Knowledge and Information Systems (2022) 64:665–701
https://doi.org/10.1007/s10115-021-01651-8

REGULAR PAPER

An improved item-based collaborative filtering using a modified Bhattacharyya coefficient and user–user similarity as weight

Pradeep Kumar Singh[1] · **Shreyashee Sinha**[2] · **Prasenjit Choudhury**[2]

Received: 2 February 2021 / Revised: 18 December 2021 / Accepted: 26 December 2021 /
Published online: 25 January 2022

Abstract

Item-based filtering technique is a collaborative filtering algorithm for recommendations. Correlation-based similarity measures such as cosine similarity, Pearson correlation, and its variants have inherent limitations on sparse datasets because items may not have enough ratings for predictions. In addition, traditional similarity measures mainly focus on the orientations of the rating vectors, not magnitude, and as a result two rating vectors with different magnitudes but oriented in the same direction, can be exactly similar. Another aspect is that on a set of items, similar users' may have different rating pattern. In addition, to calculate the similarity between items, ratings of all co-rated users are considered; however, a judicious approach is to consider the similarity between users as a weight to find the similar neighbors of a target item. To mitigate these issues, a modified Bhattacharyya coefficient is proposed in this paper. The proposed similarity measure is used to calculate user–user similarity, which in turn is used as a weight in item-based collaborative filtering. The experimental analysis on the collected MovieLens datasets shows a significant improvement of item-based collaborative filtering, when user–user similarity calculated by the proposed modified similarity measure is used as a weight.

Keywords Recommender system · Collaborative filtering · Similarity function · Similarity measure · Top-N

✉ Pradeep Kumar Singh
pradeepsingh.gla@gla.ac.in; pks.14ca1103@phd.nitdgp.ac.in

Shreyashee Sinha
shreyashee.sinha@gmail.com

Prasenjit Choudhury
prasenjit0007@yahoo.co.in

[1] Department of Computer Engineering and Application, GLA University, Mathura, Uttar Pradesh, India

[2] Department of Computer Science and Engineering, National Institute of Technology, Durgapur, West Bengal, India

1 Introduction

Our society has entered the era of big data with the advent of internet and its applications in various domains [1]. The volume, velocity, and variety of data produced on a daily basis lead to excessive information, which is commonly known as information overload. Recommender system (RS) is used as an information retrieval tool in different domains (such as e-government, e-library, e-tourism, and e-commerce) to address the problem of information overloading. Several techniques such as content-based filtering, collaborative filtering (CF), and hybrid filtering are used to recommend items to active user. CF, which was first suggested by Goldberg in 1992, is the most popular and commonly used techniques in various RSs [2]. In fact, CF techniques produced the highest volume of research papers collected from established journals in the 2013–2014 period [3]. CF can be categorized into two types: model-based and memory-based. The model-based techniques employ the use of ML and data mining for generating recommendations, while the memory-based technique uses the user rating data to compute the similarity between users and/or items. Model-based CF were introduced to overcome the cons of memory-based CF; for instance, its limitation on sparse datasets is one of the most fundamental problems and limited research has addressed this issue.

Model-based approaches although lucrative come with its own set of implementational challenges. ML approaches are preferred when there is need to decrypt a black-box setting, i.e., when the unknown mapping function that generates a known input and output is to be recognized. However, in case of rating-based RSs one is provided with a rating and is aware of the function, whose basic ideology is as simple as "I like what my friends like." Hence, memory-based approaches with their fundamental limitation in sparse dataset resolved shall be highly beneficial to this scenario since the complexity of the RSs is minimal. Further, ML approaches would require extensive feature collection and dimension reduction. In case of limited datasets, a representative cluster may turn out to be an outlier and fail to truly represent the population. Further, the benchmark data on which model-based methods are trained on often differ from real-world settings, thus leading to a performance degradation. These arguments direct the motivation of research toward memory-based CF. Moreover, the simplicity of implementation, minimal content requirement, and ability to handle cold starts and to scale well with correlated items are other contributing factors that add to the advantages.

The existing memory-based CF technique can be classified into user-based and item-based CF. The concept of user-based CF is to find users who share appreciation in a group. If the same or almost the same rated items is common between two users, then users may have similar taste. These users create a community or a neighborhood. Intrinsic and extrinsic are the two methods, utilized for rating collection [4]. These collected ratings are used in the form of matrix of size m×n, where m represents a set of users, i.e., $U = \{u_1, u_2, u_3,..., u_m\}$, and n shows a set of items, i.e., $I = \{i_1, i_2, i_3,..., i_n\}$ that are rated by user set U.

More precisely, the basic principle behind user-based CF is that users with similar preferences in the past may have similar preferences in the future. Consequently, the most critical aspect of CF for enhancing recommendation quality is to find the nearest neighbors. The similarity method therefore has a significant impact in the performance of CF. Various similarity measures (SMs) are used to find the nearest neighbors or similar users of a target user [1,3]. The respective ratings and similarity values of nearest neighbors are used to predict the taste of the target user on different items. Top-N recommendation list is generated based on the above philosophy [3,5].

User-based CF faced some major limitations such as scalability, cold start, and sparsity [6,7]. The number of customers in any E-commerce site is increasing rapidly as compared to items and due to this, computing similarities between each pair of user became quite expensive . Item-based CF, which is introduced by members of GroupLens Research Group [8], is a more feasible approach for item recommendation. The basic assumption of item-based CF is that user prefers similar items that he or she liked in the past. Item-based CF utilizes the similarity values of items for predicting the target item [9]. The item-based CF has many advantages over user-based CF. Firstly, items are relatively static and very less in numbers, compared to the number of users, so similarity can be computed offline and can be accessed when needed [10]. Secondly, it provides more accurate recommendation than user-based CF [8,11,12].

However, like user-based CF, the performance of item-based CF may be disrupted due to several reasons. The following issues may result in inaccurate prediction in item-based RS. **Firstly**, item–item similarity for some pair of items cannot be computed, if there is no co-rated items in the dataset. **Secondly**, item-based CF gives equal weight to all the co-rated users. However, rating of two similar user on a randomly chosen item may vary, in spite of impressive similarity between them, due to the fact that the randomly chosen item might have negligible or very small impact on the average rating taken over n rated items, i.e., it may be an outlier. This may happen in four cases:

- Biased rating by one user or both users; the ideal conditions of rating might have been violated.
- The particular item reflects two very different emotional/psychological involvement of the persons with the product.
- The taste of two users on the particular item may be different and
- One user may be stringent and other may be lenient in rating that item [13].

Therefore, a more judicious approach is that the similarity between a target user and each co-rated users should be considered in finding similar neighbors of the target item. **Thirdly,** traditional similarity functions such as cosine similarity and its variants are a measure of orientation, not magnitude. Two vectors with different magnitude but exactly the same orientation can have cosine similarity value of one. The above circumstances may lead to the disruption of prediction accuracy in CF.

The purpose of this paper is to examine the behavior of different SMs at different rating patterns of users. To alleviate the above-mentioned issues, we propose a new similarity function using the modified Bhattacharyya coefficient. The efficiency of the proposed similarity function is measured on the collected MovieLens datasets, and the comparative results of this paper are divided as: (i) The experimental analysis reveals that item-based CF with user–user similarity as weight is a better choice than user-based CF with item–item similarity as weight, (ii) user–user similarity computed using the modified Bhattacharyya coefficient results in improved accuracy in item-based CF, as compared to the weight calculated using traditional SMs, and (iii) the proposed similarity function using the modified Bhattacharyya coefficient attains more prediction accuracy than some recently used similarity measures in CF-based RS.

The remainder of the article is structured as follows. Section 2 elaborates the background and related work. In Sect. 3, limitations of various similarity measures are discussed. Section 4 highlights the motivation of the proposed work. The proposed recommender system is portrayed in Sect. 5. Experimental analysis is discussed in Sect. s6, and finally, Sect. s7 concludes our paper.

2 Background and related work

RS can be defined as a decision-making strategy which aims to provide the most relevant and accurate chunk of information to the user according to their preferences and taste by filtering from a huge pool of information. It assists the user to handle the information overload problem by providing them personalized and customized content, and services. Additionally, in the complex information environment, recommender engine discovers the data pattern in the dataset that could connect the user to their needs. Over the last two decades, recommenders have evolved from manual to automated engines, and their use has grown significantly in a variety of applications, resulting in the development of a diverse set of algorithms. Content-based and CF are the most commonly used approaches in RS [14]. The basic intuition of the content-based RS is that *"the recommendation of the target item depends on it's properties which are highly similar to the previously favored items by the target user"*. The fundamental steps involved in the content-based RS are: (i) finding the attributes of an item that may be recommended to the target user, (ii) making a profile of the user that portrays the sort of items that the user likes, (iii) contrasting the properties of the item with the user's profile, to figure out what item needs to be recommended. Content-based filtering applies on the attributes of the item, whereas CF uses the user's behavior also with the attributes of the item.

Memory-based CF utilizes the user-to-user and item-to-item correlations, based on the rating in order to predict the rating of the target user for the recommendation. The basic steps in memory-based CF for recommendation are: (i) creation of a user's profile from rating, (ii) selection of neighbors of a target user by utilizing similarity function, (iii) rating prediction of the target item, and (iv) recommendation of the Top-N items to the target user.

Haifeng Liu et al. have explored the major drawbacks of traditional SMs in CF, namely [15]: (i) ignoring the proportions of common ratings lead to lower accuracy, (ii) under Jaccard's similarity, it would be difficult to discern between different users the absolute value of the rating, because it considers only a proportion of the common ratings, (iii) two users with similar ratings might not have the same predicted rating for the target item, and (iv) two users with distinct rating patterns may be similar on the predicted rating of a target item. In order to mitigate these limitations, they have proposed a novel SM which is composed of three factors of similarity, i.e., proximity, impact, and popularity. The proximity factor not only measures the absolute difference between the two ratings, but also determines whether or not these ratings are in agreement and, in case of disagreement, also gives a penalty to the ratings. The impact factor shows the preferred or non-preferred nature of an item toward the user. Popularity denotes how common is the rating pattern of two users. If the difference between the average of two users' rating and the average of total user's rating is maximum, then these ratings of the two users may increase the accuracy of similarity values between the two users.

Sarik Ghazarian et al. have proposed a group recommender system that can solve the sparsity problem in memory-based CF [16]. Support vector machine learning model is applied in finding items' similarity in the proposed method. The two conceptions of similar and dissimilar users are as follows: (i) The two users are said to be similar if they have nearly the same rating pattern on the similar items, and (ii) if the two users rate two different items, then they are said to be dissimilar users.

Some hardware constraints may impede the scalability and processing efficiency of item-based CF because of the growing number of items and users. Chenyang Li and Kejing He have proposed an optimized MapReduce for item-based CF algorithm to minimize the scalability issue in item-based CF [17]. User's rating frequency provides a reasonable empirical factor

in the calculation of similarity value between items. Hence, they have also introduced an argument, named as the inverse user frequency that means the users who have lower rating frequency than the users who have higher rating frequency should be more involved in the calculation of item–item similarity. Their proposed algorithm divides the large-scale datasets into small jobs, and then, these small jobs are executed independently to reduce the execution overhead and to ensure better performance.

Cold-start problem is another significant issue in memory-based CF, and hence, Andre Luiz Vizine Pereira and Eduardo Raul Hruschka have presented a hybrid recommender system based on simultaneous co-clustering and learning that combines the CF with demographic information [18]. The advantage of their proposed system is that it can provide recommendation where no rating is available for the new user. To deal with the sparsity problem and to improve the accuracy of the CF-based system, Qusai Shambour et al. have used the Euclidean distance and cosine similarity as the SMs for making more personalized RS [19].

CF-based RS uses the entire rating database, resulting in poor scalability when more users and items are added to the database. To minimize this issue, Efthalia Karydi and Konstantinos G. Margaritis have constructed two parallel variants of the CF method which include benefits related to efficiency and the capacity to dynamically update data [20]. The first version is built in parallel using the OpenMP API, and its efficiency is assessed on a multi-core system. The second variant is a hybrid technique that utilizes both OpenMP and MPI which is tested in both homogeneous and heterogeneous clusters.

Zhongya Wang et al. have developed a CUDA-enabled parallel CF method that uses an efficient data partitioning scheme to speed up the execution [21].

In a survey of parallel and distributed collaborative filtering, Efthalia Karydi and Konstantinos Margaritis revealed that no memory-based methods have been built on distributed memory environment, and only one has been established in a shared memory framework [22]. Many parallel and distributed collaborative filtering algorithms have recently been developed, particularly in the use of graphics processing units and other platforms. It would be excellent to use a multilayer heterogeneous method that utilizes several machines to efficiently process large amounts of data and then combine a number of techniques.

With the distribution of bipartite material, Christos Sardianos et al. have presented a method for enhancing the effectiveness of CF algorithms that perform in a parallel configuration [23].

User's rating information plays an important role in the memory-based CF. The privacy of this information provides more efficient and accurate recommendation. In that direction, Dongsheng Li et al. have introduced an efficient privacy-preserving item based CF that can protect user privacy during online recommendation process [24]. Here, item similarities have been computed using the proposed un-synchronized secure multi-party computation protocol that preserves the privacy of the item similarity computation.

In the existing literature, some multi-criteria item-based CF techniques have been introduced to mitigate the limitations of the traditional single-criterion rating-based algorithm. Alper Bilge and Cihan Kaleli have discussed about a multi-criteria item-based CF framework [25]. In this paper, the authors have followed two steps in the calculation of similarity between items. In the first step, the similarity between items have been computed according to each criterion. The second step involves estimating the average of the similarity between each criterion. Pearson correlation, adjusted cosine similarity, Euclidean distance, Manhattan distance, and Chebyshev distance are the SMs used in this paper. Gediminas Adomavicius and YoungOk Kwon have explained the two new approaches, i.e., the similarity-based approach and the aggregation function-based approach for improving the multi-criteria rating information in recommender systems [26].

Table 1 Notations used in SMs

Symbol	Description
R(u,i)	rating of user u on item i
R(v,i)	rating of user v on item i
$\bar{R_u}$	average rating of user u
$\bar{R_v}$	average rating of user v
$\bar{R_i}$	average rating of item i
k(u,i)	rank of the rating of user u on item i
k(v,i)	rank of the rating of user v on item i
$\bar{k_u}$	average rank of ratings of the user u
$\bar{k_v}$	average rank of ratings of the user v

Table 1 shows the notations used in the equations of popular traditional SMs.

