that each fold is randomly split into two sets: (1) training set D_{train} that consists of 80% rating data, used for building the recommendation model to generate the top-N list items $Top(N)$; and (2) test set D_{test} that consists of 20% rating data, used as the ground truth items GT. Note that the task of recommendation is to generate the top-N list of items for all target users in D_{test}. In this case, $Top(N)$ are compared to GT in D_{test} for each target user u.

Table 1. Description of Movie Lens Dataset

Data	Description
$u.data$	The rating data lists the correlations of user id, item id, rating, timestamp
$u.info$	The number of users, items, and ratings in the $\boldsymbol{u.data}$
$u.item$	The items' information that includes the movie id, title, release date, video release date, and genres
$u.genre$	The list of the movie's genres
$u.user$	The users' demographic information that includes the user id, age, gender, occupation, zip code
$u.occupation$	The list of occupations

We measure the performance of methods in recommending a list of items to each target user u using the F1-Score formula:

$$F1\text{-}Score(N) = \frac{2\cdot(Precision(N)\cdot Recall(N))}{Precision(N)+Recall(N)} \quad (4)$$

where the Precision and Recall are calculated as:

$$Precision(N) = 100 \cdot \frac{|Top(N)\cap GT|}{N} \quad (5)$$

$$Recall(N) = 100 \cdot \frac{|Top(N)\cap GT|}{|GT|} \quad (6)$$

The reported performance results shown in this paper are the average scores of all users in the D_{test}.

Experiment Results

Imputed cluster-based CF methods performance

In this sub-section, we compare the performance of the imputed cluster-based CF methods, i.e., *UCCF* and *ICCF*. This comparison is to analyze the impact of each imputation technique to the data densities and methods. Recall that the purpose of implementing the imputation technique is to fill in the missing value entries of rating data. Meanwhile, the purpose of implementing the clustering technique is to handle the sparse data.

Table 2 lists the density comparison between the rating matrix R and the imputed rating matrix $\hat{R}$. The density percentage is achieved by comparing the number of non-zero entries with the number of users multiplied by the number of items. The statistics show that the implementation of the *mean* and *mode* imputation techniques naturally results in a 100% rating matrix density. Meanwhile, the *null* imputation technique does not change the density of the matrix since it keeps each missing entry as it is, i.e., zero values. In other words, the *null* imputation technique keeps the data to remain sparse and will solely rely on the clustering technique to solve the sparsity issue.

Table 2. Comparison of rating matrix densities

Imputation Technique	Density (%)	
	Rating Matrix $\boldsymbol{R} \in \mathbb{R}^{\boldsymbol{m\times n}}$	Imputed Rating Matrix $\boldsymbol{\hat{R}} \in \mathbb{R}^{\boldsymbol{m\times n}}$
Null	5.04	5.04
Mean	5.04	100
Mode	5.04	100

Table 3. Imputed User Cluster-based CF Method

Method	F1-Score				
	@1	@5	@10	@15	@20
UCCF-Null	**1.560**	**3.929**	**5.871**	**6.879**	**7.471**
UCCF-Mean	1.210	2.292	3.966	4.594	4.909
UCCF-Mode	1.177	2.387	3.957	4.646	5.038

Table 4. Imputed Item Cluster-based CF Method

Method	F1-Score				
	@1	@5	@10	@15	@20
ICCF-Null	**1.520**	**5.229**	**7.588**	**8.898**	**9.671**
ICCF-Mean	0.507	1.924	2.928	3.424	3.685
ICCF-Mode	0.856	2.824	3.652	4.061	4.299

Table 3 and Table 4 respectively show the performance of the imputed *UCCF* and *ICCF* methods at various top-N. We can observe that the *UCCF-Null* always performs than the *UCCF-Mean* and *UCCF-Mode* at any top-N. We can also notice the same case with the *ICCF-Null* as it consistently outperforms the *ICCF-Mean* and *ICCF-Mode*. These results indicate that the *null* imputation technique can best deal with the missing rating entries compared to the *mean* and *mode* techniques. In other words, replacing the missing entries with other than zero values is not beneficial for the recommendation system since it will make the cluster-based method to misinterpret the data, unfavorably reduce the recommendation quality. In short, the implementation of the clustering technique is sufficient for solving the sparsity issue.

