I.J. Information Technology and Computer Science, **2019, 6, 37-49**
Published Online June 2019 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2019.06.05

A New Similarity Measure Based on Simple Matching Coefficient for Improving the Accuracy of Collaborative Recommendations

Vijay Verma
Computer Engineering Department, National Institute of Technology, Kurukshetra, Haryana, India-136119
E-mail: vermavijay@nitkkr.ac.in

Rajesh Kumar Aggarwal
Computer Engineering Department, National Institute of Technology, Kurukshetra, Haryana, India-136119
E-mail: rka15969@gmail.com

Received: 13 February 2019; Accepted: 22 March 2019; Published: 08 June 2019

***Abstract*—**Recommender Systems (RSs) are essential tools of an e-commerce portal in making intelligent decisions for an individual to obtain product recommendations. Neighborhood-based approaches are traditional techniques for collaborative recommendations and are very popular due to their simplicity and efficiency. Neighborhood-based recommender systems use numerous kinds of similarity measures for finding similar users or items. However, the existing similarity measures function only on common ratings between a pair of users (i.e. ignore the uncommon ratings) thus do not utilize all ratings made by a pair of users. Furthermore, the existing similarity measures may either provide inadequate results in many situations that frequently occur in sparse data or involve very complex calculations. Therefore, there is a compelling need to define a similarity measure that can deal with such issues. This research proposes a new similarity measure for defining the similarities between users or items by using the rating data available in the user-item matrix. Firstly, we describe a way for applying the simple matching coefficient (SMC) to the common ratings between users or items. Secondly, the structural information between the rating vectors is exploited using the Jaccard index. Finally, these two factors are leveraged to define the proposed similarity measure for better recommendation accuracy. For evaluating the effectiveness of the proposed method, several experiments have been performed using standardized benchmark datasets (MovieLens-1M, 10M, and 20M). Results obtained demonstrate that the proposed method provides better predictive accuracy (in terms of MAE and RMSE) along with improved classification accuracy (in terms of precision-recall).

***Index Terms*—**Recommender Systems, Collaborative Filtering, Similarity Measures, Simple Matching Coefficient, Jaccard index, E-commerce.

I. Introduction

Nowadays, the world wide web (or simply web) serves as an important means for e-commerce businesses. The rapid growth of e-commerce businesses presents a potentially overwhelming number of alternatives to their users, this frequently results in the information overload problem. Recommender Systems (RSs) are the critical tools for avoiding information overload problem and provide useful suggestions that may be relevant to the users [1]. RSs help individuals by providing personalized recommendations by exploiting different sources of information related to users, items, and their interactions [2]. A recommender system facilitates to achieve a diverse set of goals for both the service provider and its end users. The primary goal of an RS on behalf of the service provider is increasing the revenue (by increasing the product sales) while on behalf of the end users, the primary goal of an RS is finding some useful products.

In its simplest form, the recommendation problem can be defined as providing a list of items or finding the best item for a user [3]. Primarily, there are following three broad ways to classify the RSs [4].

- Content-based Recommender Systems (CRS): The basic principle of a content-based recommender system is to recommend those items that are similar to the ones liked by a user in the past[5, 6]. For example, if a user listens to pop music, then the system may recommend the songs of the pop genre.
- Collaborative Filtering-based Recommender Systems (CFRS): The basic principle of a collaborative filtering-based recommender system is to recommend those items that other similar users liked in the past [7, 8].

- Hybrid recommendations: These methods hybridize collaborative and content-based approaches for more effective recommendations in diverse applications [9, 10].

Collaborative filtering is the most popular and widely used technique in recommender systems. A detailed discussion on CFRSs is provided in various survey articles published in the past [11, 12, 13]. Article [14] suggests that collaborative recommender systems can be further categorized into two broad classes: memory-based and model-based approaches. Memory-based algorithms [15, 16] are basically heuristic in nature and provide recommendations using the entire collection of rating data available in the UI-matrix. Model-based algorithms [17, 18, 19, 20, 21] learn a model from the rating data before providing any recommendations to users [22].

Among all CFRSs, the neighborhood-based algorithms (e.g., k-Nearest Neighbors) are the traditional ways of providing recommendations [23]. Finding similar users or items is the core component of the neighborhood-based collaborative recommendations since the fundamental assumption behind these approaches is that similar users demonstrate similar interests whereas similar items draw similar rating patterns [24]. Therefore, there exist numerous ways to define the similarity between users or items in the RS literature. Basically, there are two types of neighborhood-based approaches.

- User-based collaborative Filtering (UBCF): the main idea is as follows: firstly, identify the similar users (also called peer-users or nearest neighbors) who displayed similar preferences to those of an active user in the past. Then, the ratings provided by these similar users are used to provide recommendations.
- Item-based collaborative Filtering (IBCF): In this case, for estimating the preference value for an item i by an active u, firstly, determine a set of items which are similar to item i, then, the ratings received by these similar items from the user u are utilized for recommendations.

One substantial difference between UBCF and IBCF algorithms is that in the former case ratings of peer users are utilized whereas in the latter case active user's own ratings are used for prediction purpose. With respect to the UI-matrix, UBCF approaches define the similarities among rows (or users) whereas IBCF approaches define similarities among columns (or items) as shown in Fig. 1.

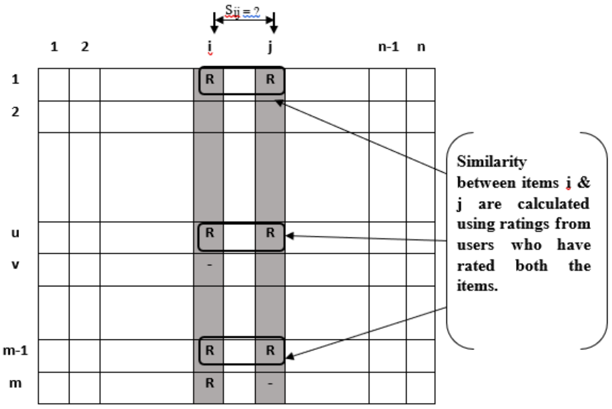

Sij = ?
1
2
i
j
n-1
n
1
2
u
v
m-1
m
R
R
R
R
-
R
R
R
-
Similarity between items i & j are calculated using ratings from users who have rated both the items.

Fig.1. An example scenario for calculating item-item similarity in IBCF.

Traditionally, various statistical measures have been utilized to define the similarity between users and items such as Pearson Correlation Coefficient (PCC)[25, 26], Constrained Pearson Correlation Coefficient(CPCC)[8], Mean Squared Difference(MSD)[8]. Measures from the linear algebra are also utilized to model the similarity, for example, the cosine similarity (COS)[14] calculates the angle between two rating vectors for representing the similarity. Additionally, heuristic-based similarity measures, Proximity-Impact-Popularity (PIP-measure) [27], and Proximity-Significance-Singularity (PSS-measure) [28] are also introduced by the researchers. Authors Bobadilla et al. have proposed various similarity measures by exploiting the different contextual information [29, 30, 31, 32].Patra et al. [33] have adopted the Bhattacharyya coefficient to define a new similarity measure that also manages the data-sparsity problem. Recently, the Jaccard similarity index [34] has been modified to relevant Jaccard similarity [35] for efficient recommendations.

Different types of similarity measures either assist in improving the various RS goals such as accuracy, diversity, novelty, serendipity, etc. or deal with various RS problems such as data sparsity cold start, etc. [36]. However, accomplishing single goal alone is not enough for effective and satisfying users' experience, therefore, an RS designer has to attain various and possibly conflicting goals. As an example, there are state-of-art model-based approaches which provide better predictive accuracy than the neighborhood-based approaches, however, accuracy alone is not sufficient for effective and satisfying users 'experience.

