An Item-Oriented Collaborative Filtering Algorithm for Recommender Systems

Sanjaya Kumar Panda[1], Manas Ranjan Senapati[2], *Fellow*, *IEI* and Pradip Kumar Sahu[3]
[1]*Department of Computer Science and Engineering*
National Institute of Technology Warangal, *Warangal* – 506004, *Telangana*, *India*
[2,3]*Department of Information Technology*
Veer Surendra Sai University of Technology, *Burla* – 768018, *Odisha*, *India*
{[1]sanjayauce, [2]manassena, [3]pradip.cet}@gmail.com

***Abstract*—Recommender systems (RSs) are one of the popular, common and widespread applications in today's business. It predicts the rating that a user is likely to give an item and finds a list of items to recommend the user. In this context, it is primarily categorized into collaborative filtering-based (CF) RS and content-based RS. In the CF, the RS suggests the items that are based on the behavior of similar users and its past behavior. One of the well-known algorithms in the CF is *k*-nearest neighbor (*k*NN), which relies on the behavior (rating) of *k* similar users (items) in oriented to users (items). However, it suffers from the unknown rating problem and the insufficient neighbor problem if the *k* similar users (items) have not rated the corresponding item (with respect to the corresponding user). Therefore, we develop an item-oriented CF (ICF) algorithm for RSs to overcome the problems associated with the *k*NN algorithm. The proposed algorithm ICF selects *k* similar items of a given item by finding the similarity count of the neighbors. We implement both the algorithms and compare with the *k*NN algorithm using four datasets to show the superiority of the proposed algorithm ICF in terms of two errors, namely mean absolute error (MAE), root mean square error (RMSE), precision, recall and *F*-score.**

***Index Terms*—Collaborative Filtering, Content-Based, Item Oriented, *k*-Nearest Neighbor, Recommender Systems.**

I. INTRODUCTION

Recommender Systems (RSs) are expanding rapidly in various aspects of electronic commerce companies and entertainment industries [1], [2]. It is one of the popular, common, significant and widespread applications in the recent years. It is also referred as an information filtering system that is used by Amazon, eBay, CDNOW, Levis, Reel.com, Netflix and YouTube. It is a multi-disciplinary area, which covers machine learning, data and web mining and information retrieval [2-4]. The goal of RS is to predict the rating that a user is likely to give an item and find a list of items to recommend the user in order to increase the sale of the electronic commerce companies and entertainment industries. For instance, Amazon [5] has reported a sale increase from \$9.9 billion to \$12.83 billion, i.e., an increase of 29% sales in a particular year.

The RS is primarily categorized into two types, namely collaborative filtering-based (CF) and content-based [1, 3, 4, 6]. In the CF, the RS considers user's past behavior and other similar users' behavior to suggest the items. Here, the user's past behavior is considered in terms of previously purchased items and the ratings given by the corresponding users to those items [4]. Note that the rating is usually given in the range of 1 to 5, where 1 represents the most dissatisfaction of the user and 5 represents the most satisfaction of the user, respectively. In the content-based, the RS considers the features or contents of the items, without using the user ratings and preferences. One of the well-known and popular algorithms in the CF recommender systems is *k*-nearest neighbor (*k*NN) [7-10]. This algorithm comes in two flavors, namely oriented to users and oriented to items. In the oriented to users (items), the rating is based on the behavior of *k* similar users (items) [8]. In this paper, we focus on oriented to items by using the concept of our earlier paper [11], which focuses on oriented to users. However, the *k*NN algorithm suffers two major problems. 1) If the *k* similar items of a given item are not rated by the corresponding user, then the *k*NN algorithm is incapable to predict the rating of the given item. We refer it by the unknown rating problem. 2) In a typical case, a given item may not have *k* neighbors. As a result, the *k*NN algorithm predicts the rating based on less than *k* neighbors, which may lead to biased rating. We refer this typical case as the insufficient neighbor problem.

In this paper, we present the following RS problem. Given a rating matrix *R* between *n* users and *m* items, the problem is to find the rating of the unpurchased items, i.e., oriented to items. For this, we develop an item-oriented CF (ICF) algorithm for RSs. The proposed algorithm ICF finds the similarity count matrix between items to determine the number of common items. Then it finds the neighbors of each unpurchased item with respect to the corresponding user. Finally, it selects the *k*-neighbor based on the top *k* similarity count values. As a result, it overcomes the two problems of the *k*NN algorithm. We implement both the proposed algorithm ICF and the well-known *k*NN algorithm in oriented to items. Further, we carry out the simulation runs of both the algorithms in sixteen instances of four different datasets and compare in terms of mean absolute error (MAE), root mean square error (RMSE), precision, recall and *F*-score [11]. The main contributions of this paper are as follows. 1) Development of a CF algorithm for RS to overcome unknown rating problem and insufficient neighbor problem. 2) Implement the proposed algorithm ICF and the *k*NN algorithm in oriented to items and validate them using training and testing datasets. 3) Simulation runs of the algorithms using four different datasets [11]. 4) Comparison of performance of the algorithms in five different metrics.