The computation equations of SMs are:

Cosine Similarity [8]

$$\mathrm{CSim}(u, v) = \frac{\sum_{i \in I}(R_{u,i})(R_{v,i})}{\sqrt[2]{\sum_{i \in I}(R_{u,i})^2}\sqrt[2]{\sum_{i \in I}(R_{v,i})^2}} \tag{1}$$

Adjusted Cosine Similarity [8]:

$$\mathrm{ACSim}(u, v) = \frac{\sum_{i \in I}(R_{u,i} - \bar{R_i})(R_{v,i} - \bar{R_i})}{\sqrt[2]{\sum_{i \in I}(R_{u,i} - \bar{R_i})^2}\sqrt[2]{\sum_{i \in I}(R_{v,i} - \bar{R_i})^2}} \tag{2}$$

Euclidean Distance [27]:

$$\mathrm{EDSim}(u, v) = \frac{1}{1 + \sqrt[2]{\sum_{i \in I}(R_{u,i} - R_{v,i})^2}} \tag{3}$$

Pearson Correlation [8]:

$$\mathrm{PCSim}(u, v) = \frac{\sum_{i \in I}(R_{u,i} - \bar{R_u})(R_{v,i} - \bar{R_v})}{\sqrt[2]{\sum_{i \in I}(R_{u,i} - \bar{R_u})^2}\sqrt[2]{\sum_{i \in I}(R_{v,i} - \bar{R_v})^2}} \tag{4}$$

Spearman Correlation [28][29]:

$$\mathrm{SCSim}(u, v) = \frac{\sum_{i \in I}(k_{u,i} - \bar{k_u})(k_{v,i} - \bar{k_v})}{\sqrt[2]{\sum_{i \in I}(k_{u,i} - \bar{k_u})^2}\sqrt[2]{\sum_{i \in I}(k_{v,i} - \bar{k_v})^2}} \tag{5}$$

For item-based CF, the equations of SMs can be derived by mutually exchanging u with i and v with j [3]. These SMs do not utilize the similarity value of a target item and other item in the computation of user similarity. Hence, K. Choi and Y. Suh [27] have proposed a new SM that outperforms the traditional SMs in the selection of neighbors for each target item.

Sparsity is a major issue in CF-based RS. Various SMs are unable to compute accurate top-k similar neighbor in a sparse scenario. Therefore, Patra et al. have provided a new SM using Bhattacharyya coefficient (BC) that outperforms the prevalent SMs. The computational equation 6 shows the SM using BC. In this equation, Jacc(u,v) represents the Jaccard similarity

between users. BC(i,j) computes the similarity on item i and item j using BC . $loc(r_{ui}, r_{vj})$ denotes the local similarity between users with respect to item i and item j.

Similarity using modified Bhattacharyya coefficient [30]:

$$BCSim(u, v) = Jacc(u, v) + \sum_{i \in I_u} \sum_{j \in I_v} BC(i, j) loc(r_{ui}, r_{vj}) \tag{6}$$

The mathematical formula of BC(i,j) and $loc(r_{u_i}, r_{v_j})$ is:

$$BC(i, j) = \sum_{h=1}^{m} \sqrt{\bar{p_{ih}} \bar{p_{jh}}} \tag{7}$$

$$loc(r_{ui}, r_{vi}) = \frac{(r_{ui} - r_{med})(r_{vi} - r_{med})}{\sqrt{\sum_{k \in I_u} (r_{uk} - r_{med})^2} \sqrt{\sum_{k \in I_v} (r_{vk} - r_{med})^2}} \tag{8}$$

Here, m identifies the number of bins. $\bar{p_{ih}}$ shows the ratio between the number of users rated the item i with rating value 'h' and the total number of users rated the item i. r_{med} is the median of the rating scale, I_u shows the set of items rated by the user u, and r_{uk} denotes the rating value of user u on item k.

A parallel CF-based RS on extending the vector (PCF-EV) is presented by Hongyi Su et al. to overcome the scalability issue [31]. To begin, the eigenvector is fairly extended to obtain the expand-vector using the expand-vector model. The expand-vectors are then used to illustrate a set of similarity evaluations. Then, the k-nearest item is determined, and the computation results are used to make more reliable recommendations to the target user. On the basis of this, additional optimization allows it to be successfully deployed to the parallel computing architecture.

3 Limitations of similarity measures

Similarity computation method plays a significant role in the prediction accuracy of CF-based RS. There is a plethora of work that exists in the literature to improve the performance of CF. However, existing SMs are not suitable in the following cases (i) when co-rated items are few or zero, and (ii) ratings of two items follow some specific patterns. The following cases are used to explain the limitations of existing SMs.

- *Case 1:* Suppose the rating vectors of 3 items and 4 users are I_1=(1,2,1,2), I_2=(2,4,2,4), and I_3=(0.5,1,0.5,1).
  In Table 2, except ED, other SMs stand testimony to the fact that two items are exactly similar in spite of their different rating patterns.
- *Case 2:* Suppose the rating vectors of two items pair are [I_1=(1,2), I_2=(1,2)], and [I_1=(1,2,1,1),I_3=(1,3,1,1)] for two and four users, respectively.
  In Table 3, I_1 and I_2 have two and I_1 and I_3 have four co-rated users, respectively. The varying lengths of rating vectors notwithstanding PC and SC compute the same similarity value in both conditions. On the contrary, in condition 1, ACS is unable to compute similarity. Thus, in such circumstances, the prediction accuracy of CF may be disrupted.
- *Case 3:* Suppose, the rating vectors of three items for four users are I_1=(1,1,1,1), I_2=(1,1,1,1), and I_3=(3,3,3,3).
  Table 4 illustrates that except ED, all SMs have some computational issues on flat rating vectors. The rating vectors of I_2 and I_3 are opposite to each other, but CS computes the

Table 2 Equal-Ratio problem [32]

Condition	Example	Similarity Measure				
	Rating vectors of items	CSim	ACSim	EDSim	PCSim	SCSim
Condition 1	I_1=(1,2,1,2), I_2=(2,4,2,4)	*1*	*−1*	**0.24**	*1*	*1*
Condition 2	I_1=(1,2,1,2), I_3=(0.5,1,0.5,1)	*1*	*−1*	**0.39**	*1*	*1*

Table 3 Unequal-Length problem [32]

Condition	Example	Similarity Measure				
	Rating vectors of items	CSim	ACSim	EDSim	PCSim	SCSim
Condition 1	I_1=(1,2), I_2=(1,2)	**1**	*NaN*	**1**	**1**	**1**
Condition 2	I_1=(1,2,1,1), I_3=(1,3,1,1)	**0.98**	*−1*	**0.50**	*1*	*1*

same similarity value in both the conditions 1 and 2. ACS, PC, and SC are unable to calculate the similarity value in both conditions.

- *Case 4:* Suppose the rating vectors of two item pair for four users are [I_1=(1,5,5,1), I_2=(5,1,1,5)], and [I_1=(1,1,5,5), I_3=(5,5,1,1)]. In Table 5, both item pairs have four co-rated users. The circumstantial evidence helps us to draw the conclusion that both items I_2 and I_3 have a rating value of 5 when item I_1 has a rating value of 1, and vice versa. In addition to this, only CS and ED compute the positive similarity value (greater than minimum similarity value) when the rating vectors of items are opposite to each other. In such a case, it won't be incorrect to conclude that CS and ED may provide inaccurate rating prediction in CF-based RS.
- *Case 5:* Suppose the rating vectors of three items for single user are I_1=(1), I_2=(3), and I_3=(5).
  In Table 6, only ED computes the different similarity values in both the conditions. Similarity values cannot be computed using PC and SC. Using CS, we can misinterpret that items I_2 and I_3 are equally similar to I_1. Items I_2 and I_3 are found to be most dissimilar to I_1 according to ACS value.
- *Case 6:* Suppose the rating vectors of two items pair are [I_1=(1,5,1,3), I_2=(5,1,3,1)], and [I_1=(1,5), I_3=(5,1)] for four and two users, respectively.
  From Table 7, a number of co-rated users for item pairs (I_1, I_3) and (I_1, I_2) are four and two, respectively. In both the conditions, CS and ED provide positive similarity value (means greater than minimum similarity value) where the rating vectors of each pair are opposite to each other. Furthermore, ACS and PC have the same similarity value in both the conditions. These nature of SMs may degrade the rating prediction accuracy in CF.
- *Case 7:* Suppose the rating vectors of three items for four users are I_1=(?,1,?,2), I_2=(2,?,?,?), and I_3=(?,?,1,?). Here, ? denotes that a user did not rate the particular item. Table 8 shows all SMs are unable to compute the similarity between items. In such a scenario, using CS, ACS, ED, PC, and SC the rating prediction accuracy of CF may decrease significantly.

The above situations highlight the fact that no single SM is suitable to compute the similarity between items in CF. Another key drawback of traditional SMs is that they give equal weightage to all co-rated users to find similar neighbors.

Table 4 Flat-Value problem [32]

Condition	Example	Similarity Measure				
	Rating vectors of items	CSim	ACSim	EDSim	PCSim	SCSim
Condition 1	I_1=(1,1,1,1), I_2=(1,1,1,1)	**1**	*NaN*	**1**	*NaN*	*NaN*
Condition 2	I_1=(1,1,1,1), I_3=(3,3,3,3)	*1*	**−1**	**0.20**	*NaN*	*NaN*

Table 5 Opposite-Value [32]

Condition	Example	Similarity Measure				
	Rating vectors of items	CSim	ACSim	EDSim	PCSim	SCSim
Condition 1	I_1=(1,5,5,1), I_2=(5,1,1,5)	*0.38*	**−1**	*0.11*	**−1**	**−1**
Condition 2	I_1=(1,1,5,5), I_3=(5,5,1,1)	*0.38*	**−1**	*0.11*	**−1**	**−1**

Table 6 Single-Value [32]

Condition	Example	Similarity Measure				
	Rating vectors of items	CSim	ACSim	EDSim	PCSim	SCSim
Condition 1	I_1=(1), I_2=(3)	*1*	*−1*	**0.33**	*NaN*	*NaN*
Condition 2	I_1=(1), I_3=(5)	*1*	**−1**	**0.20**	*NaN*	*NaN*

Table 7 Cross-Value problem [32]

Condition	Example	Similarity Measure				
	Rating vectors of items	CSim	ACSim	EDSim	PCSim	SCSim
Condition 1	I_1=(1,5,1,3), I_2=(5,1,3,1)	*0.44*	*−1*	**0.14**	*−1*	*-0.89*
Condition 2	I_1=(1,5), I_3=(5,1)	*0.38*	**−1**	*0.15*	**−1**	**−1**

To attain more rating prediction accuracy, K. Choi and Y. Suh have introduced a new similarity function that adopts the similarity value of the target item and other items as a weight for a user-based CF. The experimental results of K. Choi and Y. Suh have indicated that user-based CF using Pearson correlation obtains more recommendation accuracy than other SMs when item similarity value is used as a weight in the computation of user–user similarity. Although their proposed SM considers the aspect of CF-based RS that two similar

Table 8 Sparsity problem when no co-rated items exist [30]

Condition	Example	Similarity Measure				
	Rating vectors of items	CSim	ACSim	EDSim	PCSim	SCSim
Condition 1	I_1=(?,1,?,2), I_2=(2,?,?,?)	*NaN*	*NaN*	*NaN*	*NaN*	*NaN*
Condition 2	I_1=(?,1,?,2), I_3=(?,?,1,?)	*NaN*	*NaN*	*NaN*	*NaN*	*NaN*

Table 9 Proposed SM of K. Choi and Y. Suh at various issues

Issue	Example	Similarity of U_1 at different target item			
	Rating vectors of users	I_1	I_2	I_3	I_4
Equal-Ratio	U_1=(1,2,1,2), U_2=(2,4,2,4)	**1**	**1**	**1**	**1**
	U_1=(1,2,1,2), U_3=(0.5,1,0.5,1)	**1**	**1**	**1**	**1**
Unequal-Length	U_1=(1,2), U_2=(1,2)	*NaN*	*NaN*	*NaN*	*NaN*
	U_1=(1,2,1,1), U_3=(1,3,1,1)	*NaN*	*NaN*	*NaN*	*NaN*
Flat-Value	U_1=(1,1,1,1), U_2=(1,1,1,1)	*NaN*	*NaN*	*NaN*	*NaN*
	U_1=(1,1,1,1), U_3=(3,3,3,3)	*NaN*	*NaN*	*NaN*	*NaN*
Opposite-Value	U_1=(1,5,5,1), U_2=(5,1,1,5)	**0**	**0**	**0**	**0**
	U_1=(1,1,5,5), U_3=(5,5,1,1)	**0**	**0**	**0**	**0**
Single-Value	U_1=(1), U_2=(3)	*NaN*	*NaN*	*NaN*	*NaN*
	U_1=(1), U_3=(5)	*NaN*	*NaN*	*NaN*	*NaN*
Cross-Value	U_1=(1,5,1,3), U_2=(5,1,3,1)	**0**	**0**	**0**	**0**
	U_1=(1,5), U_3=(5,1)	**0**	**0**	**0**	**0**
No co-rated user exist	U_1=(?,1,?,2), U_2=(2,?,?,?)	*NaN*	*NaN*	*NaN*	*NaN*
	U_1=(?,1,?,2), U_3=(?,?,1,?)	*NaN*	*NaN*	*NaN*	*NaN*

users may have different opinions on a set of items, their proposed SM fails in some rating patterns of users as shown in Table 9.

Table 9 highlights the similarity values of users at different target item. In spite of computing similarity at each target item, the proposed SM of K. Choi and Y. Suh is unable to resolve many problems such as Unequal-Length, Flat-Value, Single-Value, and sparsity (when no co-rated users exist).

In this direction, to mitigate the above-mentioned issues, Z. Tan et al. have suggested a new SM. The proposed algorithm is based on physical resonance phenomenon [32]. Although the resonance similarity overcomes most of the above-discussed issues, major concern still exists in their proposed approach. (i) In reality, different applications have a different percentage of sparse data [3,30,33]. However, the comparative analysis of their proposed approach at different sparsity levels has not been discussed. (ii) The efficiency of CF algorithms is justified by various statistical metrics such as MAE, RMSE, precision, recall, F1-measure, t test, and confidence [27,30,33]. But, Z. Tan et al. have used only MAE in their comparative results. (iii) Another important aspect, i.e., on a set of items, similar users may have different rating patterns, is not addressed in their proposed approach. (iv) The ratings of all co-rated items are considered in their proposed approach, and as a result, it may also suffer from performance issues [30].