Furthermore, the neighborhood-based approaches provide serendipitous recommendations i.e. recommending items that are absolutely different with a factor of lucky discovery. The concept of serendipity augments the notion of novelty by including a factor of surprise [37]. Therefore, finding new similarity measures for neighborhood-based methods is an active thread of research among the researchers. Additionally, the existing CFRS can be easily refurbished by only replacing the similarity measurement module along with the following advantages [2].

- Simplicity: very simple to implement and require just one parameter i.e. the number of neighbors, generally.
- Efficiency: very efficient in term of performance and memory as there is no costly training phase.
- Justifiability: capable to provide concise explanations for the recommendations so that users can better understand the system.

Since accuracy is the most discussed and the most examined goal for designing an RS because if a system predicts users' interests accurately then it would be more useful for the users.

Table 1. The notations used

Symbol	Meaning
U	All the users of the system
I	All the items available in the system
R	All the ratings made by the users for the items
S	All possible preferences that can be expressed by the users such as S= {1,2,3,4,5}
r_{ui}	The rating value given by a user u ϵ U to a particular item i
I_u	The set of items rated by a particular user u
U_i	The set of users who have rated a particular item i
I_{uv}	The set of items rated by both the users u and v i.e. ($I_u \cap I_v$)
U_{ij}	The set of users who have rated both the items i and j i.e. ($U_i \cap U_j$)

Table 2. The traditional similarity measures

Ref.	Measure Name	Definitive Formula	Methodology (in brief)
[25], [26]	Pearson Correlation Coefficient (PCC)	$PCC(u,v)=\frac{\sum_{i\in I_{uv}}(r_{ui}-\bar{r}_u)(r_{vi}-\bar{r}_v)}{\sqrt{\sum_{i\in I_{uv}}(r_{ui}-\bar{r}_u)^2\sum_{i\in I_{uv}}(r_{vi}-\bar{r}_v)^2}}$	It measures the linear relationship between the rating vectors and displays the degree to which two rating vectors are linearly related.
[8]	Constrained Pearson Correlation Coefficient (CPCC)	$CPCC(u,v)=\frac{\sum_{i\in I_{uv}}(r_{ui}-r_{med})(r_{vi}-r_{med})}{\sqrt{\sum_{i\in I_{uv}}(r_{ui}-r_{med})^2\sum_{i\in I_{uv}}(r_{vi}-r_{med})^2}}$	It adapts the traditional PCC by using the median value (of the rating scale) in place of average rating.
[8]	Mean Squared Difference (MSD)	$MSD(u,v)=1-\frac{\sum_{i\in I_{uv}}(r_{ui}-r_{vi})^2}{\lvert I_{uv}\rvert}$	It is the arithmetic mean of the squares of differences between corresponding values of two rating vectors.
[14]	Cosine Similarity (COS)	$COS(u,v)=\frac{\sum_{i\in I_{uv}}(r_{ui})(r_{vi})}{\sqrt{\sum_{i\in I_{uv}}(r_{ui})^2\sum_{i\in I_{uv}}(r_{vi})^2}}$	It measures the cosine of the angle between the rating vectors.
[40]	Adjusted Cosine Similarity (ACOS)	$ACOS(i,j)=\frac{\sum_{u\in U_{ij}}(r_{ui}-\bar{r}_u)(r_{uj}-\bar{r}_u)}{\sqrt{\sum_{u\in U_{ij}}(r_{ui}-\bar{r}_u)^2\sum_{u\in U_{ij}}(r_{uj}-\bar{r}_u)^2}}$	It modifies the traditional PCC measure for computing similarities between items.

Table 3. The similarity measures based on different contextual information

Ref.	Metric Name	Methodology in Brief	Evaluation Goal	Drawback(s)
[29]	JMSD	The metric is designed by combining two factors: • mean squared difference (MSD). • Jaccard index(J)	High level of accuracy, precision, and recall.	Poor Coverage
[32]	SING-metric (Singularity Based)	For each commonly rated item, singularity concerned with the relevant and non-relevant vote is calculated using all the users in the database. Then, a traditional similarity (such as MSD) measure is modulated by the value of the singularity over three subsets of the commonly rated items. Here, the authors utilize MSD as a traditional numerical similarity measure.	Improving the coverage.	Poor Performance due to calculation/updating of the singularity values.
[30]	CJMSD	It augments the JMSD measure by adding a term corresponding to the coverage that a user u can provide to another user v.	Combine the positive aspects of JMSD and SING.	Asymmetric measure
[31]	S-measures (Significance Based)	Firstly, three different types of significances (for items, users and user-item pair) are defined Secondly, s-measures between users u and v are defined over the items for which a significance measure can be defined for users u and v. Finally, traditional similarity measures PCC, COS, and MSD are used to define s-measures PCC_s , COS_s, and MSD_s respectively.	Extends the recommendation quality measures along with new similarity measure.	Poor Performance due to calculation/updating of the significance values.

Table 4. The heuristic-based similarity measures, PIP, NHSM, and PSD

Ref.	Metric Name	Methodology in Brief	Evaluation Goal	Major Drawback(s)
[27]	PIP	It considers the three factors of similarity, *Proximity, Impact, and Popularity* for defining the overall similarity between two users u and v.	New user cold-start problem.	PIP score penalizes repeatedly.
[28]	NHSM	It considers three factors of similarity, Pr*oximity, Significance, and Singularity* and adopts a non-linear function for their computations.	Utilizing local and global rating patterns.	The definitive formula is very complex
[43]	PSD	This measure also considers the three factors of similarity, *Proximity (Px), Significance (Sf), and Distinction (Dt)* and uses the linear combination method for defining the final similarity measure.	optimal weight combination using PSO algorithm.	High-performance time.

This article proposes a new similarity measure for defining the similarities between users or items in neighborhood-based collaborative recommender systems. Firstly, we have described a way for applying the simple matching coefficient (SMC)[38] and explained how it can be applied to the common ratings between users or items. Secondly, the structural information between rating vectors is modeled using the Jaccard index [34]. Finally, the adapted version of the SMC is combined with the structural information in order to define the similarity between users or items.

We have conducted several experiments to validate the effectiveness of the proposed method against several baseline approaches using benchmark datasets, MovieLens-1M, MovieLens-10M, and MovieLens-20M. Empirical results are analyzed using various standard accuracy metrics such as MAE, RMSE, precision, and recall. Finally, it is concluded that the proposed similarity measure provides better predictive accuracy (in terms of MAE and RMSE) and better classification-based accuracy (in terms of precision) against the other baseline approaches.

II. Preliminaries and Related Works

A. Preliminaries

Table 1 summarizes the notations used throughout this article. Essentially, the recommendation problem can be formulated in two common ways [3]: finding the *best item* or providing a list *of top-N items* for a user. In the first formulation, for an active user, we find the best item by estimating the preference value using the training data. Whereas, in the second formulation, a list of top-N items is determined for an active user for making recommendations rather than predicting ratings of items for the user.

As finding similar users or items is the crucial component of the neighborhood-based collaborative recommender systems, therefore, there exist numerous ways for defining the similarities between users/items in the RS literature. Usually, the process of neighborhood-based recommendations consists of the following steps [39]:

Step I Calculate the similarity of all users with an active user u.

Step II Select a subset of users most similar to the active user u (k nearest neighbors)

Step III Find the estimated rating, r_{ui} , by using the ratings of the k nearest neighbors.

Research [24] has examined the various components of the neighborhood-based recommender systems along with their different variations in order to formalize a framework for performing collaborative recommendations.