The paper is organized as follows. Section II discussed the related work; Section III presents the problem statement; Section IV details the ICF; Section V explains simulations and results, followed by Section VI that concludes this work.

II. Related Work

With the wide adoption of Internet shopping, RSs are used in many companies and industries. To suggest a few items from a large set of items, RSs have attracted great attention in academia and industry. The main challenge for RSs is to predict the rating of the unpurchased items and suggest some of them to the users based on high predicted ratings. Many recommendation algorithms [6-11] have been introduced by focusing the above challenges in CF and content-based RS. One of the well-known algorithm in CF is *k*NN algorithm, which is memory-based method that establishes the similarities between the users and the items. Hernando et al. [7] have proposed a memory-based method that focuses on the user-based nearest neighborhood algorithm. Here, the users are selected by considering the coincidence ratio between the votes in the database. Hernando et al. [8] have introduced item style based recommendation and visualized in the form of trees. They have shown the reliability of the recommendation and additional information to the users in order to select the appropriate items. Nayak et al. [11] have presented a user-oriented CF (UCF) algorithm, which is based on the similarity count value of k users. However, they have not considered oriented to items in their algorithm.

The algorithm proposed in this paper is an extension of our earlier paper [11] and different from the *k*NN algorithm and our earlier algorithm, called UCF [11] in the following aspects. 1) The *k*NN algorithm finds the k-neighbor items based on the mean squared deviation (similarity score), whereas the proposed algorithm ICF finds the k-neighbor items based on the similarity count of the items with respect to the user. 2) The UCF algorithm is based on oriented to users, whereas the proposed algorithm ICF is based on oriented to items. 3) The proposed algorithm ICF identifies and solves the insufficient neighbor problem in addition to the unknown rating problem as identified in [11]. However, the *k*NN algorithm suffers from the above two problems. 4) The k-neighbor items of a given item are same in the *k*NN algorithm irrespective of the user, whereas it is different in the proposed algorithm ICF. The simulation results and analysis show that the superiority of the proposed algorithm over the oriented to items of the *k*NN algorithm in terms of five performance measures.

III. Problem Statement

Consider a set of users, n, a set of items, m, $m >> n$ and a rating matrix R of size $n \times m$ (see Eq. (1) [11]). An element (or a rating or a score) of the rating matrix R, R_{ij} represents the feedback value of a user U_i in a purchased item I_j. Here, we consider the feedback value in between 1 and 5. Note that 1 represents the most dissatisfaction and 5 represents the most satisfaction of a user. Moreover, we use 0 (or blank) to represent the unpurchased or unrated item.

The problem is to determine the probable rating of the unpurchased items, such that the following objectives are fulfilled. 1) Objective 1: Minimize MAE and RMSE 2) Objective 2: Maximize F-score (i.e., trade-off between precision and recall).

$$R = \begin{array}{c c} & \begin{array}{cccc} I_1 & I_2 & \cdots & I_m \end{array} \\ \begin{array}{c} U_1 \\ U_2 \\ \vdots \\ U_n \end{array} & \begin{bmatrix} R_{11} & R_{12} & \cdots & R_{1m} \\ R_{21} & R_{22} & \cdots & R_{2m} \\ \vdots & \vdots & \cdots & \vdots \\ R_{n1} & R_{n2} & \cdots & R_{nm} \end{bmatrix} \end{array} \quad (1)$$

IV. Proposed Algorithm

The item-oriented collaborative filtering (ICF) is an extension of our earlier work [11]. It is based on memory and especially designed for collaborative filtering-based recommender systems. The foremost goal of the proposed algorithm ICF is to find the ratings of the unpurchased items of the users. Note that the rating is purely based on oriented to items. This rating may be used by various companies and industries to suggest items, products, audios, videos and movies to their users.