4 Motivation

Sparsity is a major concern in item-based CF, as correlation-based SMs such as cosine similarity, Pearson correlation, and its variants fail to calculate the similarity between some items. In statistics, the Bhattacharyya distance measures the similarity of two probability distribution. It is closely related to the Bhattacharyya coefficient. Bhattacharyya coefficient named after Anil Kumar Bhattacharyya, a statistician, can be a suitable similarity measure in sparse

Table 10 Similarity value by BC

Problem	Example Rating vectors of items	Similarity Value
Equal-Ratio	I_1=(1,2,1,2), I_2=(2,4,2,4)	*0.5*
	I_1=(1,2,1,2), I_3=(0.5,1,0.5,1)	*0.5*
Unequal-Length	I_1=(1,2), I_2=(1,2)	**1**
	I_1=(1,2,1,1), I_3=(1,3,1,1)	**0.75**
Flat-Value	I_1=(1,1,1,1), I_2=(1,1,1,1)	**1**
	I_1=(1,1,1,1), I_3=(3,3,3,3)	**0**
Opposite-Value	I_1=(1,5,5,1), I_2=(5,1,1,5)	*1*
	I_1=(1,1,5,5), I_3=(5,5,1,1)	*1*
Single-Value	I_1=(1), I_2=(3)	*0*
	I_1=(1), I_3=(5)	*0*
Cross-Value	I_1=(1,5,1,3), I_2=(5,1,3,1)	*1*
	I_1=(1,5), I_3=(5,1)	*1*
No co-rated items exist	I_1=(?,1,?,2), I_2=(2,?,?,?)	**0.25**
	I_1=(?,1,?,2), I_3=(?,?,1,?)	*0.25*

dataset, because it does not consider **co-rated items and is a measure of magnitude, not orientation**. Instead, the Bhattacharyya coefficient computes the relative closeness between two statistical samples [34,35]. Table 10 shows the similarity value of items using BC on different rating patterns.

It may well be concluded from Table 10 that BC is the most suitable SM for a sparse dataset, though it fails to resolve some of the issues such as Equal-Ratio, Opposite-Value, Single-Value, and Cross-validation problems.

Although many SMs are found in the literature to enhance the rating prediction accuracy in CF, but, throughout the above discussion, we can observe that no single SM can overcome all the mentioned issues. Table 11 represents the brief description of various SMs for different aspects of CF-based RS.

In Table 11, ✓ identifies that the SM can resolve the issue. × denotes that the SM is unable to resolve the issue. And, ∗ notifies that the SM can resolve some issues of rating patterns, and others cannot be resolved.

However, similar users may have different opinions on some items [27]. For example, suppose that u_1, u_2, and u_3 are the three users, and their rating vectors on six items are $u_1 = (5, 1, 1, 5, 5, ?)$, $u_2 = (5, 5, 5, 5, 5, 1)$, $u_3 = (1, 1, 1, 1, 1, 1)$. Here, '?' denotes that user u_1 does not rate the item six. Rating vectors clearly show that users u_1 and u_2 are the most similar. However, their taste on second and third items are totally different (opposite).

Inspired by the above observations, the rating prediction accuracy of item-based CF can be improved (i) if user–user similarity is used as a weight to find similar items, and (ii) BC and modified BCSim are utilized to satisfy the varied taste of similar users.

5 Proposed recommender system

The proposed RS is divided into four modules, namely (i) data collection, (ii) data processing, (iii) rating prediction, and (iv) Top-N recommendation as presented in Fig. 1. Data collection

Table 11 Performance of various SMs at different rating patterns

Similarity	Resolved issues							
Measure	Equal-Ratio	Unequal-Length	Flat-Value	Opposite-Value	Single-Value	Cross-Value	No co-rated items exist	Different opinions of similar users
CSim	×	✓	*	×	×	×	×	×
ACSim	×	×	*	✓	*	*	×	×
EDSim	✓	✓	✓	×	✓	*	×	×
PCSim	×	*	×	✓	×	*	×	×
SCSim	×	*	×	✓	×	*	×	×
H. J. Ahn. [36]	✓	✓	✓	✓	✓	✓	×	×
JacUOD [37]	✓	✓	✓	×	✓	*	×	×
K. Choi and Y. Suh [27]	×	×	×	*	×	*	×	✓
H. Liu et al. [15]	✓	✓	✓	✓	✓	✓	×	×
W. Wang. et al. [38]	×	*	×	✓	×	*	×	×
BCSim [30]	✓	✓	*	✓	*	✓	✓	×
Z. Tan et al. [32]	✓	✓	✓	✓	✓	✓	×	×
A. Gazdar et al. [39]	✓	✓	✓	✓	✓	✓	×	×
CF_{DR}_PC [40]	✓	✓	✓	✓	✓	✓	×	×
CF_ITR [41]	✓	✓	✓	✓	✓	✓	×	×
BC	×	✓	✓	×	×	×	*	×

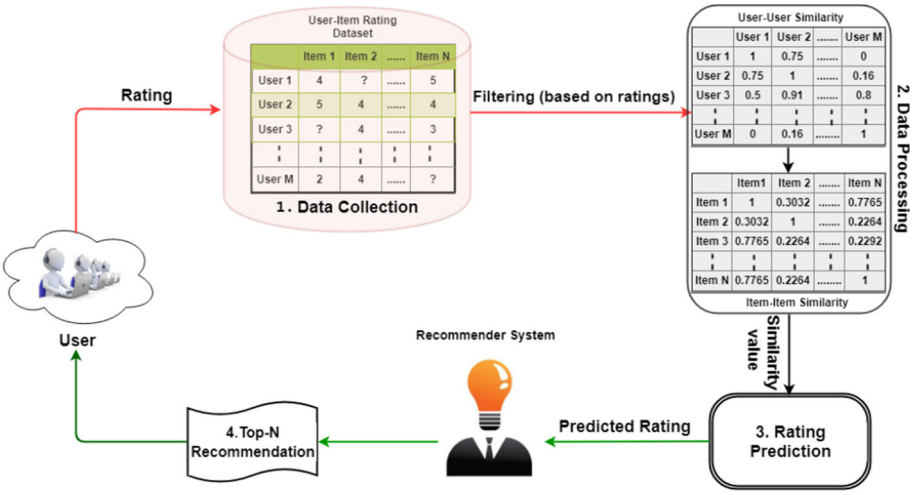

Fig. 1 Proposed Recommender System based on Item-based Collaborative Filtering

Table 12 Proposed similarity measure

Modified Similarity Measure	Equation
$CF\_B_I\_B_U$	$\text{BC(u,v)} = \sum_{h=1}^{m} \sqrt{\bar{p_{uh}} \bar{p_{vh}}}$
	$loc_{mod}(r_{iu}, r_{iv}) = \frac{BCSim(u,v)(r_{iu} - r_{med})(r_{iv} - r_{med})}{\sqrt{\sum_{k \in U_i}(r_{ik} - r_{med})^2} \sqrt{\sum_{k \in U_j}(r_{jk} - r_{med})^2}}$
	$\text{BC}Sim^u\text{(i,j)} = \text{Jacc(i,j)} + \frac{\sum_{i,j \in I} \sum_{u,v \in U} BC(u,v) loc_{mod}(r_{iu}, r_{jv})}{\sum_{u,v \in U} BCSim(u,v)}$

comprises the methods that collect the users' feedback on the target item. These collected feedbacks are converted into user–item rating matrix for further CF-based recommendations. In the next step of the proposed RS, this user–item matrix is used to find the similarity value using proposed SM.

5.1 Proposed similarity function to find similar items of a target item for a target user

This section gives the detailed information of proposed SM that uses the similarity value of the similar user in finding the top-k nearest items of a target item. The computational equations of modified BC, i.e., proposed SM, are shown in Table 12.

In Table 12, $CF\_B_I\_B_U$ denotes item similarity using BCSim weighted by user similarity using BCSim for target user u and target item i. U and I represent the set of users and items, respectively. Jacc(i,j) shows the Jaccard similarity between items. BC(u,v) finds the similarity between users u and v using BC. $loc_{mod}(r_{iu}, r_{jv})$ denotes the local similarity between items i and j with respect to user u and user v. m identifies the number of bins. $\bar{p_{uh}}$ shows the ratio between the number of items rated by user u and the number of items rated by user u with rating value h. r_{med} is the median of the rating scale, U_i demonstrates the set of users who rate the item i, and r_{ik} denotes the rating value of item i by user k.

The advantages of $CF\_B_I\_B_U$ can be illustrated in Table 13, where similarity of item I_1 and I_2 is computed on different rating patterns for different target users. Instead, Table 12 is utilized in the formation of Table 13 as discussed in the following steps.

In Equal-Ratio problem for target user 1:

$$\sum_{i,j\in I}\sum_{u,v\in U} BC(u,v)loc_{mod}(r_{iu},r_{jv}) =$$
$$[\ \mathrm{BC}(1,1)\mathrm{BCSim}(1,1)\{(r_{11}-r_{med})(r_{11}-r_{med})+(r_{11}-r_{med})(r_{21}-r_{med})+(r_{21}-r_{med})(r_{11}-r_{med})+(r_{21}-r_{med})(r_{21}-r_{med})\}+\mathrm{BC}(1,2)\mathrm{BCSim}(1,2)\{(r_{11}-r_{med})(r_{12}-r_{med})+(r_{11}-r_{med})(r_{22}-r_{med})+(r_{21}-r_{med})(r_{12}-r_{med})+(r_{21}-r_{med})(r_{22}-r_{med})\}+\mathrm{BC}(1,3)\mathrm{BCSim}(1,3)\{(r_{11}-r_{med})(r_{13}-r_{med})+(r_{11}-r_{med})(r_{23}-r_{med})+(r_{21}-r_{med})(r_{13}-r_{med})+(r_{21}-r_{med})(r_{23}-r_{med})\}+\mathrm{BC}(1,4)\mathrm{BCSim}(1,4)\{(r_{11}-r_{med})(r_{14}-r_{med})+(r_{11}-r_{med})(r_{24}-r_{med})+(r_{21}-r_{med})(r_{14}-r_{med})+(r_{21}-r_{med})(r_{24}-r_{med})\}]\ /\ (\sqrt{(r_{11}-r_{med})^2+(r_{12}-r_{med})^2+(r_{13}-r_{med})^2+(r_{14}-r_{med})^2}\ *\ \sqrt{(r_{21}-r_{med})^2+(r_{22}-r_{med})^2+(r_{23}-r_{med})^2+(r_{24}-r_{med})^2})$$

$$= [1*2.4\{(1-3)(1-3)+(1-3)(2-3)+(2-3)(1-3)+(2-3)(2-3)\}$$
$$+(0.5*1.16)\{(1-3)(2-3)+(1-3)(4-3)+(2-3)(2-3)+(2-3)(4-3)\}$$
$$+(1*2.4)\{(1-3)(1-3)+(1-3)(2-3)+(2-3)(1-3)+(2-3)(2-3)\}$$
$$+(0.5*1.16)\{(1-3)(2-3)+(1-3)(4-3)+(2-3)(2-3)+(2-3)(4-3)\}]/\{\sqrt{10}*\sqrt{4})\}$$
$$= \{2*2.4*(4+2+2+1)+1.16(2-2+1-1)\}/\sqrt{40} = 41.2/\sqrt{40} = 6.51 \text{ And,}$$
$$\sum_{u,v\in U} BCSim(u,v) = \mathrm{BCSim}(1,1)+\mathrm{BCSim}(1,2)+\mathrm{BCSim}(1,3)+\mathrm{BCSim}(1,4)$$
$$= 2.4+1.16+2.4+1.16 = 7.12$$
$$Sim^{U_1}(I_1,I_2) = \mathrm{Jacc}(I_1,I_2)+(6.51/7.12) = 1+0.91 = \mathbf{1.91}$$

In Equal-Ratio problem for target user 2:

$$\sum_{i,j\in I}\sum_{u,v\in U} BC(u,v)loc_{mod}(r_{iu},r_{jv}) =$$
$$[\ \mathrm{BC}(2,1)\mathrm{BCSim}(2,1)\{(r_{12}-r_{med})(r_{11}-r_{med})+(r_{12}-r_{med})(r_{21}-r_{med})+(r_{22}-r_{med})(r_{11}-r_{med})+(r_{22}-r_{med})(r_{21}-r_{med})\}+\mathrm{BC}(2,2)\mathrm{BCSim}(2,2)\{(r_{12}-r_{med})(r_{12}-r_{med})+(r_{12}-r_{med})(r_{22}-r_{med})+(r_{22}-r_{med})(r_{12}-r_{med})+(r_{22}-r_{med})(r_{22}-r_{med})\}+\mathrm{BC}(2,3)\mathrm{BCSim}(2,3)\{(r_{12}-r_{med})(r_{13}-r_{med})+(r_{12}-r_{med})(r_{23}-r_{med})+(r_{22}-r_{med})(r_{13}-r_{med})+(r_{23}-r_{med})(r_{23}-r_{med})\}+\mathrm{BC}(2,4)\mathrm{BCSim}(2,4)\{(r_{12}-r_{med})(r_{14}-r_{med})+(r_{12}-r_{med})(r_{24}-r_{med})+(r_{22}-r_{med})(r_{14}-r_{med})+(r_{22}-r_{med})(r_{24}-r_{med})\}]\ /\ (\sqrt{(r_{11}-r_{med})^2+(r_{12}-r_{med})^2+(r_{13}-r_{med})^2+(r_{14}-r_{med})^2}\ *\ \sqrt{(r_{21}-r_{med})^2+(r_{22}-r_{med})^2+(r_{23}-r_{med})^2+(r_{24}-r_{med})^2})$$

$$= [0.5*1.16\{(2-3)(1-3)+(2-3)(2-3)+(4-3)(1-3)+(4-3)(2-3)\}$$
$$+(1*1.5)\{(2-3)(2-3)+(2-3)(4-3)+(4-3)(2-3)+(4-3)(4-3)\}$$
$$+(0.5*1.16)\{(2-3)(1-3)+(2-3)(2-3)+(4-3)(1-3)+(4-3)(2-3)\}$$
$$+(1*1.5)\{(2-3)(2-3)+(2-3)(4-3)+(4-3)(2-3)+(4-3)(4-3)\}]/\{\sqrt{10*4}\}$$
$$= \{1.16*(2+1-2-1)+3(1-1-1+1)\}/\sqrt{40} = 0/\sqrt{40} = 0 \text{ And,}$$
$$\sum_{u,v\in U} BCSim(u,v) = \mathrm{BCSim}(2,1)+\mathrm{BCSim}(2,2)+\mathrm{BCSim}(2,3)+\mathrm{BCSim}(2,4)$$