B. Related Works

This section summarizes some of the related work for measuring similarities in the neighborhood-based collaborative recommendations. All the traditional similarity measures, their definitive formulae along with their brief methodology are summarized in Table 2. These traditional similarity measures suffer from the problem of a few co-rated items due to data sparsity. Additionally, these traditional similarity measures provide inconsistent results as explained in the next section 3.1.2. In order to remove the limitations of existing similarity measures, Bobadilla et al. [29, 30, 31, 32] proposed several similarity measures for better recommendations by exploiting different contextual information. These similarity measures have been summarized in Table 3. Additionally, Bobadilla et al. applied soft computing-based techniques for learning similarities such as genetic algorithms [41] and neural networks [42].

Ahn [27], proposed a heuristic similarity measure, called Proximity-Impact-Popularity(PIP), to resolve the new user cold-start problem by diagnosing the limitations of the existing similarity measures. This PIP measure is further analyzed by H. Liu et al. to improve the performance by utilizing the global information and adopting a non-linear function for computations, this improved version of PIP is called as a New Heuristic Similarity Model (NHSM) [28]. Research [43] further analyzes both PIP and NHSM measures and proposes a new heuristic-based similarity measure, called Proximity-Significance-Distinction (PSD) similarity. Table 4 summarizes these three heuristic-based similarity measures.

III. Proposed Method

A. *Simple Matching Coefficient (SMC)*

The Simple Matching Coefficient (SMC) [38], is a statistical measure for correlating the similarity between binary data samples. It is defined as the ratio of the total number of matching attributes to the total number of attributes present.

$$\text{SMC} = \frac{\text{number of matching attributes}}{\text{total number of attributes}} \tag{1}$$

Consider two objects, A and B, which are described using n binary attributes as shown in Table 5 (here n = 8) also these objects may be equivalently represented using a 2×2 contingency table, as shown in Table 6.

Table 5. Two sample objects described using binary attributes

Object	A1	A2	A3	A4	A5	A6	A7	A8
A	1	1	0	1	1	0	0	0
B	1	0	1	0	1	0	0	0

Table 6. The contingency table

		Object A	
		0	1
Object B	0	**N_{00}**	**N_{01}**
	1	**N_{10}**	**N_{11}**

where,

N_{00} = total number of attributes where both objects A & B are 0.

N_{01} = total number of attributes where A is 0 and B is 1.

N_{10} = total number of attributes where A is 1 and B is 0.

N_{11} = total number of attributes where both objects A & B are 1.

Therefore, (1) can be rewritten using the language of the contingency table as (2) and the similarity between the objects, A and B, using the SMC will be 0.625 (As, $N_{11} = 2$; $N_{01} = 1$; $N_{10} = 2$ & $N_{00} = 3$).

$$\text{SMC} = \frac{N_{00} + N_{11}}{N_{00} + N_{01} + N_{10} + N_{11}} \tag{2}$$

The value of the SMC always lies between 0 and 1 i.e. 0 ≤ SMC(A, B) ≤ 1, the larger value exhibits high similarities between data samples. Its implementation and interpretation are quite easy and obvious but it can be measured only for the vectors of binary attributes with equally significant states i.e. symmetric binary attributes thus called *symmetric binary similarity*. It is noted that the SMC indicates global similarity as it considers all the attributes in the application domain. Furthermore, depending upon the application context, the SMC may or may not be a relevant measure of the similarity as illustrated by the following example.

Illustrative Example 1: Let two customers brought few items C1 = { $\mathbf{I}_1$, $\mathbf{I}_4$, $\mathbf{I}_7$, $\mathbf{I}_{10}$ } and C2 = { $\mathbf{I}_2$, $\mathbf{I}_3$, $\mathbf{I}_4$, $\mathbf{I}_7$, $\mathbf{I}_{12}$ } from a store having 1000 items in total. The similarity between C1 & C2 using the SMC will be (993+2)/1000= 0.995 (As, $N_{11} = 2$; $N_{01} = 3$; $N_{10} = 2$ & $N_{00} = 993$). The SMC returns a very high value of similarity whereas the baskets have very little affinity. It is because of the fact that the two customers have brought a very small fraction of all the available items in the store. Therefore, the SMC is not the relevant measure for similarity in the Market-Basket analysis.

B. *Proposed similarity measure*

In a recommender system (RS), typically the rating data are stored in a user-item (UI) matrix. This rating data can be specified in a variety of ways such as continuous ratings [44], interval-based ratings [45], ordinal ratings [46], binary ratings, etc. depending on the application in hand. Furthermore, the rating data may also be derived implicitly from user's activities (such as buying or browsing an item) in the form of **unary ratings** (a special case of ratings) rather than explicitly specified using rating values. It is noteworthy that the recommendation process is significantly affected by the system used for recording the rating data.

Table 7. A portion of a UI-matrix

	i_1	i2	i3	i4	i5	i6	i7	i8	i9	i10	i11	..	..		i_{15}
U_1															
U_2	3		5	4			3	3	2	5	3				
U_3			1	5		4	5	4	1	2	3				
U_4															

Table 7 shows a portion of a UI-matrix as a quantitative toy-example of explicit rating data for an RS. This toy-example consists of 4 users and 15 items and the users have rated items on a 5-point scale. In this case, one can't directly utilize the SMC for calculating the similarity between users (or items) as the rating vectors are not binary vectors. The rating data can be transformed into a binary form in various possible ways. In the simplest way, one may consider any rating value as presence (i.e. 1s) and unknown value as absence (i.e. 0s) for converting the rating data as binary data. Now, again if we try to calculate the similarity between users (or items) using the SMC then it would also be inappropriate because of the fact that it is irrelevant to consider the number of matches corresponding to the unknown rating value(s). Here, the similarity between U_2 and U_3 using the SMC will be (7+6)/15= 0.866 (As, $N_{11} = 7$; $N_{01} = 1$; $N_{10} = 1$ & $N_{00} = 6$).

This section proposes a way to simulate the SMC so that it can be applied to calculate the similarity between users (or items analogously) in the recommender systems scenario. The proposed way operates on commonly rated items between two users for defining the similarity between these users. Therefore, (1) is applied within the constraint of the commonly rated items. Firstly, we find the number of items for which both users have provided matched(similar) ratings by comparing the actual rating values. Secondly, the ratio of the number of items with matched ratings with respect to the total number of commonly rated items is calculated. Since this simulation of the SMC is bounded within the limit of the commonly rated items, thus it may be called as the SMC for Commonly rated items (SMCC) as defined by (3).

$$\text{SMC} = \frac{\text{number of items with matched ratings}}{\text{total number of commonly rated items}} = \frac{I'_{uv}}{I_{uv}} \qquad (3)$$

where, I'_{uv} represents only those commonly rated items for which users have shown similar ratings i.e. the set of items with matched ratings among all commonly rated items. The items with matched ratings (i.e. the numerator part of the (3)) may further be categorized as positively matched items, negatively matched items and neutrally matched items based on the following criteria as defined by (4) and explained in Example 2.

$$\text{if rating values for an item are} \begin{cases} > \text{median value:} & \text{positively matched item} \\ < \text{median value:} & \text{negatively matched item} \\ = \text{median value(or } \pm 1\text{):neutrally matched item} \end{cases} \qquad (4)$$

Similar criteria(s) may also be defined for other types of rating systems such as continuous or interval-based rating values. Further, (4) can also be written easily for rating systems of the different lengths such as 7-point, 10-point rating scale. To best of our knowledge, no author utilizes the SMC for measuring similarity in neighborhood-based collaborative recommendations.

Semantically, the proposed simulation of the SMC i.e. SMCC simply represents the proportion of the commonly rated items for which both the users have shown the agreements in their rating values (positive/negative/neutral agreement).