The basic idea of the proposed algorithm ICF is as follows. The algorithm finds the similarity count (*SC*) matrix by taking each pair of items. Here, an element (or a value) of similarity count matrix indicates the total number of users that are rated both the items. Alternatively, it is the total number of common users for a pair of items. Moreover, an element of *SC* matrix is set to zero if and only if the same set of items is considered. Next, ICF finds the possible neighbors of the items with respect to the corresponding users and selects the k-neighbor items by considering the top k similarity count values. If there is a tie in the selection, then it can be broken by following any order. Finally, it aggregates the ratings of the k-neighbor items and predicts the ratings of the unpurchased items. Here, each item may have a different set of neighbors with respect to the users.

A. Algorithm

The pseudo code of the proposed algorithm ICF is shown in Fig. 1. Initially, the proposed algorithm ICF takes a rating matrix R of size $n \times m$ (Line 1 of Fig. 1). Here, n denotes the number of users and m denotes the number of items, respectively. Next, it initializes the similarity count matrix *SC* of size $m \times m$ and sets the values to zero (Line 2). Then it determines the k value and it is greater than or equal to 2 (Line 3). Note that the optimal value of k is beyond the scope of this paper. Next, the proposed algorithm ICF finds the similarity count value between each pair of items (Line 4-18). The similarity count value of two equal items is set to zero (Line 6-7). Similarly, the similarity count value of two different items with no common users is also set to zero (Line 9). On the other hand, the similarity count value of two different items with common users is placed in the *SC* matrix (Line 10-15). Next, the proposed algorithm ICF finds the ratings of the unpurchased items (Line 19-26). For this, it finds the possible neighbors of the items with respect to the users and selects the k-neighbor items based on top k similarity count values (Line 22). If there is a tie in the selection of neighbors, then it can be broken randomly by following any order. Finally, the proposed algorithm ICF aggregates the ratings of the k-neighbor and predicts the ratings of the unpurchased items (Line 23). The predicted ratings are stored in another matrix R'.

B. Illustration

Let us illustrate the proposed algorithm ICF using a rating matrix of size 6 × 10 as shown in Table I. The value of k is set to 2 (say). The proposed algorithm ICF finds the similarity count matrix SC from the rating matrix (Table I). The resultant SC matrix is shown in Table II. For example, the similarity count value of item I_1 and item I_2 is 2 as there are two common users (i.e., U_1 and U_2) and the similarity count value of item I_7 and item I_9 is 4 as there are four common users (i.e., U_2, U_4, U_5 and U_6). Note that hyphen symbol ("-") in Table II denotes that there is no common users between the items. For example, there is no common users between item I_5 and item I_6, item I_5 and item I_{10} and vice-versa.

Algorithm: Item-Oriented CF (ICF)

Input: 1) A rating matrix R of size $n \times m$
2) A set of n users
3) A set of m items
4) The value of k

Output: 1) A set of ratings, based on oriented to users

```
Load the rating matrix R of size n × m
Initialize the similarity count matrix SC of size m × m
Set the value of k
for i = 1, 2, 3,…, m
    for j = 1, 2, 3,…, m
        if i = j
            SC(i, j) = 0
        else
            scount = 0
            for l = 1, 2, 3,…, n
                if R(i, l) > 0 and R(j, l) > 0
                    scount = scount + 1
                endif
            endfor
            SC(i, j) = scount
        endif
    endfor
endfor
for i = 1, 2, 3,…, n
    for l = 1, 2, 3,…, m
        if R(i, l) = 0
            Find the possible neighbors (items) of item I_l with respect to user U_i and select the k-neighbor items based on the top k similarity count values (If there is a tie in the selection, then it can be broken randomly by following any order)
            Aggregate the ratings of the k-neighbor items and assign to another matrix R'(i, l)
        endif
    endfor
endfor
```

Fig. 1. Pseudo code for ICF algorithm.

TABLE I. A RATING MATRIX WITH 6 USERS AND 10 ITEMS [11]

R	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1	5	4	5		3			1		
U_2	3	4	3	4	5		3		4	
U_3		2	3	3	5					
U_4		5		2		4	4	5	4	5
U_5	2					5	5	4	3	5
U_6			3			3	4	5	5	5

TABLE II. A SIMILARITY COUNT MATRIX USING ORIENTED TO ITEMS (K = 2)

SC	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
I_1	0	2	2	1	2	1	2	2	2	1
I_2	2	0	3	3	3	1	2	2	2	1
I_3	2	3	0	2	3	1	2	2	2	1
I_4	1	3	2	0	2	1	2	1	2	1
I_5	2	3	3	2	0	-	1	1	1	-
I_6	1	1	1	1	-	0	3	3	3	3
I_7	2	2	2	2	1	3	0	3	4	3
I_8	2	2	2	1	1	3	3	0	3	3
I_9	2	2	2	2	1	3	4	3	0	3
I_{10}	1	1	1	1	-	3	3	3	3	0