Table 13 Similarity value by $CF\_B_I\_B_U$

Problem	Example Rating vectors of items	Similarity Value for target user U_1 on target item I_1	Similarity Value for target user U_2 on target item I_1
Equal-Ratio	I_1=(1,2,1,2), I_2=(2,4,2,4)	$Sim^{U_1}(I_1,I_2)$=1.91	$Sim^{U_2}(I_1,I_2)$=1
	I_1=(1,2,1,2), I_3=(0.5,1,0.5,1)	$\mathbf{Sim}^{U_1}(I_1,I_3)$=1.98	$Sim^{U_2}(I_1,I_3)$=1.57
Unequal-Length	I_1=(1,2), I_2=(1,2)	$Sim^{U_1}(I_1,I_2)$=2.6	$Sim^{U_2}(I_1,I_2)$=1.4
	I_1=(1,2,1,1), I_3=(1,3,1,1)	$Sim^{U_1}(I_1,I_3)$=2.01	$Sim^{U_2}(I_1,I_3)$=1.02
Flat-Value	I_1=(1,1,1,1), I_2=(1,1,1,1)	$Sim^{U_1}(I_1,I_2)$=2	$Sim^{U_2}(I_1,I_2)$=2
	I_1=(1,1,1,1), I_3=(3,3,3,3)	$Sim^{U_1}(I_1,I_3) = NaN$	$Sim^{U_2}(I_1,I_3) = NaN$
Opposite-Value	I_1=(1,5,5,1), I_2=(5,1,1,5)	$Sim^{U_1}(I_1,I_2)$=0	$Sim^{U_2}(I_1,I_2)$=0
	I_1=(1,1,5,5), I_3=(5,5,1,1)	$Sim^{U_1}(I_1,I_3)$=0	$Sim^{U_2}(I_1,I_3)$=0
Single-Value	I_1=(1), I_2=(3)	$Sim^{U_1}(I_1,I_2) = NaN$	–
	I_1=(1), I_3=(5)	$Sim^{U_1}(I_1,I_3)$=0	–
Cross-Value	I_1=(1,5,1,3), I_2=(5,1,3,1)	$Sim^{U_1}(I_1,I_2)$=1	$Sim^{U_2}(I_1,I_2)$=1
	I_1=(1,5), I_3=(5,1)	$Sim^{U_1}(I_1,I_3)$=1	$Sim^{U_2}(I_1,I_3)$=1
No co-rated items exist	I_1=(?,1,?,2), I_2=(2,?,?,?)	$Sim^{U_1}(I_1,I_2)$=1.79	$Sim^{U_2}(I_1,I_2) = NaN$
	I_1=(?,1,?,2), I_3=(?,?,1,?)	$Sim^{U_1}(I_1,I_3) = NaN$	$Sim^{U_2}(I_1,I_3)$1.60

Table 14 Notations

Notations	Descriptions
U	$\{u_1, u_2, ..., u_x, u_m\}$, a set of users and $1 \leq x \leq m$
I	$\{i_1, i_2, ..., i_y, i_n\}$, a set of items and $1 \leq y \leq n$
UI	The user–item rating matrix
I_{uv}	a set of items co-rated by users u and v
U_{ij}	a set of users who rated the items i and j
DesSort()	a sorting function
$\otimes$	operators used in BCSim
$\odot$	operators used in the prediction approach
$UI_{predicted}$(u,i)	The predicted rating of user u on item i

$$= 1.16 + 1.5 + 1.16 + 1.5 = 5.32$$
$$Sim^{U_2}(I_1, I_2) = \text{Jacc}(I_1, I_2) + (0/5.32) = 1 + 0 = \mathbf{1.0}.$$

Similarly, we can compute the desired similarity value for each rating patterns.

In Table 13, in most of the cases, $CF\_B_I\_B_U$ resolves the illustrated issues. In reality, the user–item rating dataset may be highly sparse and non-co-rated. In such scenario, from Table 13, we can notice that $CF\_B_I\_B_U$ computes different similarity values for each rating pattern.

5.2 Rating prediction of target items for the target user

After calculating similarity values, the prediction approach has been used in the rating prediction of the target item [42–44]. Firstly, all the similarity between target item and other items are calculated, and then, Top-k similar items with highest similarity values are selected as the closest similar item of the target item. In the last step of prediction approach, the aggregation function is used to predict the rating of target user on the target item. The equation of prediction approach becomes as follows:

$$\hat{r_{u,i}} = \bar{r_i} + \frac{\sum_{j=1}^{k} Sim^u(i, j)(r_{u,j} - \bar{r_j})}{\sum_{j=1}^{k} |Sim^u(i, j)|} \tag{9}$$

Here, $\hat{r_{u,i}}$ shows the predicted rating of target item i for target user u and k is the number of the closest similar items of the target item. Sim^u(i,j) identifies the similarity between target item i and other item j for target user u. After determining all the predicted rating of target items, a list of Top-N item is selected and recommended to the target users.

5.3 Proposed algorithm

A detailed process for rating prediction using the proposed SM is given in Algorithm 1 using notations as discussed in Table 14.

Algorithm 1 *Rating prediction using Proposed Similarity Function*

1: ***Input :*** *user-item rating dataset* ($UI_{m\times n}$).
2: ***Output :*** *Predicted rating for unrated items.*
3: ***Step 1: Finding user-user similarity***
4: ***for*** $u \in U$ ***do***
5: ***for*** $v \in U$ ***do***
6: ***for*** $i \in I$ ***do***
7: $USim(u,v) = R(u,i) \bigotimes R(v,i)$
8: ***Step 2: Finding item-item similarity for target user u using user-user similarity values as weighting factor***
9: ***for*** $i \in I$ ***do***
10: ***for*** $j \in I$ ***do***
11: ***for*** $v \in U_{ij}$ ***do***
12: $Sim^u(i,j) = USim(u,v) \bigotimes (R(v,i) \bigotimes R(v,j))$
13: ***Step 3: Arranging the item-item similarity value in descending order***
14: ***for*** $u \in U$ ***do***
15: ***for*** $i \in I$ ***do***
16: $Sim^u(i,j) = DesSort(Sim^u(i,j))$
17: ***Step 4: Prediction of items' rating for the target user using top-k similar items***
18: ***for*** $u \in U$ ***do***
19: ***for*** $i \in I$ ***do***
20: ***if*** $UI(u,i) == 0$ ***then***
21: ***for*** $j \in I$ ***do***
22: $UI_{predicted}(u,i) = Sim^u(i,j) \bigodot R(u,j)$

5.4 Time complexity of the proposed algorithm

The proposed algorithm is divided into four steps. First step calculates the similarity between users. The similarity value between two users is used as the weighing factor to find the item similarity value of target users in step 2. All item–item similarity values are arranged in descending order in step 3. And the final step utilizes the outputs of the previous steps as an input in the prediction of rating of the item for the target user. Lines 4 to 7 take O(m), O(m), O(n), O(n) time, respectively, in execution [45]. Therefore, the time complexity of step 1 is O(m^2n^2). Similarly, step 2 takes O(m^2n^2) time complexity due to O(n), O(n), O(m), and O(m) from lines 9 to 12, respectively. In step 3, sorting technique is applied to arrange the item similarity value. Therefore, it has various complexity for different cases. Lines 14 to 16 take $\Omega(mn)$, $\theta(mnlogn)$, and O(mn^2) as best, average, and worst case execution time, respectively. And in the final step, the execution time from line 18 to 22 is O(m), O(n), O(1), O(k), and O(1), respectively. Hence, the time complexity to execute the step 4 is O(mnk). Here, k is the total number of predicted ratings. Table 15 shows the total time complexity of the $CF\_B_I\_B_U$.

5.5 Illustrative example

In this section, we provide an illustrative example using the modified traditional SMs (under proposed idea, i.e., utilization of user similarity as a weight in an item-based CF) as well as

Table 15 Time complexity of $CF\_B_I\_B_U$

Case	Execution Complexity
Best	$\Omega(m^2n^2) + \Omega(m^2n^2) + \Omega(mn) + \Omega(mnk) \approx \Omega(m^2n^2)$
Average	$\Theta(m^2n^2) + \Theta(m^2n^2) + \Theta(mnlogn) + \Theta(mnk) \approx \Theta(m^2n^2)$
Worst	$O(m^2n^2) + O(m^2n^2) + O(mn^2) + O(mnk) \approx O(m^2n^2)$

Table 16 Notations used in the modified traditional similarity measures

Notation	Description
$CF\_C_I\_C_U$	(CSim^u(i,j)) item similarity using CSim weighted by user similarity using CSim for target user u and target item i.
$CF\_A_I\_A_U$	(ACSim^u(i,j)) item similarity using ACSim weighted by user similarity using ACSim for target user u and target item i.
$CF\_E_I\_E_U$	(EDSim^u(i,j)) item similarity using EDSim weighted by user similarity using EDSim for target user u and target item i.
$CF\_P_I\_P_U$	(PCSim^u(i,j)) item similarity using PCSim weighted by user similarity using PCSim for target user u and target item i.
$CF\_S_I\_S_U$	(SCSim^u(i,j)) item similarity using SCSim weighted by user similarity using SCSim for target user u and target item i.

Table 17 Modified traditional similarity measures

Modified Similarity Measure	Equation
$CF\_C_I\_C_U$	$CSim^u(i,j) = \frac{\sum_{u,v\in U} CSim(u,v)^2 (R_{i,v})(R_{j,v})}{\sqrt[2]{\sum_{u,v\in U}(CSim(u,v)*R_{i,v})^2}\ \sqrt[2]{\sum_{u,v\in U}(CSim(u,v)*R_{j,v})^2}}$
$CF\_A_I\_A_U$	$ACSim^u(i,j) = \frac{\sum_{u,v\in U} ACSim(u,v)^2 (R_{i,v}-\bar{R_v})(R_{j,v}-\bar{R_v})}{\sqrt[2]{\sum_{u,v\in U}\{ACSim(u,v)*(R_{i,v}-\bar{R_v})\}^2}\ \sqrt[2]{\sum_{u,v\in U}\{ACSim(u,v)*(R_{j,v}-\bar{R_v})\}^2}}$
$CF\_E_I\_E_U$	$EDSim^u(i,j) = \frac{1}{1+\sqrt[2]{\sum_{u,v\in U}\{EDSim(u,v)*(R_{i,v}-R_{j,v})\}^2}}$
$CF\_P_I\_P_U$	$PCSim^u(i,j) = \frac{\sum_{u,v\in U} PCSim(u,v)^2 (R_{i,v}-\bar{R_i})(R_{j,v}-\bar{R_j})}{\sqrt[2]{\sum_{u,v\in U}\{PCSim(u,v)*(R_{i,v}-\bar{R_i})\}^2}\ \sqrt[2]{\sum_{u,v\in U}\{PCSim(u,v)*(R_{j,v}-\bar{R_j})\}^2}}$
$CF\_S_I\_S_U$	$SCSim^u(i,j) = \frac{\sum_{u,v\in U} SCSim(u,v)^2 (k_{i,v}-\bar{k_i})(k_{j,v}-\bar{k_j})}{\sqrt[2]{\sum_{u,v\in U}\{SCSim(u,v)*(k_{i,v}-\bar{k_i})\}^2}\ \sqrt[2]{\sum_{u,v\in U}\{SCSim(u,v)*(k_{j,v}-\bar{k_j})\}^2}}$

using the proposed SM ($CF\_B_I\_B_U$). Table 16 shows the notations used in the modified traditional SMs, and the equations of modified traditional SMs are represented in Table 17.

Here, $R_{i,v}$, and $R_{i,v}$ show the rating value of items i and j rated by user v. $\bar{R_j}$, $\bar{k_i}$, and $\bar{k_j}$ represent the average rating of item j, average rank of the item i, and average rank of the item j, respectively. k(i,v), and k(j,v) show the rank of the rating of items i and j rated by user v.

Table 18 shows the rating information of nine users and eleven items, where T_u and T_i are the target user and item, respectively. Table 19 depicts the similarity between target user and other user using the traditional SMs.

Table 18 User–item rating information

User	Item I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	T_i
U_1	0.5	5	3	2	4	5	3	1	1	2.5	2
U_2	1	4	2	2.5	3	3.5	3	3	2.5	1	2.5
U_3	3.5	3.5	1	4	2.5	4	3.5	2.5	2.5	3	3
U_4	3.5	2	2.5	4	3	2	4	2.5	3.5	3.5	4.5
U_5	4	2.5	4	4	3.5	1	4.5	5	4	3.5	5
U_6	2	5	4.5	1	2	1.5	1	1.5	1	2.5	0.5
U_7	2.5	4.5	3	1	5	3	1	2.5	3	3	1
U_8	1.5	4	3	3.5	4	3.5	3.5	2	3	2	3
T_u	1	3	2.5	3.5	4	4	2	2.5	5	5	-

Table 19 Similarity between target user and other user

Users	CSim	ACSim	EDSim	PCSim	SCSim	BCSim
T_u-U_1	0.852277	− 0.0492	0.029197	0.2249	0.2098	1.1194
T_u-U_2	0.887483	− 0.19823	0.035714	0.0762	0.02492	1.0909
T_u-U_3	0.89628	0.091387	0.040404	− 0.0235	− 0.07234	1
T_u-U_4	0.908919	0.237249	0.045455	− 0.0141	− 0.07188	0.6697
T_u-U_5	0.862183	0.076422	0.02649	− 0.3427	− 0.53814	0.7233
T_u-U_6	0.767129	− 0.40385	0.019704	− 0.1491	− 0.06542	1.2696
T_u-U_7	0.904434	0.202836	0.043478	0.312	0.51131	1.2785
T_u-U_8	0.926307	0.201527	0.054795	0.2673	0.20939	1

5.5.1 Calculation for the similarity between target item and other item

$\mathrm{C}Sim^{T_u}(T_i\text{-}I_1) = \frac{A}{\sqrt[2]{B} * \sqrt[2]{C}}$

where A, B, and C are the symbols used in the proposed SMs as shown in Table 12.

A = (0.726*0.5*2) + (0.788*1*2.5) + (0.803*3.5*3) + (0.826*3.5*4.5) + (0.743*4*5) + (0.588*2*0.5) + (0.818*2.5*1) + (0.858*1.5*3).

$A \approx 45.504$.

$B = (0.852*0.5)^2 + (0.887*1)^2 + (0.896*3.5)^2 + (0.909*3.5)^2 + (0.862*4)^2 + (0.767*2)^2 + (0.904*2.5)^2 + (0.926*1.5)^2$.

$B \approx 42.221$.

$C = (0.852*2)^2 + (0.887*2.5)^2 + (0.896*3)^2 + (0.909*4.5)^2 + (0.862*5)^2 + (0.767*0.5)^2 + (0.904*1)^2 + (0.926*3)^2$. $C \approx 59.059$.

$\mathrm{C}Sim^{T_u}(T_i\text{-}I_1) = \frac{45.504}{\sqrt[2]{42.221} * \sqrt[2]{59.059}} \approx 0.912$.

Similarly, the similarity value of target item and other item is calculated using the traditional SMs and the proposed SMs, respectively, as shown in Table 20. Based on these similarity values using the SMs, Table 21 shows the ranking of similar items for T_i.