Furthermore, in order to utilize the structural information between the two users, we have exploited the Jaccard index (JI). Jaccard index considers only the non-numerical information of the ratings i.e. ignores the absolute values of ratings between the users. It is the ratio of commonly rated items to all the items rated by either of the two users, as shown in (5).

$$J(u,v) = \frac{|I_u \cap I_v|}{|I_u \cup I_v|} \qquad (5)$$

Finally, for defining the similarity between two users, (3) and (5) are combined as follows:

$$\text{Sim}_{JSMCC}(u,v) = \text{Jaccard}(u,v) \times \text{SMCC}(u,v) \qquad (6)$$

The first term of the above equation exploits the structural similarity between the users by considering non-numerical information, while the second term exploits the absolute rating values of each commonly rated item between the users. Since the proposed similarity measure combines the Jaccard index with the proposed simulation of the SMC, therefore, may be called JSMCC.

Illustrative Example 2: Again, consider the toy-example shown in Table 3. Here, the users have rated the items on a scale from 1 to 5 (so median value is 3). For calculating the similarity between users, U2 and U3, firstly we calculate the term SMCC (U2, U3) which is equal to 4/7 = 0.571.

As, the commonly rated items are {i3, i4, i7, i8, i9, i10, i11} and among these items:

positively matched items = {i4} (because the item i4 received the rating values which are greater than the median value from both users);

negatively matched items = {i9} (because the item i9 received the rating values which are less than the median value from both users);

neutrally matched items = {i8, i11} (because these items received the rating values which are equal to the median value (or ±1) from both users).

Therefore, items with matched ratings will be {i4} ∪ {i9} ∪ {i8, i11} i.e. {i4, i8, i9, i11} while the items with non-matched ratings will be {i3, i7, i10}.

Secondly, we calculate the Jaccard similarity i.e. Jaccard(U2, U3) which comes out to be 7/9 = 0.778. Finally, the similarity between these users, using the proposed similarity i.e. JSMCC, will be 0.571 × 0.778 = 0.44.

C. Rationale

The Pearson Correlation Coefficient (PCC) and the Cosine similarity measures (COS) are the most widely used and accepted similarity measures in neighborhood-based collaborative recommendations according to the general bibliography in the field. The Pearson Correlation Coefficient measures the strength of the linear relationship between two rating vectors whereas COS measures the cosine of the angle between rating vectors. Albeit, these two commonly used similarity measures have been shown to be advantageous/fruitful in many collaborative recommender systems but they are inadequate in some situations as explained below:

1) The PCC is invariant to both scaling and location. As an illustration, the value of the PCC between the rating vectors V1:(1,2,1,2,1) and V2:(2,4,2,4,2), V2 = 2×V1 i.e. PCC (V1, V2) = 1, however both rating vectors are not similar. Further consider a rating vector V3: (3,5,3,5,3), V3 = 2×V1 +1, so the PCC (V1, V3) = 1, which are also not similar.

2) The COS similarity measure is invariant to scaling i.e. the value of the COS between the rating vectors V1:(1,2,1,2,1) and V2:(2,4,2,4,2), V2 = 2×V1, COS (V1, V2) = 1, however, both rating vectors are not similar.

3) If the two rating vectors are flat i.e. V1:(1,1,1,1,1) and V2:(5,5,5,5,5) then the PCC cannot be calculated (as the denominator part is zero) while the COS results in 1 regardless of the differences between the rating vectors.

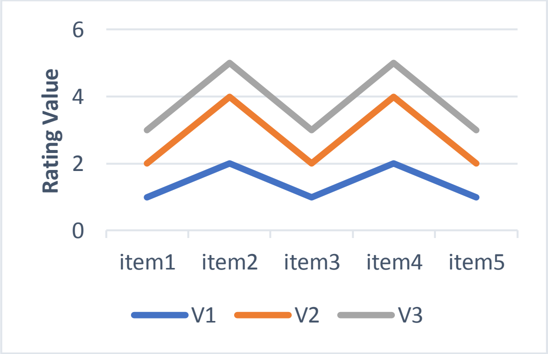
4) If the length of rating vectors is 1 i.e. the number of commonly rated items/users is 1, then, the PCC cannot be calculated and the COS results in 1 irrespective to the rating values.

5) Both PCC and COS are equivalent when the mean is zero.

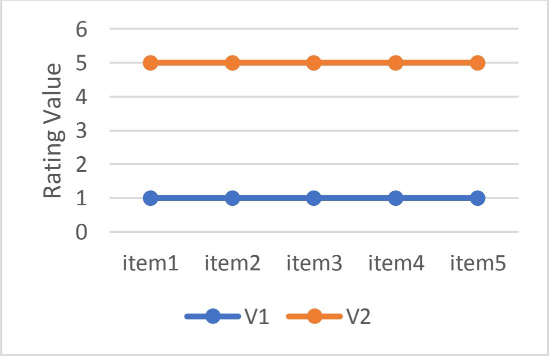
The chances of occurrence of the above-mentioned problems are very high when sufficient ratings are not available for similarity calculation i.e. dataset is sparse.

Illustrative Example 3: Let us consider the rating vectors as shown in Fig. 2(a), which shows three rating vectors, V1, V2, and V3, corresponding to users U1, U2, and U3 respectively. Here, also note that V2 =2×V1 and V3 = 2×V1 +1; the traditional similarity measures such as PCC and COS result in inadequate values as explained earlier, while the proposed SMCC(U1, U2) = 3/5 =0.6 and Jaccard(U1,U2) = 5/8 = 0.625 (assuming | Iu ∪ Iv |= 8), therefore, JSMCC(U1, U2) will be 0.6 × 0.625 = 0.375.

Fig. 2(b) shows two flat rating vectors V1 and V2 = 5 × V1; For such vectors, PCC is not defined while the COS always results in 1 regardless of the differences between the rating vectors. Here, in this case, the proposed similarity measure provides zero similarity which is consistent with their conflicting nature.



(a)



(b)

Fig.2. Illustrative rating vectors showing deceptive values of the PCC & COS (a) Scaling and shifting of rating vectors (b) Flat rating vectors.

While calculating the similarity between two users, the proposed similarity measure uses all the ratings given by two users in contrast to traditional measures that operate only on the co-rated items. Additionally, for commonly rated items, the proposed measure just compares the values of the rating pair to determine the interests of the two users for a particular item, which is quite an intuitive indicator of the similarities between the users. Finally, the empirical results on the different datasets (MovieLens-1M, 10M, and 20M) demonstrate the superiority of the proposed similarity measure against the traditional as well as state-of-art similarity measures in term of predictive accuracy, as explained in the experiment section.

IV. Experiments

There has been an ample amount of research in the area of recommender systems in nearly the last twenty-five years[23]. Therefore, an RS designer has a considerable range of algorithms available for developing his/her system and must find the most suitable approach based on his goals along with a few given constraints. Similarly, a researcher who proposes a new recommendation algorithm must evaluate the suggested approach against the various existing approaches by using some evaluation metric.

In both cases, academia and industry, one has to analyze the relative performance of the candidate algorithms by conducting some experiments. There are three primary types of experiments for evaluating recommender systems; offline experiments, user studies, and online experiments[47]. The later two types of experiments involve users whereas offline experiments require no communication with real users.

Offline experiments are typically the easiest to conduct and hence the most popular/accepted way of testing various recommendation approach[3]. An offline experiment is conducted with historical data, such as ratings, this pre-collected dataset must be chosen carefully to avoid any biases in handling users, items and ratings. Here, we have performed offline experiments on the standardized benchmark rating data sets from the

MovieLens website which is collected and made publicly available by the GroupLens research lab[48]. Table 8 summarizes these datasets briefly and more detailed description of the MovieLens datasets can be found in the article [49].