Note that from this on forward, we use *UCCF-Null* and *ICCF-Null* to represent the imputed cluster-based CF methods as they have shown to perform the best compared to the others.

Effect of number of clusters C and neighbor size D

We study the impact of the number of clusters C used when employing the K-Means clustering algorithm (see Fig. 4) to the imputed cluster-based CF methods. In this investigation, we implement a various number of cluster $C = \{2, 4, 6, 8, 10, 20, 30, 40, 50, \dots, 900\}$.

Fig. 5 and Fig. 6 show that the *UCCF-Null* and *ICCF-Null* respectively reach their best performances when C are 550 and 20. Given that $m = 983$ and $n = 1682$, we can assume that the average number of members in each cluster of *UCCF* is very few in compared to that of *ICCF* since $\left(\frac{983}{550}\right) \ll \left(\frac{1682}{20}\right)$.

The neighbor size D is used to calculate the rating predictions, i.e., Equation (2) and (4), that determine the top-N list of recommendations. In this experiment, we implement a various neighbor size $D = \{2, 5, 10, 20, 30, \dots, 100\}$.

Fig. 7 and Fig. 8 show that both the *UCCF-Null* and *ICCF-Null* achieve the best results when $D = 2$. The results indicate that we only need a very small number of D on the cluster-based CF methods. In the case of *UCCF-Null*, its performance is degrading when $D > 2$ and is saturating when $D \geq 10$. These findings indicate that the number of members of each cluster is within the range of 1 to 10. On the other hand, the performance of *ICCF-Null* is degrading when $D > 2$ and is saturating when $D \geq 20$. These findings indicate that the number of members of each cluster is within the range of 1 to 20.