5.5.2 Calculation of rating prediction of the target item for the target user

$r_{\hat{T_u,T_i}}$ using $\mathrm{C}Sim^{T_u}(T_i\text{-}I_1) = \bar{r_{T_i}} + \frac{X}{Y}$

where $\bar{r_{T_i}} = 2.6875$
X = 0.912*(1-2.313) + 0.982*(2-2.75) + 0.985*(2-2.938) + 0.934*(2.5-2.5) + 0.950*(5-2.563) ≈ -0.543
Y = 0.912 + 0.982 + 0.985 + 0.934 + 0.950 ≈ 4.763
$\hat{r_{T_u,T_i}} = 2.6875 + \frac{-0.543}{4.763} \approx 2.57$.

The predicted rating of the target item is calculated using the proposed SMs in Table 22. We consider only the Top-5 similar items in the rating prediction.

6 Experiments

This section explains the effectiveness of the proposed SMs in item-based CF and is divided into two subsections, i.e., (i) experimental setup and (ii) comparative analysis.

6.1 Experimental setup

The experiments of the proposed SM are tested on the MovieLens datasets. Table 23 shows the details of collected datasets.

To demonstrate the performance of proposed SMs in different sparse datasets, the collected datasets are further divided into subsets. These subsets are created by randomly removing 10%, 20%, and 30% of ratings from Datasets 1 and 2, and 10%, 30%, and 50% of ratings from Dataset 3, respectively [3,46–48]. Brief details of these subsets are listed in Table 24.

Furthermore, in this paper, these deleted ratings are predicted by various CF algorithms and different accuracy/performance metrics have been used in the comparative analysis. These metrics are MAE, RMSE, precision, recall, F1-measure, and accuracy. The equations of these metrics are [3,7,46–48]:

$$MAE = \frac{\sum_{i=1}^{N} |p_i - \hat{q}_i|}{N} \tag{10}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (p_i - \hat{q}_i)^2}{N}} \tag{11}$$

From equations 10 and 11, the predicted and actual rating of item i is represented by p_i and $\hat{q}_i$, respectively. N denotes the total number of predicted items in the dataset.

$$Precision = \frac{\#t_p}{\#t_p + \#f_p} \tag{12}$$

$$Recall = \frac{\#t_p}{\#t_p + \#f_n} \tag{13}$$

$$F1 - Measure = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{14}$$

$$Accuracy = \frac{\#t_p + \#t_n}{\#t_p + \#t_n + \#f_p + \#f_n} \tag{15}$$

The ratings above 3 are addressed to be a high rating (recommended), and less than 3 is a low rating (not recommended) to determine the precision, recall, F1-measure, and accuracy. Symbol # denotes the 'number of'. The classification of the possible results, i.e., t_p, f_n, f_p, and t_n, is calculated using Table 25 [7,46,47].

Table 20 Similarity between target item and other item

Items	CSim	CSim^u(i,j)	ACSim	ACSim^u(i,j)	EDSim	EDSim^u(i,j)	PCSim	PCSim^u(i,j)	SCSim	SCSim^u(i,j)	BCSim	BCSim^u(i,j)
T_i-I_1	0.91	0.91	0.36	0.78	0.21	0.88	0.58	0.64	0.57	0.71	1.15	1.24
T_i-I_2	0.74	0.75	− 0.82	− 0.92	0.12	0.80	− 0.92	− 0.93	− 0.92	− 0.96	0.89	1.30
T_i-I_3	0.80	0.82	− 0.44	− 0.83	0.15	0.85	− 0.19	0.43	− 0.27	0.64	0.99	1.38
T_i-I_4	0.98	0.98	0.87	0.96	0.38	0.94	0.92	0.94	0.95	0.97	0.81	1.24
T_i-I_5	0.84	0.85	− 0.52	− 0.27	0.16	0.83	0.05	− 0.29	− 0.01	− 0.40	0.69	1.24
T_i-I_6	0.76	0.76	− 0.45	0.16	0.14	0.83	− 0.28	− 0.60	− 0.20	− 0.73	0.73	1.12
T_i-I_7	0.99	0.99	0.88	0.96	0.39	0.95	0.95	0.95	0.99	0.99	1.05	1.14
T_i-I_8	0.93	0.93	0.47	0.53	0.24	0.88	0.67	0.80	0.60	0.71	0.97	1.11
T_i-I_9	0.95	0.95	0.57	0.83	0.27	0.90	0.75	0.67	0.77	0.66	0.58	1.08
T_i-I_{10}	0.90	0.90	− 0.01	− 0.16	0.21	0.88	0.39	0.44	0.51	0.57	0.52	1.08

Table 21 Ranking of similar items for the target item

	Ranking using											
Items	CSim	CSim^u(i,j)	ACSim	ACSim^u(i,j)	EDSim	EDSim^u(i,j)	PCSim	PCSim^u(i,j)	SCSim	SCSim^u(i,j)	BCSim	BCSim^u(i,j)
T_i-I_1	5	5	5	4	5	6	5	5	5	5	1	3
T_i-I_2	10	10	10	10	10	10	10	10	10	10	5	2
T_i-I_3	8	8	7	9	8	7	9	7	9	7	3	1
T_i-I_4	2	2	2	1	2	2	2	2	2	2	6	4
T_i-I_5	7	7	9	8	7	9	7	8	7	8	8	5
T_i-I_6	9	9	8	7	9	8	8	9	8	9	7	7
T_i-I_7	1	1	1	2	1	1	1	1	1	1	2	6
T_i-I_8	4	4	4	5	4	4	4	3	4	3	4	8
T_i-I_9	3	3	3	3	3	3	3	4	3	4	9	9
T_i-I_{10}	6	6	6	6	6	5	6	6	6	6	10	10

Table 22 Predicted rating of target item

S. No.	Proposed similarity measures	Predicted rating
1	$CSim^u$(i,j) or $(CF\_C_I\_C_U)$	$2.57 \approx 2.5$
2	$ACSim^u$(i,j) or $(CF\_A_I\_A_U)$	$2.89 \approx 3.0$
3	$EDSim^u$(i,j) or $(CF\_E_I\_E_U)$	$3.58 \approx 3.5$
4	$PCSim^u$(i,j) or $(CF\_P_I\_P_U)$	$3.48 \approx 3.5$
5	$SCSim^u$(i,j) or $(CF\_S_I\_S_U)$	$2.80 \approx 3.0$
6	$BCSim^u$(i,j) or $(CF\_B_I\_B_U)$	$1.21 \approx 1.0$

Table 23 Details of the collected datasets

Dataset	# Users	# Items	# Ratings	Sparsity (%)	Rating domain
MovieLens [33] *ml-100k* (Dataset1)	943	1682	100000	93.695	1 to 5.0 with one increments
MovieLens [33] *ml-1m* (Dataset2)	6040	3952	1000209	95.809	1 to 5.0 with one increments
Film trust [33] (Dataset3)	1508	2071	35494	99.988	0.5 to 5.0 with half increments

6.2 Comparative analysis

Comparative results of this section can be divided into four parts.

- K. Choi and Y. Suh have proved that Pearson correlation, using item–item similarity as a weight in a user-based CF, attains more accuracy than other algorithms. Therefore, to measure the effectiveness of the proposed idea, the SM using Pearson correlation is compared with the CF algorithm proposed by K. Choi and Y. Suh, where user similarity using Pearson correlation is calculated by item similarity using Pearson correlation ((CF_P_P)) [27].
- To find the most accurate SM, this paper compares item-based CF with users similarity as weight calculated using cosine similarity, Pearson correlation, its variants, and modified Bhattacharyya coefficient ($CF\_C_I\_C_U$, $CF\_A_I\_A_U$, $CF\_E_I\_E_U$, $CF\_P_I\_P_U$, $CF\_S_I\_S_U$, and $CF\_B_I\_B_U$).
- This work demonstrates a comparison between the most accurate SM calculated in the previous part, and some recently used SMs in CF-based RS (A CF algorithm for density enrichment using Pearson correlation (CF_{DR}_PC), and CF algorithm using improved triangle similarity (CF_ITR)).
- Lastly, the comparison of complexities has been discussed among CF_P_P, $CF\_C_I\_C_U$, $CF\_A_I\_A_U$, $CF\_E_I\_E_U$, $CF\_P_I\_P_U$, $CF\_S_I\_S_U$, $CF\_B_I\_B_U$, CF_{DR}_PC, and CF_ITR.

6.2.1 Comparison of $CF\_P_I\_P_U$ and CF_P_P

Figures 2, 3, 4, 5, 6, 7, 8, 9, 10 represent the comparison between $CF\_P_I\_P_U$ and CF_P_P on the basis of various performance metrics in different datasets. From the observations, we collect results on different values of k in top-k neighbors.

Figures 2, 3, 4, 5, 6, 7 serve as significant evidence to choose the accurate SM between $CF\_P_I\_P_U$ and CF_P_P. We can notice that $CF\_P_I\_P_U$ attains less prediction errors than CF_P_P for datasets1 and 2 at different values of k-nearest neighbors. In dataset3, for less sparse subset, i.e., ML_7, $CF\_P_I\_P_U$ fails to outperform CF_P_P, but, for more sparse

Table 24 Used subsets in the experiments

Dataset	# Users (U)	# Items (I)	Subset	Density Index $\frac{\#R*100}{\#U*\#I}$	$\frac{\#\text{ Ratings}}{\#\text{ Users}}$	$\frac{\#\text{ Ratings}}{\#\text{ Items}}$
Dataset1	943	1682	ML_1	5.67	95.44	53.50
			ML_2	5.04	84.83	47.56
			ML_3	4.41	74.23	41.61
Dataset2	6040	3706	ML_4	4.021	149.03	242.90
			ML_5	3.57	132.47	215.91
			ML_6	3.12	115.91	188.922
Dataset3	1508	2071	ML_7	1.023	21.18	15.42
			ML_8	0.80	16.48	12.00
			ML_9	0.57	11.77	8.57

Table 25 Classification of the potential outcomes used in performance metrics

Ratings	Prediction	
	Recommended (Presumed Rating >= 3)	Not Recommended (Presumed Rating < 3)
Original Rating >= 3	t_p	f_n
Original Rating < 3	f_p	t_n

subsets, i.e., ML_8 and ML_9, it outperforms CF_P_P in terms of less prediction error at various values of k-nearest neighbors.

Figures 8, 9, 10 demonstrate the comparison between $CF\_P_I\_P_U$ and CF_P_P on the basis of precision, recall, F1-measure, and accuracy. It is observed that $CF\_P_I\_P_U$ has higher values of mentioned performance metrics than the CF_P_P in Datasets 1 and 2 at various sparsity. In Dataset 3, $CF\_P_I\_P_U$ has low precision, recall, F1-measure, and accuracy at 10% sparsity than CF_P_P , but higher precision, recall, F1-measure, and accuracy at 30% and 50% sparsity.

From all three datasets, recommendation accuracy goes in favor of $CF\_P_I\_P_U$. Hence, $CF\_P_I\_P_U$ is better SM than CF_P_P, and utilization of user–user similarity as a weight in an item-based CF is a judicious approach.

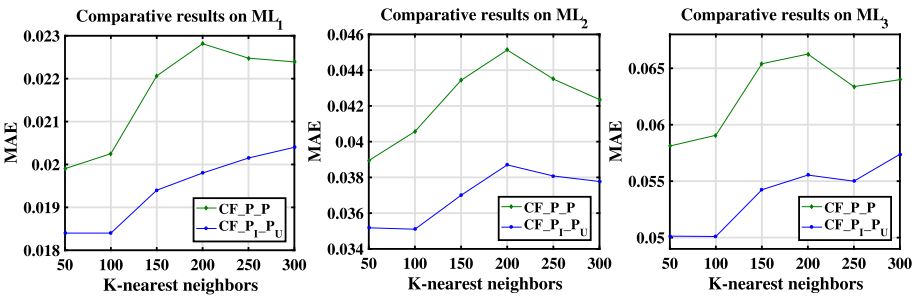

Fig. 2 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on MAE values at Dataset1

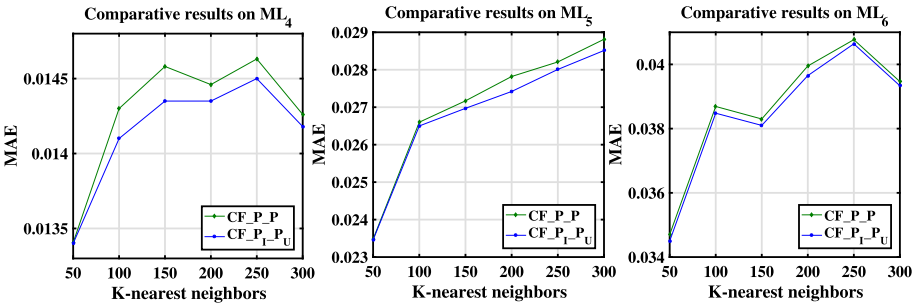

Fig. 3 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on MAE values at Dataset 2

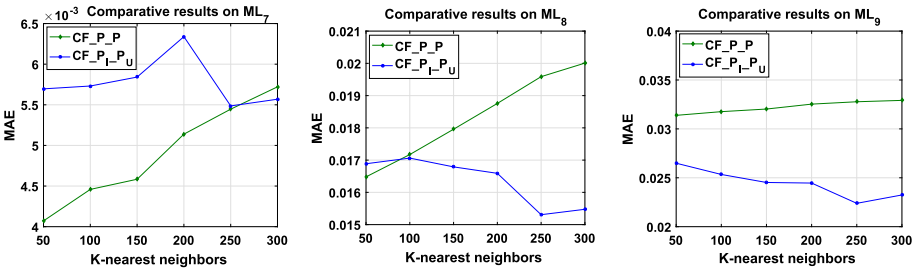

Fig. 4 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on MAE values at Dataset 3

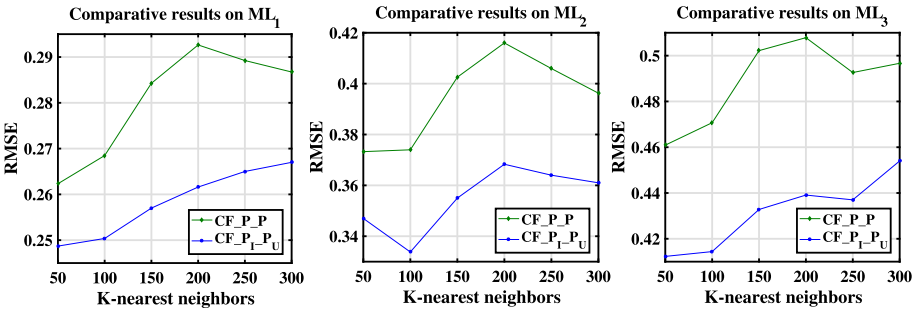

Fig. 5 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on RMSE values at Dataset 1