A. Experimental Design

We have employed the traditional user-based collaborative filtering (UBCF) algorithm with k-Nearest Neighbors(kNN) technique for validating the effectiveness of the proposed method against the baseline approaches. However, the proposed similarity measure takes the place of its counter component in the traditional UBCF as defined according to (6). Table 9 summarizes the details of the software and systems used for conducting all the experiments. The high-performance computing system is required to process the latest MovieLens-20M dataset because of its huge size which requires high memory requirement with more processing power. Furthermore, we have used the recently introduced framework, Collaborative Filtering 4 Java (CF4J)[50], for performing experiments. CF4J library has been designed to carry out collaborative filtering-based recommendation research experiments. It provides a parallelization framework specifically designed for CF and supports extensibility of the main functionalities. Simply, it has been developed to satisfy the needs of the researchers by providing access to any intermediate value generated during the recommendation process i.e. one can say that it is a “library developed from researchers to researchers”.

Table 8. The brief description of the datasets used

Dataset	Release Date	Brief Detail	Sparsity Level
MovieLens-1M	2/2003	• 1000,209 ratings from 6040 users on 3900 movies • Each user has rated at least 20 movies.	$1-\frac{1000,209}{6040*3900}=95.754$
MovieLens-10M	1/2009	• 10,000,054 ratings from 71567 users on 10681 movies • Each user has rated at least 20 movies.	$1-\frac{10000054}{71567*10681}=98.692$
MovieLens-20M	4/2015	• 20,000,263 ratings from 7138493 users on 27278 movies • Each user has rated at least 20 movies.	$1-\frac{20000263}{138493*27278}=99.470$

B. Evaluation metrics

Based on the general bibliography in the RSs research, accuracy is the most explored property for designing an RS. The fundamental rationale is that if a recommender system suggests more accurate recommendations then it will be preferred by users. Therefore, accuracy is considered as the most important goal for evaluating various candidate recommendation approaches[51]. Additionally, accuracy measurement is independent of the GUI provided by a recommender system. The offline experiments are well suited for simulating the accuracy measurement using the standardized benchmark datasets such as MovieLens. Here, we have evaluated the following two types of accuracy measures:

- Accuracy of predictions
- Accuracy of classifications

a. Measuring the accuracy of rating predictions

Let it is known that a user u has rated an item i with the rating r_{ui} and a recommendation algorithm predicts this rating value as $\widehat{r_{ui}}$. Then, one can say that estimation error for this rating will be $e_{ui} = \widehat{r_{ui}} - r_{ui}$ i.e. the diffrence between the predicted score and the actual score. In order to compute the overall error of a system, for a given test set T of user-item pairs, Mean Absolute Error(MAE) and Root Mean Squared Error(RMSE) are the most popular metrics used in the literature as shown below by (7) and (8) respectively. MAE computes the average error while the RMSE sums up the squared errors, therefore, RMSE penalizes large error values.

$$MAE = \frac{1}{|T|}\sum_{(u,i)\in T} |(\hat{r}_{ui} - r_{ui})| \tag{7}$$

$$RMSE = \sqrt{\frac{1}{|T|}\sum_{(u,i)\in T} (\hat{r}_{ui} - r_{ui})^2} \tag{8}$$

b. Measuring the accuracy of classifications

There are situations where it is not necessary to predict the actual rating values of items in order to provide recommendations to users but rather a recommender system tries to provide a list of items (topN items) that may be relevant or useful for users. As an illustration, when a user add a movie to the queue, Netflix[46] recommends a set of movies with respect to the added movie, here, the user is not concerned with the actual rating values of these suggested set of movies but rather the system correctly predicts that the user will add these movies to the queue.

Table 9. The details of the software and systems used.

High Performance Computing System	Intel(R) Xeon(R) Gold 6132(Skylake) 14 cores CPU@ 2.6GHz with **96GB** RAM
Eclipse Java EE IDE	4.7.2 or higher
Java version (JRE)	Java 1.8 or higher
CF4J library version (framework)	1.1.1 or higher

In the recommender system scenario, the aim of a classification chore is to recognize the N most useful items for an active user. Therefore, we can use classic

Information Retrieval (IR) metrics to evaluate recommender systems. Precision and Recall are the most examined IR metrics, these metrics can be adapted to RSs without any difficulty in the following manner.

Precision: the proportion of top recommendations that are also relevant recommendations.

Recall: the proportion of all relevant recommendations that appear in top recommendations.

Formally, Let R(u) is the total number of recommended items to the user u, T(u) is the theoretical maximum possible number of relevant recommendations for that user u (which comes out to be the testing set actually), and β is the rating threshold to measure the relevancy. So the precision and recall may be defined using (9) and (10) respectively. Similar to the IR jargon, these metrics can also be defined using a 2 × 2 contingency table as shown in Table 10.

Table 10. Classification of all possible outcomes of a recommendation

	Recommended	**Not Recommended**
Preferred	True-Positive (tp)	False-Negative (fn)
Not preferred	False-Positive (fp)	True-Negative (tn)

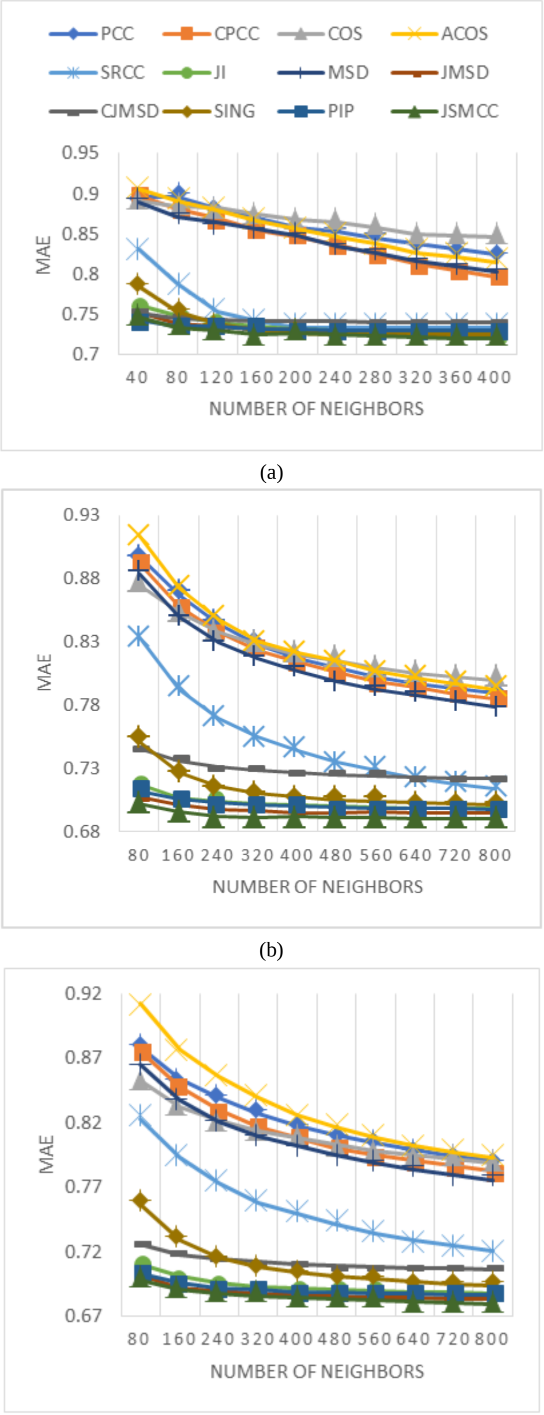
V. Results and Discussions

The effectiveness of the proposed measure has been evaluated against the various baseline similarity measures, listed in Table 11 and explained earlier in the section of related work.