TABLE III. A PROBABLE RATING MATRIX USING ICF ALGORITHM

R'	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1	-	-	-	4.50	-	3.00	3.00	-	3.00	3.00
2-NN	-	-	-	I_2, I_3	-	I_1, I_8	I_1, I_8	-	I_1, I_8	I_1, I_8
U_2	-	-	-	-	-	3.50	-	3.50	-	3.50
2-NN	-	-	-	-	-	I_7, I_9	-	I_7, I_9	-	I_7, I_9
U_3	2.50	-	-	-	-	2.50	2.50	2.50	2.50	2.50
2-NN	I_2, I_3	-	-	-	-	I_2, I_3	I_2, I_3	I_2, I_3	I_2, I_3	I_2, I_3
U_4	4.50	-	3.50	-	3.50	-	-	-	-	-
2-NN	I_2, I_7	-	I_2, I_4	-	I_2, I_4	-	-	-	-	-
U_5	-	3.50	3.50	4.00	3.50	-	-	-	-	-
2-NN	-	I_1, I_7	I_1, I_7	I_7, I_9	I_1, I_7	-	-	-	-	-
U_6	3.50	3.50	-	3.50	3.50	-	-	-	-	-
2-NN	I_3, I_7	I_3, I_7	-	I_3, I_7	I_3, I_7	-	-	-	-	-

Next, the proposed algorithm ICF finds the possible neighbors of the unpurchased items and selects k-neighbor items of the unpurchased items with respect to the users. The 2-neighbor of the unpurchased items is shown in odd rows of Table III (except the first row). For example, the possible neighbors of the unpurchased item I_4 with respect to the user U_1 is items I_1, I_2, I_3, I_5 and I_8, respectively, and their similarity count is 1, 3, 2, 2 and 1, respectively (Table II). However, the 2-neighbor of item I_4 is item I_2 and item I_3, respectively, as these two items retain the top 2 similarity count values. Note that there is a tie between the similarity count value of item I_3 and item I_5 and it is broken randomly by considering any order. The rating of these items (i.e., I_2 and I_3) is 4 and 5, respectively. Therefore, the probable rating of item I_4 is $\frac{4+5}{2}$ = 4.50. Similarly, the possible neighbors of the unpurchased item I_1 with respect to the user U_3 is items I_2, I_3, I_4 and I_5, respectively, and their similarity count is 2, 2, 1 and 2, respectively. However, the 2-neighbor of item I_1 is item I_2 and item I_3, respectively. The rating of these items is 2 and 3, respectively. Therefore, the probable rating of item I_1 is $\frac{2+3}{2}$ = 2.50.

For the sake of comparison, we also illustrate the existing algorithm kNN using the same rating matrix as shown in Table I. The kNN algorithm finds the similarity value between each pair of items using mean squared deviation (MSD) function. The resultant similarity matrix S is shown in Table IV. Like ICF, the similarity value between the same items is set to zero. Next, the kNN algorithm determines the k-neighbor items. It is represented in bold color (column-wise). For example, the closest neighbor of item I_1 is items I_2, I_3, I_4 and I_9, respectively, and their MSD is 1.00, 0.00, 1.00 and 1.00, respectively. As there is a tie among the items I_2, I_4 and I_9, the kNN algorithm selects item I_2 randomly. Therefore, the probable rating of the item I_1 with respect to user U_3 is $\frac{2+3}{2}$ = 2.50, which is the average rating of item I_2 and item I_3. The probable rating matrix R' is shown in Table

V. Here, bold color indicates that the probable rating is based on a single neighbor.

It is noteworthy to mention that the *k*NN algorithm is incapable to find the rating of the item I_8 with respect to user U_3. This is due to the following notable reason. The 2-neighbor of item I_8 is item I_9 and item I_{10}, respectively. Therefore, the probable rating of item I_8 is the average rating of these two items. However, these items are not rated by user U_3. On the other hand, most of the probable ratings are determined using one and only one neighbor. As a result, the *k*NN algorithm is incapable to provide the approximate ratings. Unlike the *k*NN algorithm, the ICF algorithm is able to find the rating of item I_8 with respect to user U_3, i.e., 2.50. It shows the superiority of the proposed algorithm ICF over the *k*NN algorithm in the rating matrix as shown in Table I.