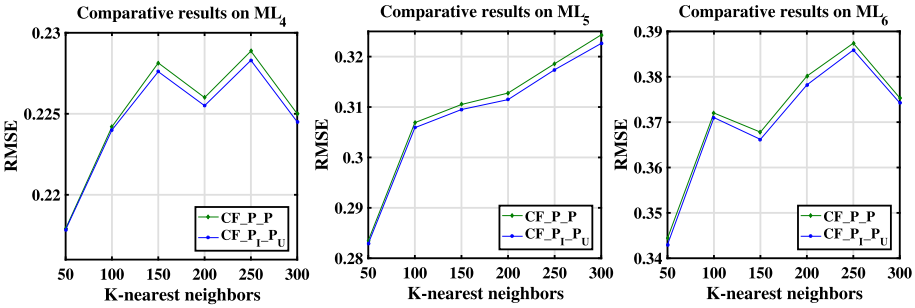

Fig. 6 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on RMSE values at Dataset 2

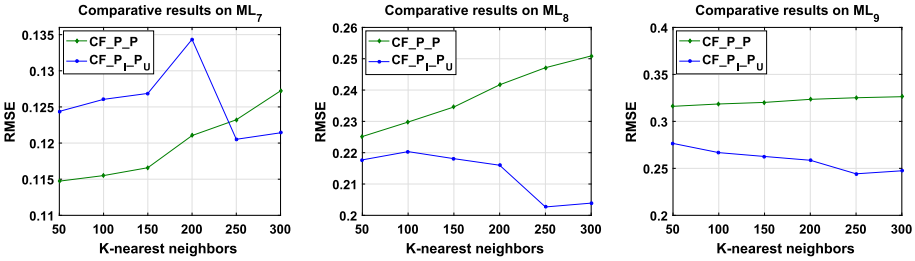

Fig. 7 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on RMSE values at Dataset 3

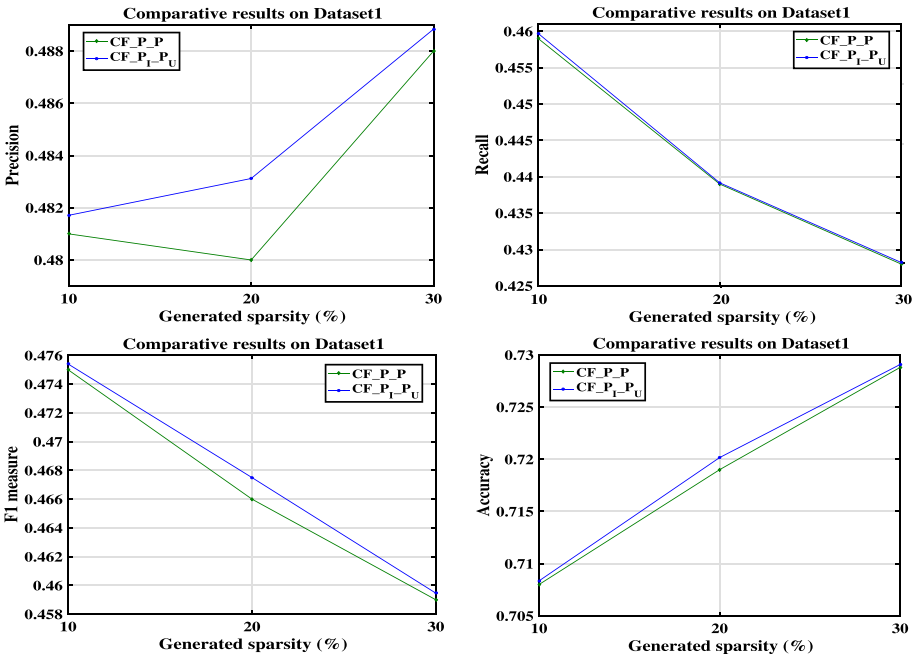

Fig. 8 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on precision, recall, F1-measure, and accuracy at Dataset 1

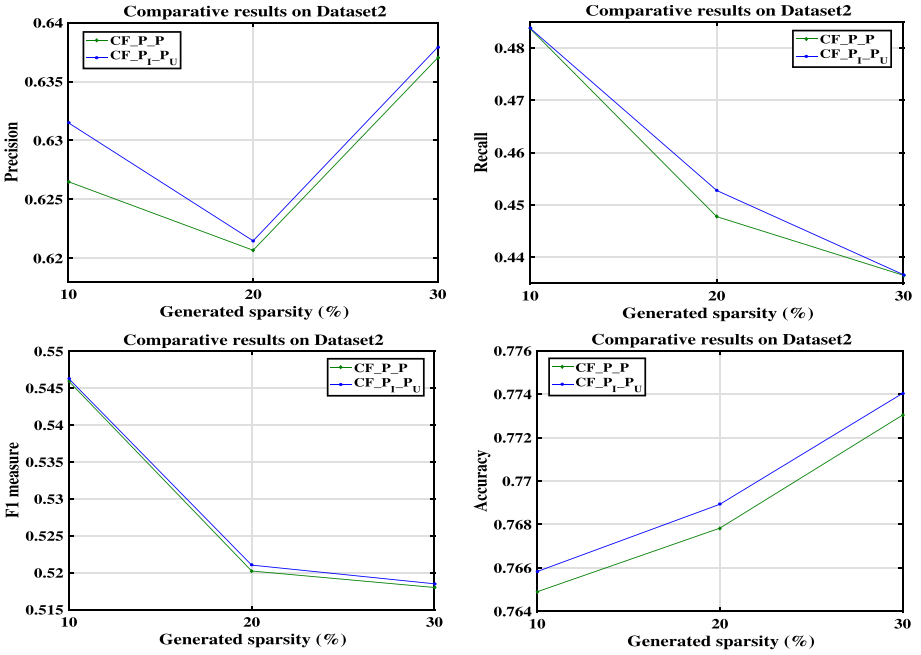

Fig. 9 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on precision, recall, F1-measure, and accuracy at Dataset 2

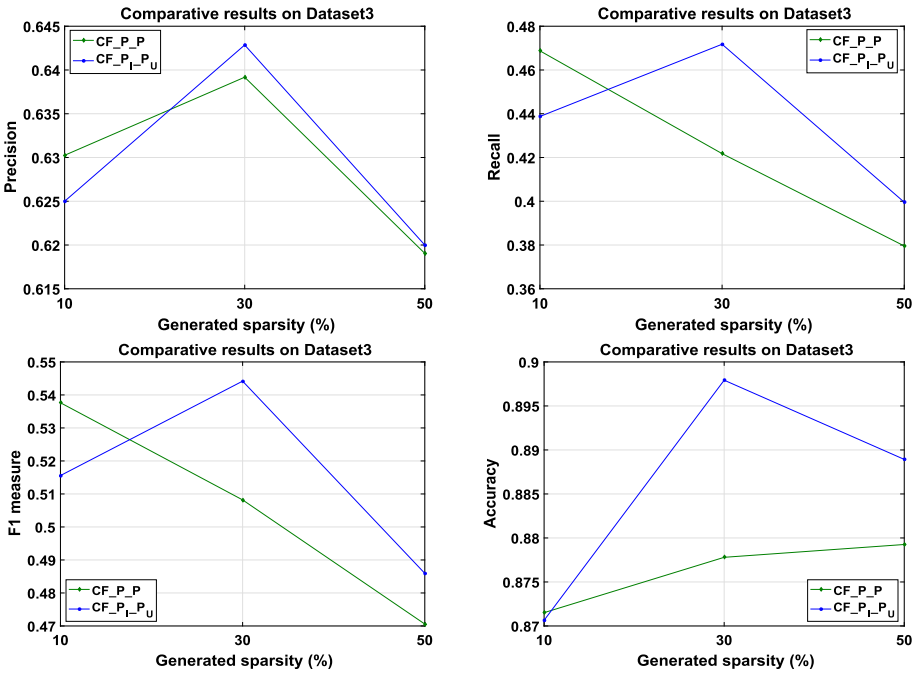

Fig. 10 Comparison of $CF\_P_I\_P_U$ and CF_P_P based on precision, recall, F1-measure, and accuracy at Dataset 3

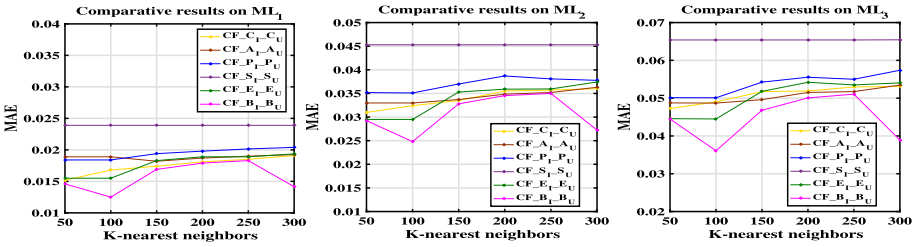

Fig. 11 Comparison based on MAE values at Dataset 1

6.2.2 Comparison of proposed similarity function using traditional similarity measures

To evaluate the performance of proposed SM under different traditional SMs, this section demonstrates the comparative analysis based on MAE, RMSE, precision, recall, F1-measure, and accuracy. In the comparative analysis, we compute all performance metrics at various k values (i.e., 50, 100, 150, 200, 250, and 300) in Top-k neighbors. The performance of $CF\_B_I\_B_U$ is found to be relatively better than others with respect to MAE values as shown in Figs. 11, 12, 13.

In Figs. 14, 15, 16, $CF\_B_I\_B_U$ demonstrates comparatively low RMSE values than the other approaches for all neighbors at all datasets. Figures 17, 18, 19 highlight the comparison among $CF\_C_I\_C_U$, $CF\_A_I\_A_U$, $CF\_E_I\_E_U$, $CF\_P_I\_P_U$, $CF\_S_I\_S_U$, and $CF\_B_I\_B_U$ on the basis of precision, recall, F1-measure, and accuracy. It is evident that $CF\_B_I\_B_U$ attains better prediction accuracy than others in all datasets and recommendation accuracy

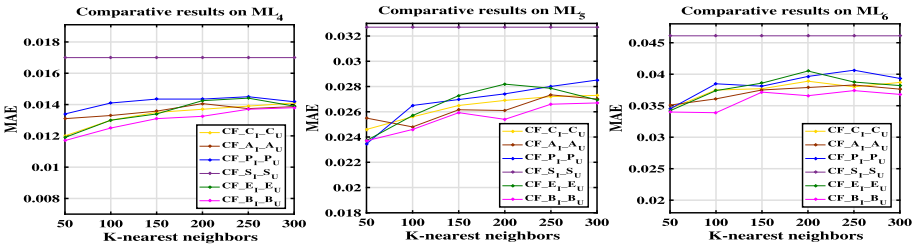

Fig. 12 Comparison based on MAE values at Dataset 2

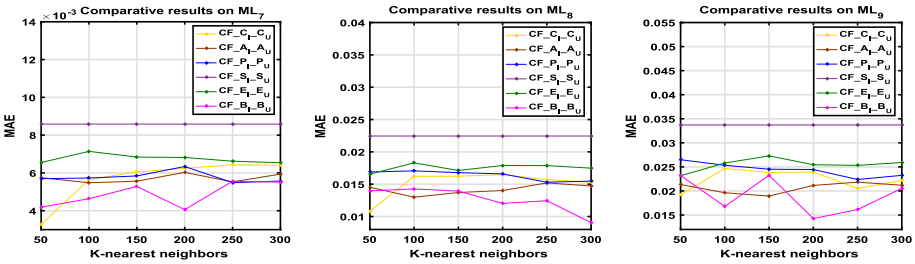

Fig. 13 Comparison based on MAE values at Dataset 3

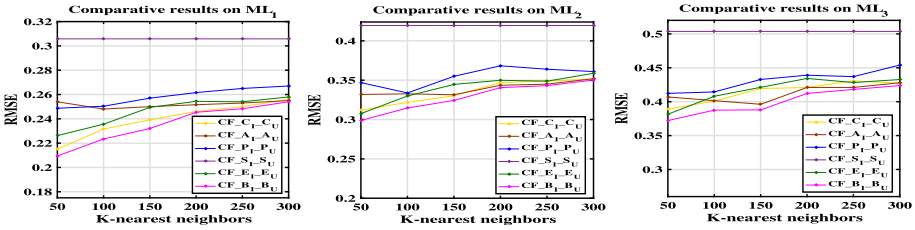

Fig. 14 Comparison based on RMSE values at Dataset 1

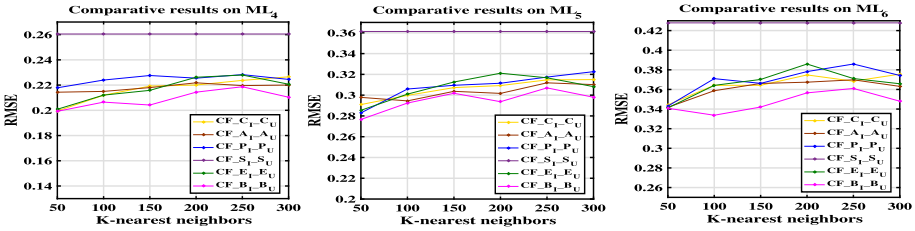

Fig. 15 Comparison based on RMSE values at Dataset 2

further goes in favor of $CF\_B_I\_B_U$. Hence, $CF\_B_I\_B_U$ outperforms other similarity functions.