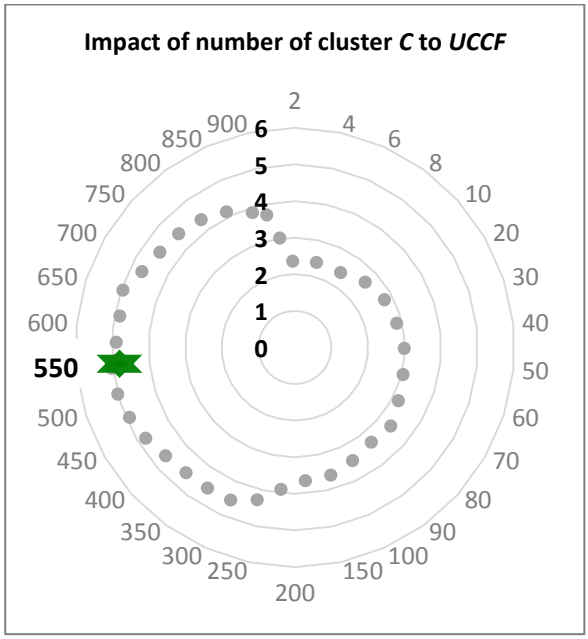

Fig. 5. Impact of number of cluster C to *UCCF*

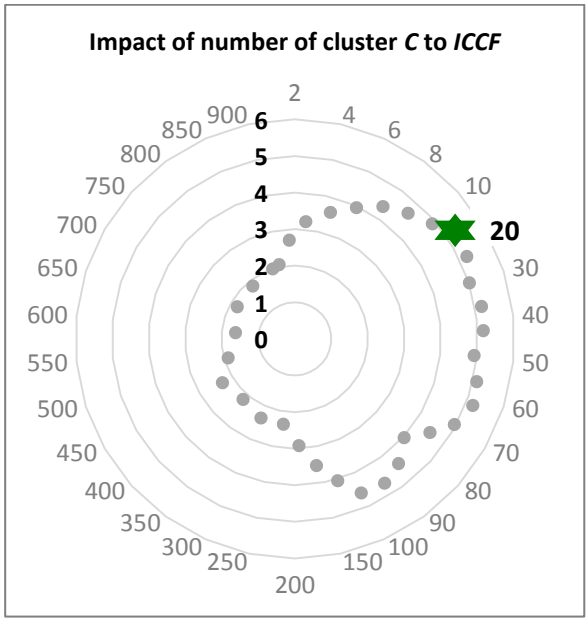

Fig. 6. Impact of number of cluster C to *ICCF*

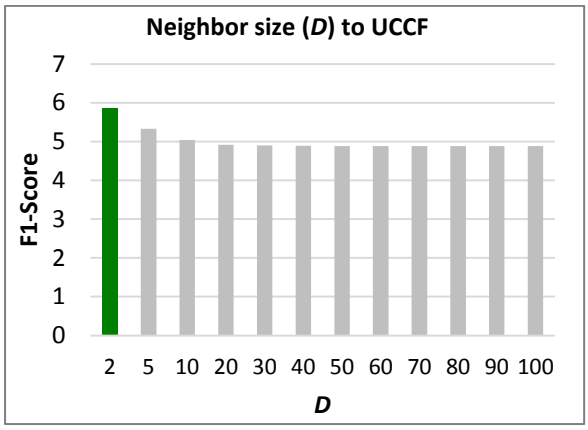

Fig. 7. Impact of neighbor size *D* to *UCCF*

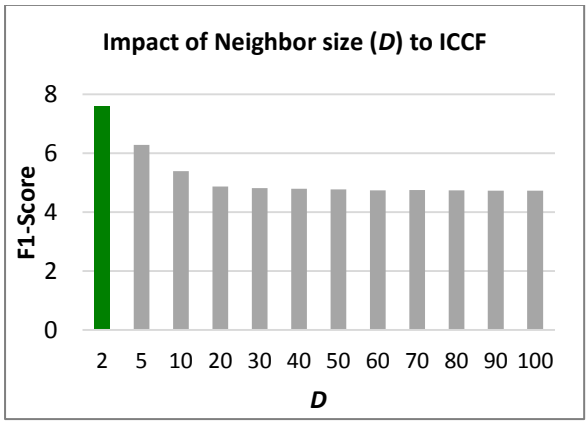

Fig. 8. Impact of neighbor size *D* to *ICCF*

Cluster-based VS Traditional CF methods

This sub-section benchmarks the performance of our *UCCF* and *ICCF* to their traditional counterparts, i.e., user-based (*UCF*) [17] and item-based (*ICF*) methods [3]. The neighbor sizes of *UCF* and *ICF* are experimentally fined-tuned as 2 and 150, respectively, to achieve their best performances.

Fig. 9 shows that *UCCF* and *ICCF* always significantly outperform the *UCF* and *ICF* at any top-*N* list of recommendations. The results confirm that the implementation of a cluster-based approach can improve the recommendation quality of traditional CF methods.

Additionally, it is also worthwhile to observe that *ICCF* outperforms *UCCF*. This finding confirms that the item-based model generally outperforms the user-based [2, 3].

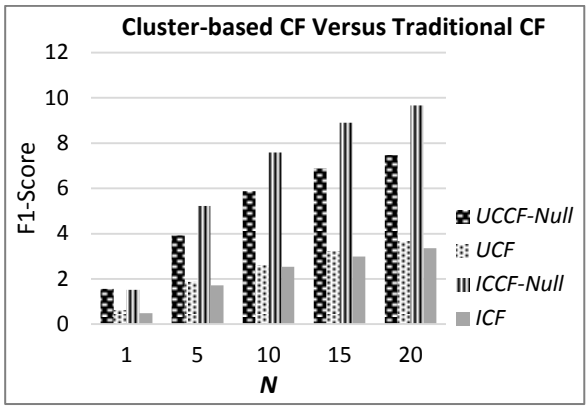

Fig. 9. Comparison of cluster-based and traditional CF methods

CONCLUSION

Our imputed cluster-based CF method implements a combination of an imputation technique and the cluster-based CF approach to deal with the missing rating entries, for generating the list of recommendations. This paper uses three variations of imputation techniques, i.e., *null*, *mean*, and *mode*. The cluster-based approach is employed by using the K-Means as the clustering technique, and either the user-based (*UCCF*) or the items-based (*ICCF*) model as the CF approach. The empirical analysis shows that the *null* imputation technique performs the best compared to the *mean* and *mode* techniques. This outcome suggests that the implementation of the clustering technique is sufficient for solving the sparsity issue such that imputing the missing entries is not necessary. The performance comparison reveals that our cluster-based methods always outperform the traditional CF methods, i.e., *UCF* and *ICF*. The results confirm that the implementation of a cluster-based approach can improve the recommendation quality of traditional CF methods.