TABLE IV. A SIMILARITY MATRIX USING ORIENTED TO ITEMS ($K = 2$)

S	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
I_1	0.00	1.00	**0.00**	**1.00**	4.00	9.00	4.50	10.00	1.00	9.00
I_2	**1.00**	0.00	1.00	3.33	3.67	1.00	1.00	4.50	**0.50**	**0.00**
I_3	**0.00**	1.00	0.00	**0.50**	4.00	**0.00**	**0.50**	10.00	2.50	4.00
I_4	**1.00**	3.33	0.50	0.00	**2.50**	4.00	2.50	9.00	2.00	9.00
I_5	4.00	3.67	4.00	2.50	0.00	-	4.00	4.00	1.00	-
I_6	9.00	1.00	**0.00**	4.00	-	0.00	**0.33**	2.00	2.67	1.67
I_7	4.50	1.00	0.50	2.50	4.00	**0.33**	0.00	1.00	1.50	0.67
I_8	10.00	4.50	10.00	9.00	4.00	2.00	1.00	0.00	**0.67**	**0.33**
I_9	**1.00**	**0.50**	2.50	2.00	**1.00**	2.67	1.50	**0.67**	0.00	1.67
I_{10}	9.00	**0.00**	4.00	9.00	-	1.67	0.67	**0.33**	1.67	0.00

TABLE V. A PROBABLE RATING MATRIX USING KNN ALGORITHM

R'	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1	-	-	-	5.00	-	**5.00**	**5.00**	-	2.50	2.50
U_2	-	-	-	-	-	3.00	-	**4.00**	-	**4.00**
U_3	2.50	-	-	-	-	**3.00**	**3.00**	**?**	**2.00**	**2.00**
U_4	**5.00**	-	**4.00**	-	3.00	-	-	-	-	-
U_5	-	4.00	3.50	**2.00**	**3.00**	-	-	-	-	-
U_6	**3.00**	5.00	-	**3.00**	**5.00**	-	-	-	-	-
2-NN	I_2, I_3	I_9, I_{10}	I_1, I_6	I_1, I_3	I_4, I_9	I_3, I_7	I_3, I_6	I_9, I_{10}	I_2, I_8	I_2, I_8

In order to further compare the proposed algorithm ICF and the existing algorithm *k*NN, we randomly remove some of the ratings from Table I and Table III (proposed), and Table I and Table V (existing), and the rest of the ratings are placed in Table VI and Table VII, respectively. These are referred as the training dataset for the proposed algorithm ICF and the existing algorithm *k*NN, respectively. On the other hand, the removed ratings are placed in Table VIII and Table IX, respectively. We call these matrix as the testing dataset for the proposed algorithm ICF and the existing algorithm *k*NN, respectively. It is noteworthy to mention that we have not followed the ratio between training dataset and testing dataset for the sake of illustration, which is usually 80:20. Moreover, we have not considered the unknown item rating (i.e., item I_8) of the *k*NN algorithm in the testing dataset (Table VII) to avoid biased result.

Next, we apply the training dataset of the proposed algorithm ICF and the existing algorithm *k*NN to find the probable ratings. The probable rating matrix of these algorithms is presented in Table X and Table XI, respectively. Now we compare the performance of both algorithms in terms of MAE, RMSE, precision, recall, *F*-score and the total number of unknown ratings *UN* as shown in Table XII. This table shows the better performance of the proposed algorithm over the *k*NN algorithm in terms of six performance metrics. The description of the performance metrics is shown in our earlier paper [11].

TABLE VI. A TRAINING MATRIX OF ICF ALGORITHM

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1	5.00	4.00		4.50	3.00		3.00			3.00
U_2		4.00	3.00	4.00	5.00	3.50			4.00	3.50
U_3	4.00	2.00	3.00	3.00	5.00		3.00		3.00	
U_4	5.00		5.00	2.00	3.50	4.00	4.00	5.00	4.00	5.00
U_5	2.00	4.50		4.00				4.00		5.00
U_6	5.00	4.00	3.00	4.50	4.00	3.00		5.00	5.00	5.00

TABLE VII. A TRAINING MATRIX OF KNN ALGORITHM

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1	5.00	4.00		5.00	3.00		5.00			2.50
U_2		4.00	3.00	4.00	5.00	3.00			4.00	4.00
U_3	2.50	2.00	3.00	3.00	5.00		3.00	?	2.00	
U_4	5.00		4.00	2.00	3.00	4.00	4.00	5.00	4.00	5.00
U_5	2.00	4.00		2.00				4.00		5.00
U_6	3.00	5.00	3.00	3.00	5.00	3.00		5.00	5.00	5.00