Table 11. The various baseline approaches compared with the proposed method

S.No.	Similarity Measure	Ref.
1	Pearson Correlation Coefficient (PCC)	[25]
2	Constrained Pearson Correlation Coefficient (CPCC)	[8]
3	Cosine Similarity (COS)	[14]
4	Adjusted Cosine Similarity (ACOS)	[40]
5	SRCC	
6	Jaccard Index	[34]
7	Mean Squared Difference (MSD)	[8]
8	Jaccard Mean Squared Difference (JMSD)	[29]
9	Coverage-Jaccard-Mean Squared Difference (CJMSD)	[30]
10	Singularity-based similarity (SING)	[32]
11	Proximity-Impact-Popularity (PIP measure)	[27]

In order to determine the predictive accuracy, in terms of MAE and RMSE, we have used 80% of the rating data as training data and the remaining 20 % is used for testing purpose. Further, for each test user, 20% of his ratings are used for validation. Training and test data are chosen randomly in order to avoid any bias from the data selection. The values of MAE and RMSE are calculated for the proposed method along with various baseline approaches by varying the number of neighbors i.e. the neighborhood-size. MAE and RMSE values are compared for the proposed method against various baseline approaches in Fig. 3 and Fig. 4 respectively for (a) MovieLens-1M dataset (b) MovieLens-10M dataset, and (c) MovieLens-20M dataset.



(a)

(b)

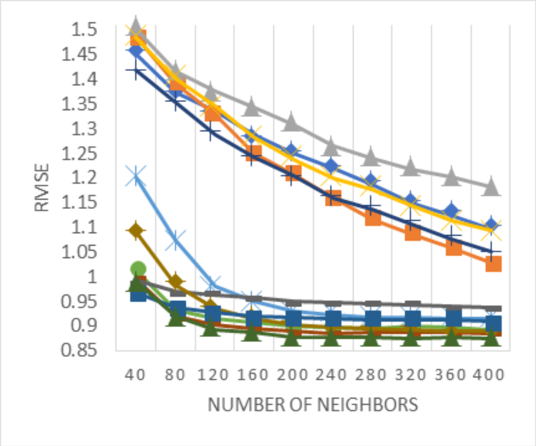
(c)

Fig.3. The MAE values for (a) MovieLens-1M (b) MovieLens-10M (c) MovieLens-20M datasets.

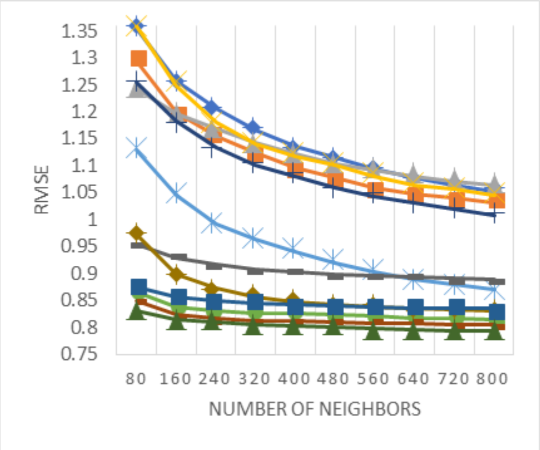
As shown in Fig. 3(a)-(c), the empirically obtained values of MAE demonstrate that the proposed similarity measure results in lowers the MAE values against baseline approaches that indicates the better performance in terms of predictive accuracy. Likewise, the values of

RMSE, as shown in Fig. 4(a)-(c), represent the superiority of the proposed method against the various baseline measures. Based on the application at hand, either MAE or RMSE may be selected for evaluating the predictive accuracy. There is no decisive answer to this debate, research [52] compares and contrasts the relative advantages of MAE and RMSE metrics. Here, the proposed measure provides better results against the various baseline measures for both metrics, MAE and RMSE.

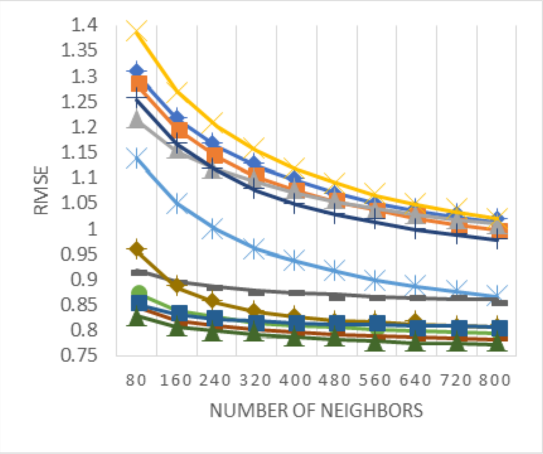
Furthermore, these predictive accuracy metrics such as MAE and RMSE are not enough for reflecting the true users' experience, therefore, additional classification based metrics such as precision and recall are also used to evaluate the effectiveness of an algorithm [53]. We have also calculated the values of precision against the varying size of the recommended list i.e. top-N recommendations. Also, while evaluating precision values, the neighborhood size for each test user is kept to be fixed according to the datasets. Neighborhood-size is chosen in such a way that after the chosen size there is no significant improvement in the MAE/RMSE values for each dataset. Table 12 summarizes the different parameter values used for measuring precision values. Fig. 5 compares the precision values for different size of the recommended list for all three datasets (a) MovieLens-1M (b) MovieLens-10M (c) MovieLens-20M.



(a)



(b)



(c)

Fig.4. The RMSE values for (a) MovieLens-1M (b) MovieLens-10M (c) MovieLens-20M datasets.

Table 12. The parameter involved in experimenting with the classification-based accuracy

	ML-1M	ML-10M	ML-20M
Test-Users %	20%		
Test-Items %	20%		
Rating Threshold (β)	4		
Recommendation size	{10,15,20,25}		
Neighborhood-Size	200	320	400

As shown in Fig. 5 (a)-(c), the empirically obtained values of precision demonstrate that the proposed similarity measure provides better results for precision than the various baseline approaches, which shows the better performance in terms of classification accuracy. Based on the empirical results, we believe that the proposed method provides better results due to the following facts:

- It takes into account all the items rated by users, in contrast to the traditional measures which consider only commonly rated items (as the structural information between users are utilized using the Jaccard index)
- The actual rating values of all the commonly rated items are exploited to determine the proportion of items for which two users have shown similar rating pattern rather than finding any types of correlations (such as in PCC, CPCC) or angles (such as in COS) or other complex relation (such as in PIP, SING) between the rating vectors. This formulation is very simple and intuitive in nature.