REFERENCES

[1] J. A. Konstan and J. Riedl, "Recommender systems: from algorithms to user experience," *User Modeling and User-Adapted Interaction*, vol. 22, pp. 101-123, 2012.

[2] C. C. Aggarwal, *Recommender Systems: The Textbook*. Switzerland: Springer International Publishing, 2016.

[3] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-based Collaborative Filtering Recommendation Algorithms," in Proceeding of *The 10th International Conference on World Wide Web*, Hong Kong, pp. 285-295, 2001.

[4] G. Guo, J. Zhang, and D. Thalmann, "Merging trust in collaborative filtering to alleviate data sparsity and cold start," *Knowledge-Based Systems*, vol. 57, pp. 57-68, 2014.

[5] N. Ifada and R. Nayak, "How Relevant is the Irrelevant Data: Leveraging the Tagging Data for a Learning-to-Rank Model," in Proceeding of *The 19th ACM International Conference on Web Search and Data Mining*, San Francisco, California, US, pp. 23-32, 2016.

[6] N. P. Kumar and Z. Fan, "Hybrid User-Item Based Collaborative Filtering," *Procedia Computer Science*, vol. 60, pp. 1453-1461, 2015.

[7] H. Koohi and K. Kiani, "User based Collaborative Filtering using fuzzy C-means," *Measurement*, vol. 91, pp. 134-139, 2016.

[8] P. Phorasim and L. Yu, "Movies recommendation system using collaborative filtering and k-means," *International Journal of Advanced Computer Research*, vol. 7, pp. 52-59, 2017.

[9] N. Sun, Y. Zhuang, and S. Ni, "A Trust-Based Collaborative Filtering Algorithm Using a User Preference Clustering," *Management Science and Engineering*, vol. 11, pp. 9-19, 2017.

[10] J. Zhang, Y. Lin, M. Lin, and J. Liu, "An effective collaborative filtering algorithm based on user preference clustering," *Applied Intelligence*, vol. 45, pp. 230-240, 2016.

[11] G. Guo, J. Zhang, and N. Yorke-Smith, "Leveraging multiviews of trust and similarity to enhance clustering-based recommender systems," *Knowledge-Based Systems*, vol. 74, pp. 14-27, 2015.

[12] N. Ifada, E. H. Prasetyo, and Mula'ab, "Employing sparsity removal approach and Fuzzy C-Means clustering technique on a movie recommendation system," in Proceeding of *The 2018 International Conference on Computer Engineering, Network and Intelligent Multimedia (CENIM)*, Surabaya, Indonesia, pp. 157-162, 2018.

[13] M. Ranjbar, P. Moradi, M. Azami, and M. Jalili, "An imputation-based matrix factorization method for improving accuracy of collaborative filtering systems," *Engineering Applications of Artificial Intelligence*, vol. 46, pp. 58-66, 2015.

[14] X. Yuan, L. Han, S. Qian, G. Xu, and H. Yan, "Singular value decomposition based recommendation using imputed data," *Knowledge-Based Systems*, vol. 163, pp. 485-494, 2019.

[15] M. A. Ghazanfar and A. Prugel, "The advantage of careful imputation sources in sparse data-environment of recommender systems: Generating improved SVD-based recommendations," *Informatica*, vol. 37, pp. 61-92, 2013.

[16] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*, Third ed. Waltham, USA: Morgan Kaufmann, 2012.

[17] G. Adomavicius and A. Tuzhilin, *"*Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-art and Possible Extensions,*" IEEE Transactions on Knowledge and Data Engineering, vol. 17, pp. 734-749, 2005.*