TABLE VIII. A TESTING MATRIX OF ICF ALGORITHM

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1			5.00			3.00		1.00	3.00	
U_2	3.00						3.00	3.50		
U_3						3.00		2.50		3.00
U_4		5.00								
U_5			4.50		3.50	5.00	5.00		3.00	
U_6							4.00			

TABLE IX. A TESTING MATRIX OF KNN ALGORITHM

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1			5.00			5.00		1.00	2.50	
U_2	3.00						3.00	4.00		
U_3						3.00				2.00
U_4		5.00								
U_5			3.50		3.00	5.00	5.00		3.00	
U_6							4.00			

TABLE X. A PROBABLE RATING MATRIX OF ICF ALGORITHM

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1			3.75			3.75		4.75	3.75	
U_2	4.50						4.50	3.75		
U_3								3.50		4.00
U_4		3.50								
U_5			4.50		4.50	4.50	3.00		4.50	
U_6							4.75			

TABLE XI. A PROBABLE RATING MATRIX OF KNN ALGORITHM

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1			5.00			5.00		3.25	3.25	
U_2	3.00						3.00	4.00		
U_3						3.00		2.00		2.00
U_4		4.50								
U_5			?		4.50	?	?		4.50	
U_6							3.00			

TABLE XII. A DETAILED COMPARISON OF THE PROPOSED ALGORITHM ICF AND THE *k*NN ALGORITHM

Algorithm	*k*NN	Proposed
MAE	1.5000	1.1765
RMSE	2.4689	1.4272
Precision	0.8333	0.8824
Recall	0.7692	1.0000
***F*-score**	0.8000	0.9375
Number of unknown ratings	3	0

V. SIMULATION AND RESULTS

The simulations of the proposed algorithm ICF and the existing algorithm *k*NN were processed in a system with following configuration and environment. 1) The system contains an Intel Core *i*3-2330M CPU processor with 2.20 GHz and 2.20 GHz frequency. 2) It also contains 4 GB internal memory and Windows 7 professional 64-bit operating system. 3) The system is loaded with MATLAB

R2014a, version 8.3.0.532 [12-17]. The following parameter settings were carried out to take a snapshot of the results. 1) The value of k is set to 2 for both the algorithms. 2) The threshold value is set to 2.5 for determining precision and recall. The optimal value of these parameters is beyond the scope of this paper.

We took datasets of our earlier paper [11] to carry out the simulation runs. There are four datasets and each dataset contains four instances. The structure of the instance is stated as follows. 1) Instance id (ix) 2) The total number of users (n) and 3) The total number of items (m). For example, i4_128_200 denotes the instance id as 4, n as 128 and m as 200.

First, we compare the proposed algorithm ICF and the kNN algorithm in terms of MAE and RMSE using four different datasets as shown in Table XIII. The graphical representation of the same is also shown in Fig. 2. Note that the results are shown by averaging the four instances of each dataset. It is clear from Table XIII and Fig. 2 that the proposed algorithm ICF minimizes both the errors than the kNN algorithm. The rationality behind this is that the proposed algorithm ICF selects the k-neighbor items based on the similarity count matrix. As a result, it overcomes the problems associated with the kNN algorithm.

TABLE XIII. COMPARISON OF MAE AND RMSE FOR THE PROPOSED ALGORITHM ICF AND THE kNN ALGORITHM

Algorithm	**ICF**		***k*NN**	
Instance	**MAE**	**RMSE**	**MAE**	**RMSE**
***i*1_128_200**	1.1415	1.4278	1.2672	1.6977
***i*2_128_200**	1.1351	1.4130	1.3090	1.7520
***i*3_128_200**	1.1443	1.4370	1.3153	1.7856
***i*4_128_200**	1.1453	1.4237	1.3018	1.7468
***i*1_256_350**	1.1665	1.4415	1.3550	1.7857
***i*2_256_350**	1.1385	1.4181	1.3465	1.7987
***i*3_256_350**	1.1545	1.4299	1.2805	1.7276
***i*4_256_350**	1.1338	1.4089	1.3096	1.7311
***i*1_384_500**	1.1426	1.4137	1.3094	1.7483
***i*2_384_500**	1.1442	1.4181	1.3213	1.7566
***i*3_384_500**	1.1376	1.4126	1.3125	1.7518
***i*4_384_500**	1.1540	1.4287	1.3082	1.7520
***i*1_512_650**	1.1420	1.4119	1.3295	1.7640
***i*2_512_650**	1.1475	1.4173	1.3243	1.7635
***i*3_512_650**	1.1413	1.4135	1.3107	1.7449
***i*4_512_650**	1.1277	1.4000	1.3039	1.7392

Next, we compare both the algorithms in terms of precision, recall and F-score as shown in Table XIV and Fig. 3. As seen from this table and figure, the proposed algorithm ICF outperforms the existing algorithm kNN in terms of F-score and achieves a trade-off between precision and recall.