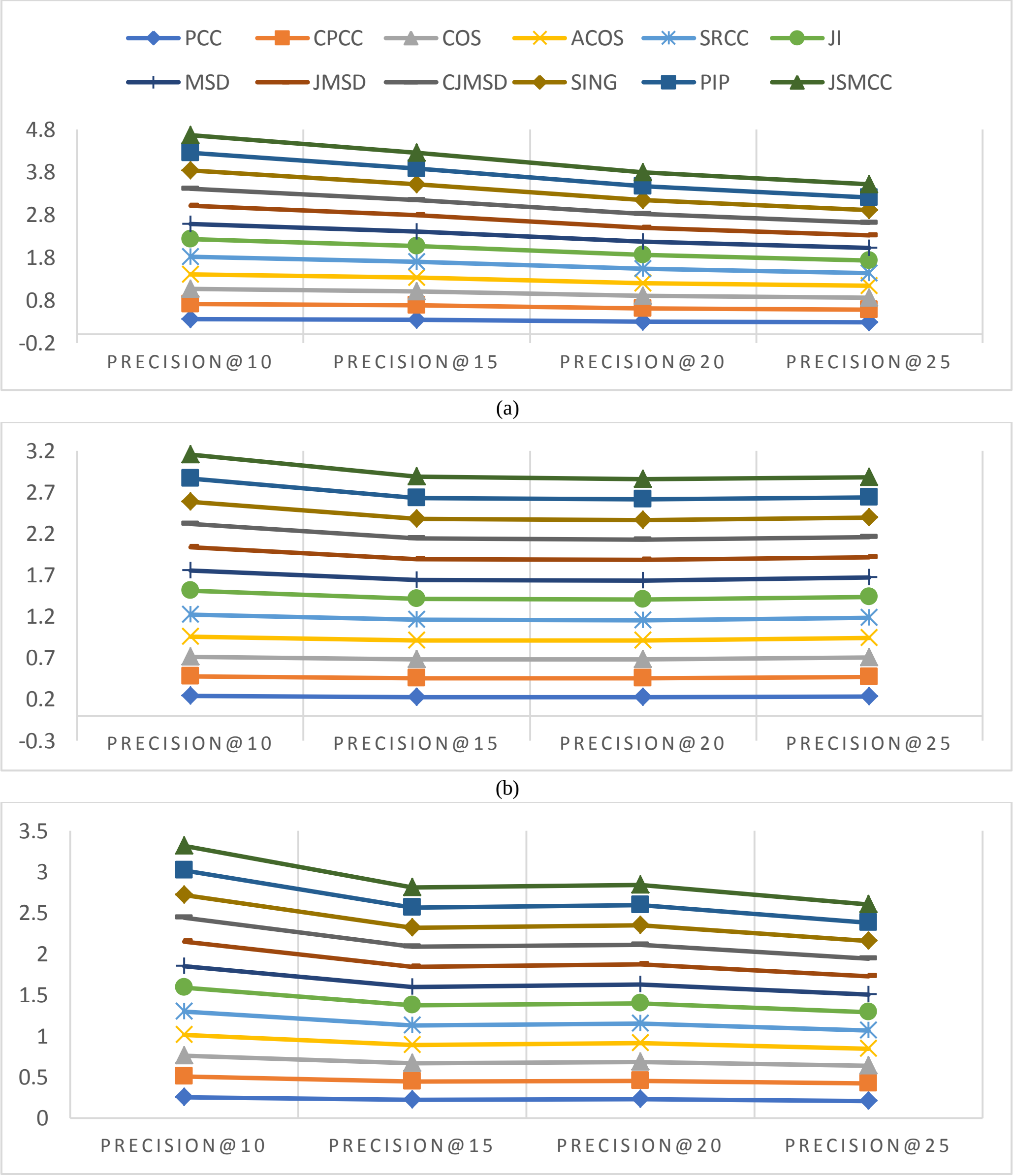
$$\text{Precision}_u = \frac{|\{i \in R(u) : r_{ui} > \beta\}|}{|R(u)|} = \frac{\#tp}{\#tp + \#fp} \qquad (9)$$

$$\text{Re}\,call_u = \frac{|\{i \in R(u) : r_{ui} > \beta\}|}{|T(u)|} = \frac{\#tp}{\#tp + \#fn} \qquad (10)$$

VI. Conclusions

This article proposes a new similarity measure for neighborhood-based collaborative recommender systems. Finding similar users or items is an important step in collaborative filtering-based recommendation approaches. The proposed similarity measure is based on an adapted version of the simple matching coefficient together with the Jaccard index. Firstly, we have described a way to simulate the simple matching coefficient so that it can be applied to calculate the similarity between users or items, also, the structural information between rating vectors are also exploited using the Jaccard index.

Finally, several experiments have been performed for validating the effectiveness of the proposed measure against baseline methods. The empirical results demonstrate that the proposed method provides better predictive accuracy (in terms of MAE and RMSE metrics) and better classification accuracy (in terms of precision metric) than the baseline approaches. We believe that web-portals or e-commerce websites may utilize the proposed similarity measure into their existing recommender systems for improved accuracy because they have to just upgrade only the similarity measurement module. The effect of the proposed similarity measure on other RS goals such as diversity, coverage, etc. may be analyzed as the future task.



(a)

(b)

(c)

Fig.5. The relative values of precision for varying size of the recommended list (a) MovieLens-1M (b) MovieLens-10M (c) MovieLens-20M datasets (Here, Precision values are shown on a scale while real values are between 0 and 1).

REFERENCES

[1] P. Resnick and H. R. Varian, Recommender systems, vol. 40, no. 3. 1997.

[2] F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, Recommender Systems Handbook, 1st ed. Berlin, Heidelberg: Springer-Verlag, 2010.

[3] C. C. Aggarwal, Recommender Systems: The Textbook, 1st ed. Springer Publishing Company, Incorporated, 2016.

[4] M. Balabanović and Y. Shoham, "Fab: Content-based, Collaborative Recommendation," Commun. ACM, vol. 40, no. 3, pp. 66–72, Mar. 1997.

[5] K. Lang, "NewsWeeder : Learning to Filter Netnews (To appear in ML 95)," Proc. 12th Int. Mach. Learn. Conf., 1995.

[6] C. Science and J. Wnek, "Learning and Revising User Profiles: The Identification of Interesting Web Sites," Mach. Learn., vol. 331, pp. 313–331, 1997.

[7] W. Hill, L. Stead, M. Rosenstein, and G. Furnas, "Recommending and evaluating choices in a virtual community of use," in Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '95, 1995.

[8] U. Shardanand and P. Maes, "Social information filtering: Algorithms for Automating 'Word of Mouth,'" Proc. SIGCHI Conf. Hum. factors Comput. Syst. - CHI '95, pp. 210–217, 1995.

[9] Billsus Daniel and Pazzani Michael J., "User modeling for adaptative news access. ," User Model. User-adapt. Interact., vol. 10, pp. 147–180, 2002.

[10] R. Burke, "Hybrid recommender systems: Survey and experiments," User Model. User-Adapted Interact., 2002.

[11] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative Filtering beyond the User-Item Matrix : A Survey of the State of the Art and Future Challenges," ACM Comput. Surv., vol. 47, no. 1, pp. 1–45, 2014.

[12] X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," Adv. Artif. Intell., vol. 2009, no. Section 3, pp. 1–19, 2009.

[13] M. D. Ekstrand, "Collaborative Filtering Recommender Systems," Found. Trends® Human–Computer Interact., vol. 4, no. 2, pp. 81–173, 2011.

[14] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," Proc. 14th Conf. Uncertain. Artif. Intell., vol. 461, no. 8, pp. 43–52, 1998.

[15] D. Joaquin and I. Naohiro, "Memory-Based Weighted-Majority Prediction for Recommender Systems," Res. Dev. Inf. Retr., 1999.

[16] A. Nakamura and N. Abe, "Collaborative Filtering Using Weighted Majority Prediction Algorithms," in Proceedings of the Fifteenth International Conference on Machine Learning, 1998, pp. 395–403.

[17] D. Billsus and M. J. Pazzani, "Learning collaborative information filters," Proc. Fifteenth Int. Conf. Mach. Learn., vol. 54, p. 48, 1998.

[18] T. Hofmann, "Collaborative filtering via gaussian probabilistic latent semantic analysis," Proc. 26th Annu. Int. ACM SIGIR Conf. Res. Dev. information Retr. - SIGIR '03, p. 259, 2003.

[19] L. Getoor and M. Sahami, "Using probabilistic relational models for collaborative filtering," Work. Web Usage Anal. User Profiling, 1999.