6.2.3 Comparison of $CF\_B_I\_B_U$, CF_{DR}_PC, and CF_ITR

Figures 20, 21, 22, 23, 24, 25, 26, 27, 28 represent the comparison among $CF\_B_I\_B_U$, CF_{DR}_PC, and CF_ITR based on various accuracy metrics at different datasets. From

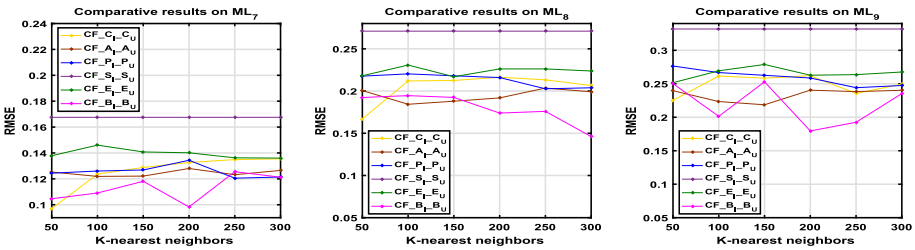

Fig. 16 Comparison based on RMSE values at Dataset 3

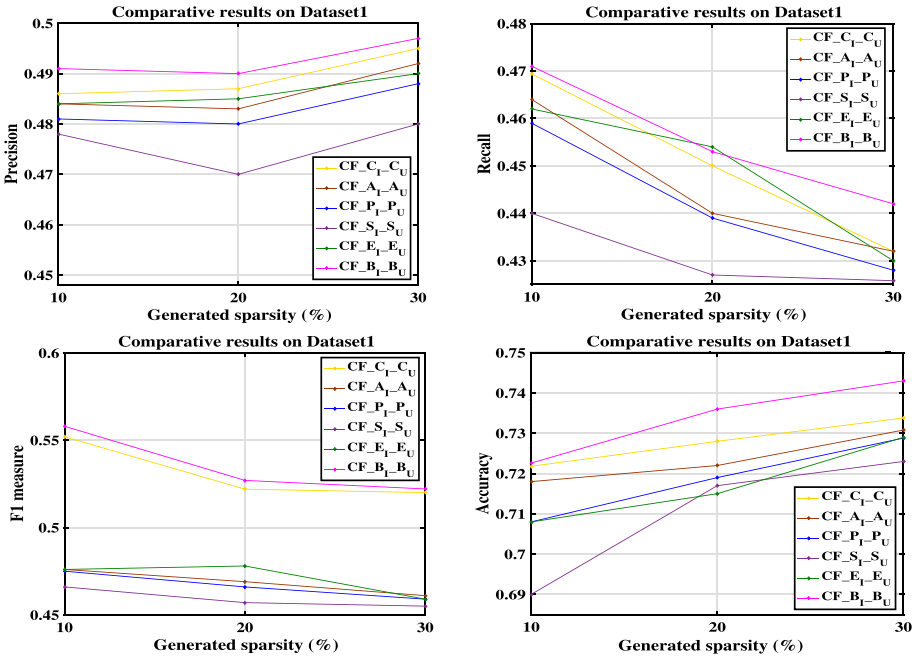

Fig. 17 Comparison among $CF\_C_I\_C_U$, $CF\_A_I\_A_U$, $CF\_E_I\_E_U$, $CF\_P_I\_P_U$, $CF\_S_I\_S_U$, and $CF\_B_I\_B_U$ at Dataset 1

Figs. 20, 21, 22, 23, 24, 25, we notice that in most of the cases, $CF\_B_I\_B_U$ has low MAE and low RMSE value as compared to CF_{DR}_PC, and CF_ITR. Figures 26, 27, 28 depict the comparison based on precision, recall, F1-measure, and accuracy at three datasets. Here, $CF\_B_I\_B_U$ attains more precision, recall, F1-measure and accuracy than other recently used SMs for all datasets.

6.2.4 Complexity comparison

Table 26 shows the comparison among various CF algorithms based on their computational complexity. Here, m and n denote the total numbers of users and items, respectively.

From Table 26, we observe that $CF\_B_I\_B_U$ takes more execution time than other algorithms due to its high complexity.

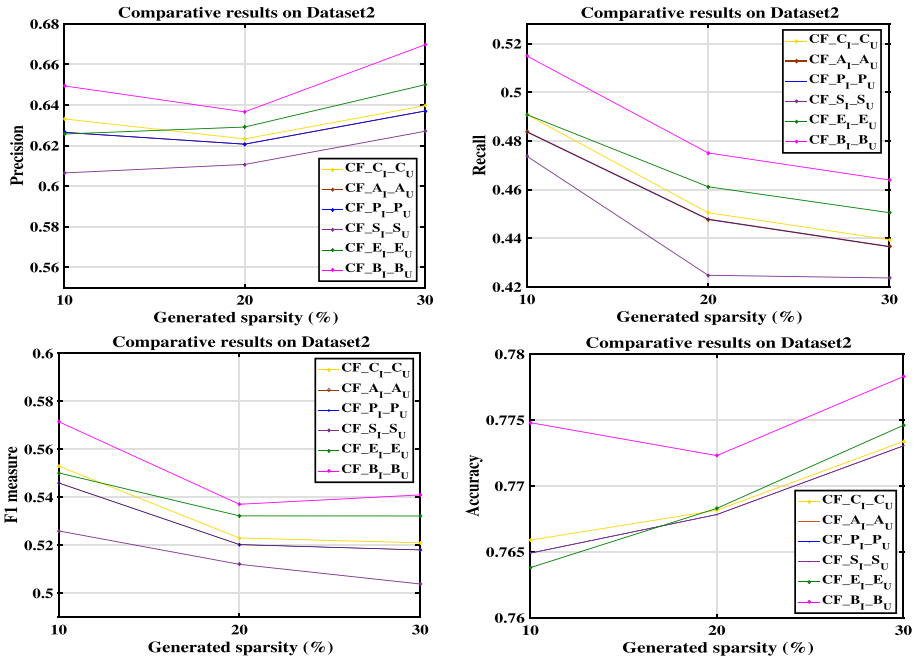

Fig. 18 Comparison among $CF\_C_I\_C_U$, $CF\_A_I\_A_U$, $CF\_E_I\_E_U$, $CF\_P_I\_P_U$, $CF\_S_I\_S_U$, and $CF\_B_I\_B_U$ at Dataset 2

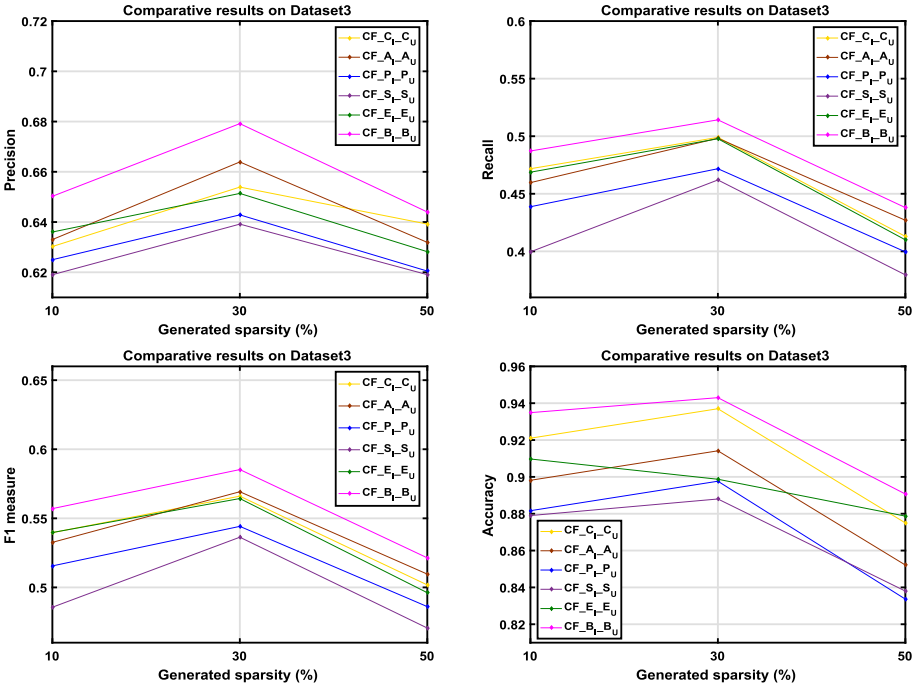

Fig. 19 Comparison among $CF\_C_I\_C_U$, $CF\_A_I\_A_U$, $CF\_E_I\_E_U$, $CF\_P_I\_P_U$, $CF\_S_I\_S_U$, and $CF\_B_I\_B_U$ at Dataset 3

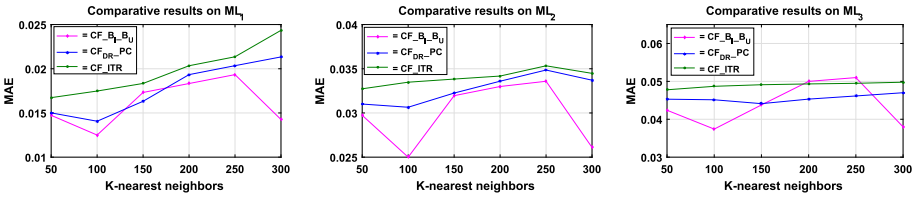

Fig. 20 Comparison based on MAE values at Dataset 1

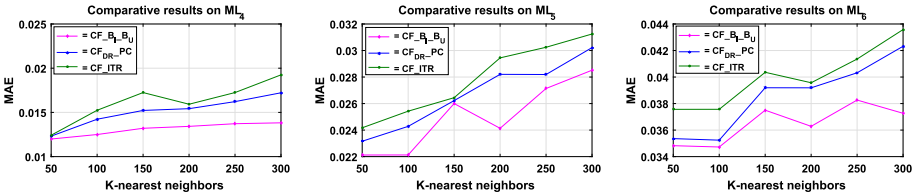

Fig. 21 Comparison based on MAE values at Dataset 2

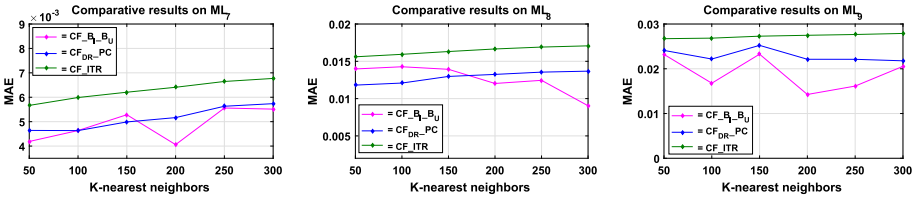

Fig. 22 Comparison based on MAE values at Dataset 3

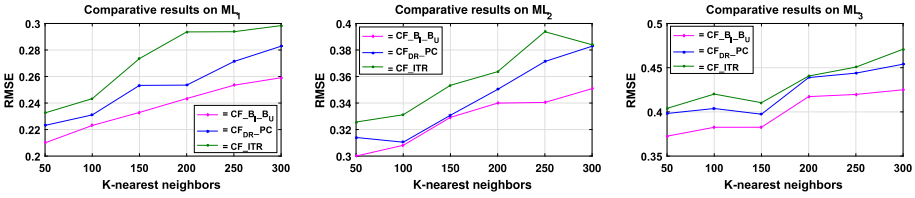

Fig. 23 Comparison based on RMSE values at Dataset 1

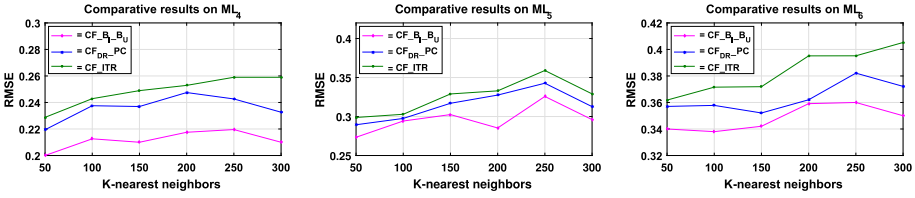

Fig. 24 Comparison based on RMSE values at Dataset 2

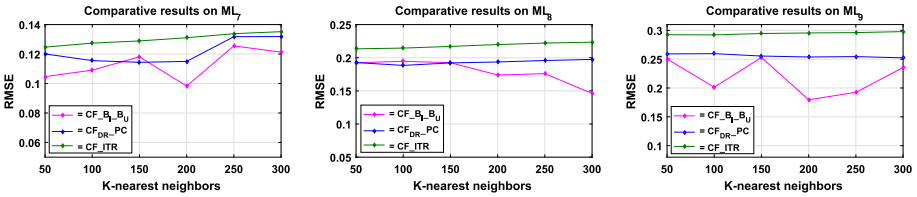

Fig. 25 Comparison based on RMSE values at Dataset 3

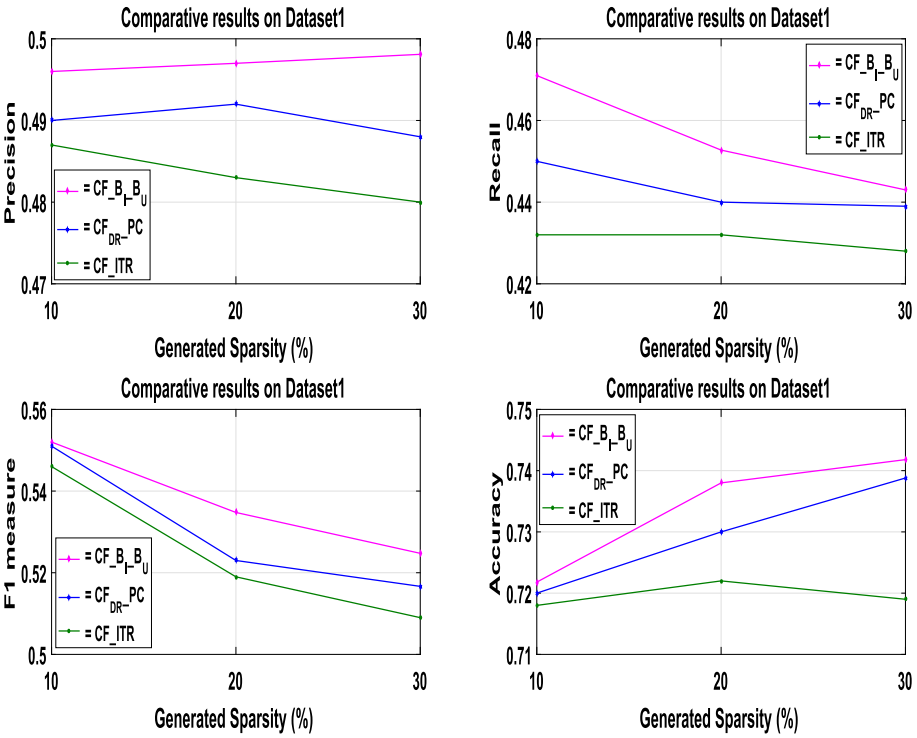

Fig. 26 Comparison among $CF\_B_I\_B_U$, CF_{DR}_PC, and CF_ITR at Dataset 1

Table 26 Complexity comparison

Algorithm	Running Complexity
CF_P_P	O(mn(m+n))
$CF\_C_I\_C_U$	O(mn(m+n))
$CF\_A_I\_A_U$	O(mn(m+n))
$CF\_E_I\_E_U$	O(mn(m+n))
$CF\_P_I\_P_U$	O(mn(m+n))
$CF\_S_I\_S_U$	O(mn(m+n))
$CF\_B_I\_B_U$	O(m^2n^2)
CF_{DR}_PC	O(m^2n)
CF_ITR	O(m^2n)