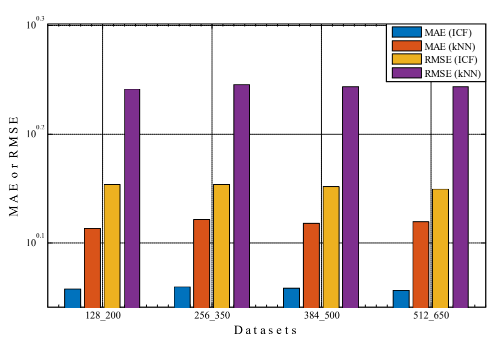

Fig. 2. Graphical comparison of MAE and RMSE.

TABLE XIV. COMPARISON OF PRECISION, RECALL AND *F*-SCORE FOR THE PROPOSED ALGORITHM ICF AND THE kNN ALGORITHM

Algorithm	**ICF**			***k*NN**		
Instance	**Precision**	**Recall**	***F*-score**	**Precision**	**Recall**	***F*-score**
***i*1_128_200**	0.7478	0.8882	0.8120	0.8100	0.7015	0.7518
***i*2_128_200**	0.7365	0.8619	0.7943	0.8112	0.7028	0.7531
***i*3_128_200**	0.7481	0.8755	0.8068	0.8246	0.6925	0.7528
***i*4_128_200**	0.7476	0.8555	0.7979	0.8081	0.7040	0.7525
***i*1_256_350**	0.7306	0.8325	0.7782	0.7985	0.6843	0.7370
***i*2_256_350**	0.7447	0.8600	0.7982	0.8043	0.6907	0.7432
***i*3_256_350**	0.7401	0.8383	0.7861	0.8123	0.7027	0.7535
***i*4_256_350**	0.7405	0.8574	0.7947	0.8006	0.6943	0.7437
***i*1_384_500**	0.7322	0.8396	0.7822	0.7976	0.6967	0.7437
***i*2_384_500**	0.7343	0.8328	0.7804	0.7977	0.6956	0.7431
***i*3_384_500**	0.7447	0.8613	0.7988	0.8039	0.7024	0.7497
***i*4_384_500**	0.7224	0.8414	0.7774	0.7973	0.7004	0.7458
***i*1_512_650**	0.7343	0.8427	0.7848	0.7967	0.6969	0.7435
***i*2_512_650**	0.7318	0.8296	0.7776	0.7982	0.6948	0.7429
***i*3_512_650**	0.7310	0.8377	0.7807	0.8004	0.6985	0.7460
***i*4_512_650**	0.7279	0.8306	0.7759	0.8026	0.7014	0.7486

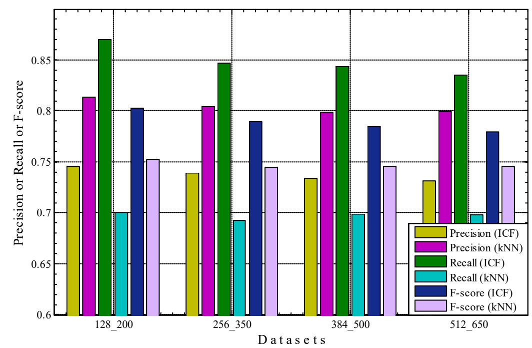

Fig. 3. Graphical comparison of precision, recall and F-score.

VI. CONCLUSION

In this paper, we have developed an item-oriented collaborative filtering (ICF) algorithm for recommender systems. The algorithm is an extension of our earlier paper [11]. The algorithm selects k-neighbor items by looking the similarity count with other items. The algorithm resolves two problems, namely the unknown rating and the insufficient neighbor that is associated with the well-known kNN algorithm. It was implemented and tested rigorously on sixteen instances of four different sizes of datasets. The simulation results have been produced in terms of MAE, RMSE, precision, recall and F-score, and compared with the kNN algorithm. The results show the superiority of the proposed algorithm ICF over the existing algorithm kNN.