[20] B. Marlin, "Modeling User Rating Profiles for Collaborative Filtering," in Proceedings of the 16th International Conference on Neural Information Processing Systems, 2003, pp. 627–634.

[21] D. Pavlov and D. Pennock, "A maximum entropy approach to collaborative filtering in dynamic, sparse, high-dimensional domains," Proc. Neural Inf. Process. Syst., pp. 1441–1448, 2002.

[22] K. Laghmari, C. Marsala, and M. Ramdani, "An adapted incremental graded multi-label classification model for recommendation systems," Prog. Artif. Intell., vol. 7, no. 1, pp. 15–29, 2018.

[23] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry, "Using collaborative filtering to weave an information tapestry," Commun. ACM, vol. 35, no. 12, pp. 61–70, 1992.

[24] J. O. N. Herlocker and J. Riedl, "An Empirical Analysis of Design Choices in Neighborhood-Based Collaborative Filtering Algorithms," Inf. Retr. Boston., pp. 287–310, 2002.

[25] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens : An Open Architecture for Collaborative Filtering of Netnews," in Proceedings of the 1994 ACM conference on Computer supported cooperative work, 1994, pp. 175–186.

[26] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl, "GroupLens: applying collaborative filtering to Usenet news," Commun. ACM, vol. 40, no. 3, pp. 77–87, 1997.

[27] H. J. Ahn, "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem," Inf. Sci. (Ny)., vol. 178, no. 1, pp. 37–51, 2008.

[28] H. Liu, Z. Hu, A. Mian, H. Tian, and X. Zhu, "A new user similarity model to improve the accuracy of collaborative filtering," Knowledge-Based Syst., vol. 56, pp. 156–166, 2014.

[29] J. Bobadilla, F. Serradilla, and J. Bernal, "A new collaborative filtering metric that improves the behavior of recommender systems," Knowledge-Based Syst., vol. 23, no. 6, pp. 520–528, 2010.

[30] J. Bobadilla, F. Ortega, A. Hernando, and Á. Arroyo, "A balanced memory-based collaborative filtering similarity measure," Int. J. Intell. Syst., vol. 27, no. 10, pp. 939–946, Oct. 2012.

[31] J. Bobadilla, A. Hernando, F. Ortega, and A. Gutiérrez, "Collaborative filtering based on significances," Inf. Sci. (Ny)., vol. 185, no. 1, pp. 1–17, 2012.

[32] J. Bobadilla, F. Ortega, and A. Hernando, "A collaborative filtering similarity measure based on singularities," Inf. Process. Manag., vol. 48, no. 2, pp. 204–217, 2012.

[33] B. K. Patra, R. Launonen, V. Ollikainen, and S. Nandi, "A new similarity measure using Bhattacharyya coefficient for collaborative filtering in sparse data," Knowledge-Based Syst., vol. 82, pp. 163–177, 2015.

[34] P. Jaccard, "Distribution comparée de la flore alpine dans quelques régions des Alpes occidentales et orientales," Bull. la Socit Vaudoise des Sci. Nat., vol. 37, pp. 241–272, 1901.

[35] S. Bag, S. K. Kumar, and M. K. Tiwari, "An efficient recommendation generation using relevant Jaccard similarity," Inf. Sci. (Ny)., vol. 483, pp. 53–64, 2019.

[36] J. Díez, D. Martínez-Rego, A. Alonso-Betanzos, O. Luaces, and A. Bahamonde, "Optimizing novelty and diversity in recommendations," Prog. Artif. Intell., 2018.

[37] N. Good et al., "Combining Collaborative Filtering with Personal Agents for Better Recommendations," in Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence, 1999, pp. 439–446.

[38] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, 3rd ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2011.

[39] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, “An algorithmic framework for performing collaborative filtering,” in Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR ’99, 1999, pp. 230–237.

[40] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, “Item-based collaborative filtering recommendation algorithms,” Proc. tenth Int. Conf. World Wide Web - WWW ’01, pp. 285–295, 2001.

[41] J. Bobadilla, F. Ortega, A. Hernando, and J. Alcalá, “Improving collaborative filtering recommender system results and performance using genetic algorithms,” Knowledge-Based Syst., vol. 24, no. 8, pp. 1310–1316, 2011.

[42] J. Bobadilla, F. Ortega, A. Hernando, and J. Bernal, “A collaborative filtering approach to mitigate the new user cold start problem,” Knowledge-Based Syst., vol. 26, pp. 225–238, 2012.

[43] B. Jiang, T. T. Song, and C. Yang, “A heuristic similarity measure and clustering model to improve the collaborative filtering algorithm,” ICNC-FSKD 2017 - 13th Int. Conf. Nat. Comput. Fuzzy Syst. Knowl. Discov., pp. 1658–1665, 2018.

[44] “Jester Collaborative Filtering Dataset.” [Online]. Available: https://www.ieor.berkeley.edu/~goldberg/jester-data/. [Accessed: 10-Jan-2019].

[45] “Amazon.com: Online Shopping for Electronics, Apparel, Computers, Books, DVDs &amp; more.” [Online]. Available: https://www.amazon.com/. [Accessed: 08-Feb-2019].

[46] “Netflix – Watch TV Programmes Online, Watch Films Online.” [Online]. Available: https://www.netflix.com/. [Accessed: 08-Feb-2019].

[47] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, “Evaluating collaborative filtering recommender systems,” ACM Trans. Inf. Syst., vol. 22, no. 1, pp. 5–53, 2004.

[48] “MovieLens | GroupLens.” [Online]. Available: https://grouplens.org/datasets/movielens/. [Accessed: 22-Dec-2018].

[49] F. M. Harper and J. A. Konstan, “The MovieLens Datasets,” ACM Trans. Interact. Intell. Syst., vol. 5, no. 4, pp. 1–19, 2015.

[50] F. Ortega, B. Zhu, J. Bobadilla, and A. Hernando, “CF4J: Collaborative filtering for Java,” Knowledge-Based Syst., vol. 152, pp. 94–99, 2018.

[51] A. Gunawardana and G. Shani, “A survey of accuracy evaluation metrics of recommendation tasks,” J. Mach. Learn. Res., vol. 10, pp. 2935–2962, 2009.

[52] T. Chai and R. R. Draxler, “Root mean square error (RMSE) or mean absolute error (MAE)? -Arguments against avoiding RMSE in the literature,” Geosci. Model Dev., vol. 7, no. 3, pp. 1247–1250, 2014.

[53] G. Carenini and R. Sharma, “Exploring More Realistic Evaluation Measures for Collaborative Filtering,” in Proceedings of the 19th National Conference on Artifical Intelligence, 2004, pp. 749–754.

Authors’ Profiles

Vijay Verma is currently working as an Assistant Professor at the Computer Engineering Department, National Institute of Technology (NIT), Kurukshetra, India. He holds M.Tech. degree from Indian Institute of Technology (IIT), Roorkee, India. His research interest includes Data Mining, Recommender Systems, Personalization, and related areas.

Dr. Rajesh Kumar Aggarwal holds his Ph.D. degree and M. Tech. degree in Computer Engineering from the NIT, Kurukshetra in 2014 and 2004 respectively. Currently, he is working as a Professor at the Computer Engineering Department, National Institute of Technology (NIT), Kurukshetra, India. His research interest includes Speech Recognition, Information Retrievals, and Personalization Technologies.

How to cite this paper: Vijay Verma, Rajesh Kumar Aggarwal, "A New Similarity Measure Based on Simple Matching Coefficient for Improving the Accuracy of Collaborative Recommendations", International Journal of Information Technology and Computer Science(IJITCS), Vol.11, No.6, pp.37-49, 2019. DOI: 10.5815/ijitcs.2019.06.05