Expert Systems With Applications 185 (2021) 115482

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Review

Survey of similarity functions on neighborhood-based collaborative filtering

Halime Khojamli [a], Jafar Razmara [a, *]

[a] Department of Computer Science, University of Tabriz, Tabriz, Iran

ARTICLE INFO

Keywords:
Recommender systems
Collaborative filtering
Neighborhood-based approach
Similarity functions
Similarity measures

ABSTRACT

Today, recommender systems play a vital role in the acceleration of searches by internet users to find what they are interested in. Among the strategies proposed for recommender systems, collaborative filtering has received due attention regarding its simplicity and efficiency. The key factor for the success of this strategy returns to the similarity calculation methods that affect the accuracy of its recommendations. Regarding the large volume of articles published in the field of collaborative filtering for the development of recommender systems, it is necessary to provide a comprehensive review of the similarity functions and their efficiency presented in the field. Of course, several surveys have already been published to investigate the similarity functions proposed for collaborative filtering, but these articles either have looked briefly at these functions or reviewed a few numbers of them. In this study, the effort was to provide a comprehensive study on the similarity functions proposed for collaborative filtering with a special focus on the rating-based and neighbor-based approaches. After a brief explanation of each similarity function, some popular evaluation metrics were used for their evaluation using the MovieLens datasets. The comparative evaluation results presented in this article provide a highly useful reference for researchers in this field to choose their appropriate similarity function.

1. Introduction

Nowadays, with the explosive growth in the volume of information available via internet resources, people spend a lot of time to search and find their interested items. This has formed the forthcoming challenge of information overload which hampers internet users to access their appropriate items timely. Recommender systems have been proposed to deal with the problem of information overload through filtering unfavorable items and suggesting items according to user's preferences, interest, or observed behavior (Yan & Tang, 2019). These suggestions are created through monitoring or examining the behavior of users to choose items based on their preferences.

Among different strategies developed for recommender systems, collaborative filtering is the class of the most famous, successful (Saeed, 2017; Ahmadian, 2017; Huang, Yu, & Wang, 2018; Wang, Deng, Gao, & Zhang, 2017) and widely used algorithms (Yan & Tang, 2019). This popularity is due to its simplicity and efficiency in the recommendation of items based on the user's interests. The methods assume that the future behavior of each user is more likely to his/her past behavior. Suggestions that this type of recommender systems provide to an active user are based on the previous ratings by this active user and other similar users, and there is no additional information about the items and users (Chen et al., 2018).

The methods within the collaborative filtering approach are generally divided into neighborhood-based and model-based (Kluver, Ekstrand, & Konstan, 2018). This classification was first proposed in 1998 by Breese, Heckerman, and Kadie (1998). Based on definition, the model-based methods mostly attempt to model the system using the matrix factorization while the neighborhood-based methods work as an extension of the K-Nearest Neighbor (KNN) classifiers. These algorithms are based on the fact that similar users have the same rating patterns and similar items receive the same rates (Aggarwal, 2016). Though the studies for developing new algorithms are continuing, these algorithms are still used in several active systems. These basic approaches are generally simple and flexible as well as show a competitive performance. As a result, they have been more popular to be employed in the development of recommender systems (Kluver et al., 2018).

The developed nearest neighbour-based collaborative filtering algorithms work based on two different strategies: rating-oriented and ranking-oriented. The methods based on the rating-oriented strategy

* Corresponding author.
E-mail address: razmara@tabrizu.ac.ir (J. Razmara).

https://doi.org/10.1016/j.eswa.2021.115482
Received 13 June 2020; Received in revised form 21 March 2021; Accepted 24 June 2021
Available online 6 July 2021

(Herlocker et al., 2002; Breese et al., 1998) use the rating information from other similar users for pblueiction of a user's potential ratings on unrated items. These methods calculate the similarity between two users through their rating scores on the commonly rated items. Unlike the rating-oriented strategy, the ranking-oriented methods (McNee, Riedl, & Konstan, 2006) produce a user's preference item list directly without using the rating information. In general, the neighborhood-based collaborative filtering approach has become one of the most popular methods used in recommmender systems due to their remarkable advantages including interpretability, strong robustness, and competitive performance (Hofmann, 2004).