In our future work, we will consider different types of items, such as consumable and non-consumable, and an item validity matrix for consumable items to enhance the capability of the proposed algorithm. On the other hand, our future effort is aimed to assign weightage to the items, which may be based on the popularity of items and to classify the users (i.e., regular user and non-regular user) in order to develop efficient algorithms.

ACKNOWLEDGMENT

The first version of this paper has appeared in proceedings of 5th *International Conference on Parallel*, *Distributed and Grid Computing* (*PDGC* 2018), *IEEE*. We would like to express our sincere thanks to all anonymous reviewers for their valuable comments and research directions, which significantly help us to extend this paper.

REFERENCES

[1] M. Kunaver and T. Pozri, "Diversity in Recommender Systems - A Survey", Knowledge-Based Systems, Elsevier, Vol. 123, pp. 154-162, 2017.

[2] R. Bodily and K. Verbert, "Review of Research on Student-Facing Learning Analytics Dashboards and Educational Recommender System", IEEE Transactions on Learning Technologies, Vol. 10, Issue 4, pp. 1-14, 405-418, 2017.

[3] D. Kotkov, S. Wang and J. Veijalainen, "A Survey of Serendipity in Recommender Systems", Knowledge-Based Systems, Elsevier, Vol. 111, pp. 180-192, 2016.

[4] J. Bobadilla, F. Ortega, A. Hernando and A. Gutierrez, "Recommender System Survey", Knowledge-Based Systems, Elsevier, Vol. 46, pp. 109–132, 2013.

[5] Amazon's Recommendation Secret, http://fortune.com/2012/07/30/amazons-recommendation-secret/, Accessed on 15th November 2018.

[6] J. Bobadilla, A. Hernando, F. Ortega and J. Bernal, "A Framework for Collaborative Filtering Recommender Systems", Expert Systems with Applications, Elsevier, Vol. 38, pp. 14609-14623, 2011.

[7] A. Hernando, J. Bobadilla, F. Ortega and J. Tejedor, "Incorporating Reliability Measurements into the Predictions of a Recommender System", Information Sciences, Elsevier, Vol. 218, pp. 1-16, 2013.

[8] A. Hernando, J. Bobadilla, F. Ortega and A. Gutierrez, "Trees for Explaining Recommendations Made Through Collaborative Filtering", Information Sciences, Elsevier, Vol. 239, pp. 1-17, 2013.

[9] V. Subramaniyaswamy and R. Logesh, "Adaptive KNN Based Recommender System Through Mining of User Preferences", Wireless Personal Communications, Springer, Vol. 97, Issue 2, pp. 2229-2247, 2017.

[10] D. Adeniyi, Z. Wei and Y. Yongquan, "Automated Web Usage Data Mining and Recommendation System Using K-Nearest Neighbor (KNN) Classification Method", Applied Computing and Informatics, Elsevier, Vol. 12, Issue 1, pp. 90-108, 2016.

[11] S. Nayak and S. Panda, "A User-Oriented Collaborative Filtering Algorithm for Recommender Systems", *Fifth International Conference on Parallel, Distributed and Grid Computing*, *IEEE*, pp. 1-7, 2018.

[12] M. Kumar, I. Gupta, S. Panda and P. Jana, "Granularity-Based Workflow Scheduling Algorithm for Cloud Computing", The Journal of Supercomputing, Springer, Vol. 73, pp. 5440-5464, 2017.

[13] S. Panda, S. Nanda and S. Bhoi, "A Pair-Based Task Scheduling Algorithm for Cloud Computing Environment", Journal of King Saud University – Computer and Information Sciences, Elsevier, pp. 1-12, 2018.

[14] S. Pande, S. Panda and S. Das, "A Customer-Oriented Task Scheduling for Heterogeneous Multi-Cloud Environment", International Journal of Cloud Applications and Computing, Vol. 6, Issue 4, pp. 1-17, 2016.

[15] S. Panda, I. Gupta and P. Jana, "Task Scheduling Algorithms for Multi-Cloud Systems: Allocation-Aware Approach", Information Systems Frontiers, Springer, pp. 1-19, 2017.

[16] S. Panda, S. Pande and S. Das, "Task Partitioning Scheduling Algorithms for Heterogeneous Multi-Cloud Environment", Arabian Journal for Science and Engineering, Springer, Vol. 43, Issue 2, pp. 913-933, 2018.

[17] S. Panda and P. Jana, "SLA-Based Task Scheduling Algorithms for Heterogeneous Multi-Cloud Environment", The Journal of Supercomputing, Springer, Vol. 73, pp. 2730-2762, 2017.