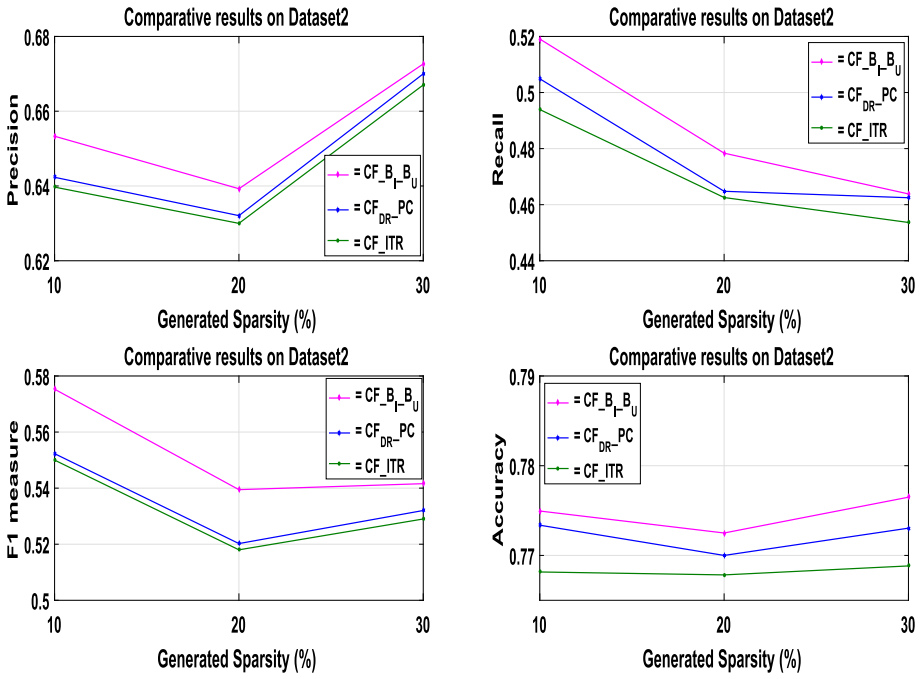

Fig. 27 Comparison among $CF\_B_I\_B_U$, CF_{DR}_PC, and CF_ITR at Dataset 2

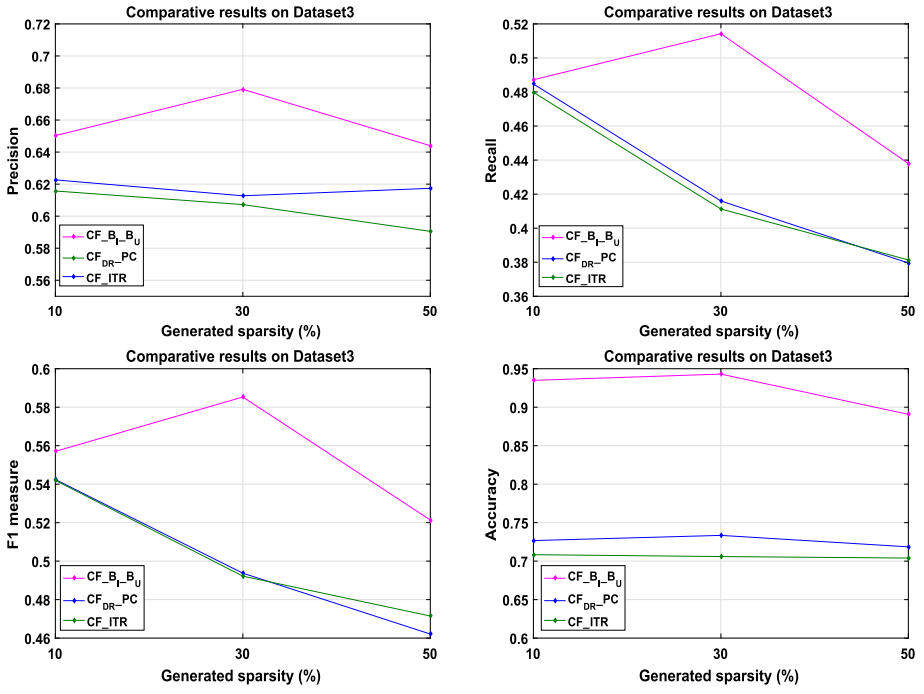

Fig. 28 Comparison among $CF\_B_I\_B_U$, CF_{DR}_PC, and CF_ITR at Dataset 3

Hence, through the comparative analysis with various sparse datasets, we can rightly conclude that the $CF\_B_I\_B_U$ performs better than other CF algorithms in spite of getting high complexity. However, various e-commerce websites adopt various cloud offerings like Amazon SageMaker, which utilizes libraries for performing the computation on parallel GPU instances. Common cloud offerings for techniques like MapReduce can also be employed to achieve the required parallelism in methods. Therefore, the proposed approach can be implemented in parallel to resolve the scalability issue.

7 Conclusions

Classical memory-based recommender systems were outmoded by model-based systems in the first place due to their inability to handle sparse datasets and due to inherent drawbacks of traditional similarity measures. Owing to the simplicity, serendipitous nature, and interpretability of memory-based recommendation systems, addressing their key blockers will make them the most attractive approach and open tremendous scope for promising strides. In this paper, the proposed similarity function mitigates the issue related to the sparsity of datasets and factors in different magnitudes of rating vectors. The proposed function first computes the similarity between two users, and then, it utilizes the weight of each user in the calculation of similarity between a target item and other items. The experimental results show that this function outperforms various modified similarity measures and the well-known user similarity function by K. Choi and Y. Suh, in terms of lower MAE and RMSE values and higher precision, recall, F1-measure, and accuracy over MovieLens and Film trust datasets.

The future work may aim to address the chronic cold-start problem, i.e., recommending without a historical record and finding a new similarity measure beyond direction orientation. Furthermore, future work will be focused on parallel and distributed collaborative filtering algorithms to mitigate the scalability issue and to reduce the complexity of the proposed approach.

References

1. Zheng K, Yang Z, Zhang K, Chatzimisios P, Yang K, Xiang W (2016) Big data-driven optimization for mobile networks toward 5g. IEEE Netw 30:44–51
2. Su X, Khoshgoftaar TM (2009) A survey of collaborative filtering techniques. Adv Artif Intel 4(2–4):2
3. Singh PK, Pramanik PKD, Choudhury P (2018) A comparative study of different similarity metrics in highly sparse rating dataset, In: V. Balas, N. Sharma, A. Chakrabarti (Eds.), Data Management, Analytics and Innovation (vol. 2), Vol. 839 of Advances in Intelligent Systems and Computing, Springer, pp. 45–60. https://doi.org/10.1007/978-981-13-1274-8_4
4. Li D, Miao C, Chu S, Mallen J, Yoshioka T, Srivastava P (2018) Stable Matrix Approximation for Top-N Recommendation on Implicit Feedback Data
5. Jorge AM, Vinagre J, Domingues M, Gama J, Soares C, Matuszyk P, Spiliopoulou M (2017) Scalable Online Top-N Recommender Systems. Springer International Publishing, Berlin
6. Schafer JB, Konstan J, Iedl J (1999) Recommender systems in e-commerce, In: Proceedings of the 1st ACM Conference on Electronic Commerce, ACM, pp. 158–166
7. Singh PK, Pramanik PKD, Choudhury P (2019) Collaborative filtering in recommender systems: Technicalities, challenges, applications and research trends, In: G. Shrivastava, S. L. Peng, H. Bansal, K. Sharma, M. Sharma (Eds.), New Age Analytics: Transforming Internet, Apple Academic Press
8. Sarwar B, Karypis G, Konstan J, Riedl J (2001) Item-based collaborative filtering recommendation algorithms, In: Proceedings of the 10th International Conference on World Wide Web, ACM, pp. 285–295
9. Kant S, Mahara T (2018) Merging user and item based collaborative filtering to alleviate data sparsity. Int J Syst Assurance Eng Manag 9(1):173–179

10. Comparison of user-based and item-based collaborative filtering, https://medium.com/@www.bbb8510/comparison-of-user-based-and-item-based-collaborative-filtering-f58a1c8a3f1d, online; Accessed 30-April-2019
11. Yang Z, Wu B, Zheng K, Wang X, Lei L (2016) A survey of collaborative filtering-based recommender systems for mobile internet applications. IEEE Access 4:3273–3287
12. Boström P, Filipsson M (2017) Comparison of user based and item based collaborative filtering recommendation services
13. Panda SK, Bhoi SK, Singh M (2020) A collaborative filtering recommendation algorithm based on normalization approach, J Ambient Intel Hum Comput 1–23
14. Pal A, Parhi P, Aggarwal M (2017) An improved content based collaborative filtering algorithm for movie recommendations, In: Tenth International Conference on Contemporary Computing (IC3), IEEE, pp. 1–3
15. Liu H, Hu Z, Mian AU, Tian H, Zhu X (2014) A new user similarity model to improve the accuracy of collaborative filtering. Knowledge-Based Syst 56:156–166
16. Ghazarian S, Nematbakhsh MA (2015) Enhancing memory-based collaborative filtering for group recommender systems. Expert Syst Appl 42(7):3801–3812
17. Li C, He K Cbmr: An optimized mapreduce for item-based collaborative filtering recommendation algorithm with empirical analysis, Concurrency and Computation: Practice and Experience 29(10)
18. Vizine Pereira AL, Hruschka ER (2015) Simultaneous co-clustering and learning to address the cold start problem in recommender systems. Knowledge-Based Syst 82:11–19
19. Shambour Q, Hourani M, Fraihat S An item-based multi-criteria collaborative filtering algorithm for personalized recommender systems, Int J Adv Computer Sci Appl, 7
20. Karydi E, Margaritis KG (2012) Parallel implementation of the slope one algorithm for collaborative filtering, In: 16th Panhellenic Conference on Informatics, pp. 174–179
21. Wang Z, Liu Y, Ma P (2014) A cuda-enabled parallel implementation of collaborative filtering. Procedia Comput Sci 30:66–74
22. Karydi E, Margaritis K (2016) Parallel and distributed collaborative filtering: a survey. ACM Computing Surveys (CSUR) 49(2):1–41
23. Sardianos C, Ballas Papadatos G, Varlamis I (2019) Optimizing parallel collaborative filtering approaches for improving recommendation systems performance. Information 10(5):155
24. Li D, Chen C, Lv Q, Shang L, Zhao Y, Lu T, Gu N (2016) An algorithm for efficient privacy-preserving item-based collaborative filtering. Future Generat Comput Syst 55:311–320
25. Bilge A, Kaleli C (2014) A multi-criteria item-based collaborative filtering framework, In: 11th International Joint Conference on Computer Science and Software Engineering, pp. 18–22
26. Adomavicius G, Kwon Y (2007) New recommendation techniques for multicriteria rating systems. IEEE Intel Syst 22(3):48–55
27. Choi K, Suh Y (2013) A new similarity function for selecting neighbors for each target item in collaborative filtering. Knowledge-Based Syst 37:146–153
28. Bobadilla J, Hernando A, Ortega F, Gutiérrez A (2012) Collaborative filtering based on significances. Inf Sci 185(1):1–17
29. Ricci F, Rokach L, Shapira B, Kantor PB (2010) Recommender Systems Handbook, 1st edn. Springer-Verlag, New York Inc
30. Patra BK, Launonen R, Ollikainen V, Nandi S (2015) A new similarity measure using bhattacharyya coefficient for collaborative filtering in sparse data. Knowledge-Based Systems 82:163–177
31. Su H, Wang C, Zhu Y, Yan B, Zheng H (2014) Parallel collaborative filtering recommendation model based on expand-vector, in: International Conference on Multisensor Fusion and Information Integration for Intelligent Systems (MFI), pp. 1–6
32. Tan Z, He L (2017) An efficient similarity measure for user-based collaborative filtering recommender systems inspired by the physical resonance principle, IEEE Access PP 1–1
33. Singh PK, Sinha M, Das S, Choudhury P (2020) Enhancing recommendation accuracy of item-based collaborative filtering using bhattacharyya coefficient and most similar item, Applied Intelligence 1–24
34. Goudail F, Réfrégier P, Delyon G (2004) Bhattacharyya distance as a contrast parameter for statistical processing of noisy optical images. J Opt Soc Am A 21(7):1231–1240. https://doi.org/10.1364/JOSAA.21.001231
35. Toussaint G (1972) Comments on "the divergence and bhattacharyya distance measures in signal selection.". IEEE Trans Commun 20(3):485–485
36. Ahn HJ (2008) A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. Inf Sci 178(1):37–51
37. Sun H-F, Chen J-L, Yu G, Liu C-C, Peng Y, Chen G, Cheng B (2012) Jacuod: a new similarity measurement for collaborative filtering. J Computer Sci Technol 27(6):1252–1260

38. Wang W, Lu J, Zhang G (2014) A new similarity measure-based collaborative filtering approach for recommender systems, In: Foundations of Intelligent Systems, Springer, pp. 443–452
39. Gazdar A, Hidri L (2020) A new similarity measure for collaborative filtering based recommender systems. Knowledge-Based Syst 188:105058
40. Margaris D, Spiliotopoulos D, Karagiorgos G, Vassilakis C (2020) An algorithm for density enrichment of sparse collaborative filtering datasets using robust predictions as derived ratings. Algorithms 13(7):174
41. Iftikhar A, Ghazanfar MA, Ayub M, Mehmood Z, Maqsood M (2020) An improved product recommendation method for collaborative filtering. IEEE Access 8:123841–123857
42. Boratto L, Carta S, Fenu G (2017) Investigating the role of the rating prediction task in granularity-based group recommender systems and big data scenarios. Inf Sci 378:424–443
43. Koohi H, Kiani K (2017) A new method to find neighbor users that improves the performance of collaborative filtering. Expert Syst Appl 83:30–39
44. Liu Y, Feng, Lu J (2017) Collaborative filtering algorithm based on rating distance, In: Proceedings of the 11th International Conference on Ubiquitous Information Management and Communication, ACM, pp. 66: 1–66:7
45. Aggarwal CC (2016) Neighborhood-based collaborative filtering. Springer, Berlin
46. Singh PK, Setta S, Pramanik PKD, Choudhury P (2019) Improving the accuracy of collaborative filtering based recommendations by considering the temporal variance of top-n neighbors, In: Proceedings of the International Conference on Innovative Computing and Communication (ICICC-2019), Ostrava, Czech Republic
47. Singh PK, Pramanik PKD, Debnath NC, Choudhury P (2019) A novel neighborhood calculation method by assessing users' varying preferences in collaborative filtering, In: Proceedings of the 34th International Conference on Computers and Their Applications (CATA 2019), no. 58 in EPiC Series in Computing, Honolulu, Hawaii, pp. 345–355. https://doi.org/10.29007/3xfj
48. Singh PK, Pramanik PKD, Choudhury P (2019) An improved similarity calculation method for collaborative filtering-based recommendation, considering the liking and disliking of categorical attributes of items. J Inf Optim Sci 40(2):397–412. https://doi.org/10.1080/02522667.20191580881