The sparsity challenge, along with other challenges in recommender systems such as cold-start and scalability has led the researchers to explore and propose several strategies for recommender systems based on different similarity functions introducing interesting ideas. The ideas are commonly trying to overcome the problems with recommender systems and increase their performance (Deng et al., 2019). The key point in designing the methods based on the collaborative filtering approach is to calculate the similarity between users (or items) using an efficient function. The similarity function has a direct impact on the performance of both neighborhood-based and model-based methods (Yan & Tang, 2019; Liu et al., 2017; Aghdam, Analoui, & Kabiri, 2015; Aghdam, Analoui, & Kabiri, 2016). The functions are also applied in other fields of recommender systems (Yang, Wei, Wu, Zhang, & and Zhang, 2009; Aghdam, Analoui, & Kabiri, 2016).

Many articles have been published to review the similarity functions designed for recommender systems, but there is no comprehensive study to collect, investigate, and compare these functions. Recent reviews have coveblue a few numbers of these functions, while the number of introduced similarity functions is much more. The current study was organized based on the belief that a comprehensive review of the available similarity functions can assist software developers to design more efficient models for recommender systems regarding their properties and capabilities. On the other hand, the use of these functions in other research fields such as clustering and data mining can be extended by this study. The purpose of this research was to investigate in detail the different similarity functions and their related algorithms based on the collaborative filtering approach in terms of their limitations, efficiency, and scalability.

1.1. Prior Related Surveys

Several articles (Bobadilla, Ortega, Hernando, & Gutiérrez, 2013; Adomavicius & Tuzhilin, 2005; Isinkaye, Folajimi, & Ojokoh, 2015; Reddy & Govindarajulu, 2017) have been published to review the algorithms proposed for recommender systems which were designed based on different data mining, machine learning, or any other techniques. These articles have a brief overview of the available algorithms as well as the comparison and evaluation of their performance. The lack of comprehensive reports may due to a large number of similarity functions and their related algorithms making the review procedure difficult. On the other hand, there are several review articles (Chen et al., 2018; Cacheda, Carneiro, Fernández, & Formoso, 2011; Su & Khoshgoftaar, 2009; Kluver et al., 2018) with a focus on the algorithms based on the collaborative filtering approach, whereas a few traditional similarity functions were investigated.

In general, the performance of existing neighborhood-based algorithms depends on their similarity function, which uses a rating matrix on co-rated items. However, due to the sparsity of the rating matrix, the methods do not obtain high accuracy. Therefore, to model a proper recommender system, evaluation of the similarity functions can be helpful (Stephen, Xie, & and Rai, 2017). Hence, to examine the similarity functions and identifying the appropriate functions, some researches have been reported (Saranya, Sadasivam, & Chandralekha, 2016; Singh, Pramanik, & Choudhury, 2019; Katpara & Vaghela, 2016; Herlocker, Konstan, & Riedl, 2002) using different evaluation methods. Hassanieh, Jaoudeh, Abdo, and Demerjian (2018) examined the behavior of several traditional similarity functions for collaborative filtering at different volume percentages of a dataset. Arsan, Koksal, and Bozkus (2016) believed that users' flavors are changed during the time, but items' properties remain constant, and thus, they used similarity functions on items instead of users' choice and performed comparisons. Spertus, Sahami, and and Buyukkokten (2005) evaluated the similarity functions on the binary rating matrix. Among these reviewed articles, Agarwal and Chauhan (2017) studied the largest number of similarity functions (including 13 functions) but still their set cover a small fraction of all known functions used in recommender systems. Besides, regarding that these review articles have used different evaluation criteria for investigation of different similarity functions, collecting their results in a comprehensive review was not facilitated.

1.2. Our work

In this article, the effort is to collect and investigate the similarity functions presented in the field of neighborhood-based collaborative filtering. As far as we know, this is the first survey trying to collect all the similarity functions in this field as much as possible and provide a comparative evaluation of them. The functions are introduced first and then examined on the MovieLens dataset as a standard benchmark for evaluation of recommender systems using different evaluation metrics. Regarding that the neighborhood-based methods use rating-oriented or ranking-oriented strategies (Shams, 2018), they use different mechanisms in applying similarity functions. It would therefore not be fair to apply and compare their similarity functions in equal criteria. The rating-oriented algorithms pay more attention to the numeric value of the ratings. This is while the ranking-oriented methods have nothing to do with these values and use the rank of that rate among the others. The computational time of these methods are usually higher than that of the rating-oriented methods. Furthermore, the methods come with two pairwise and listwise views (Tsuchiya & Nobuhara, 2018; Tsuchiya & Nobuhara, 2019). In the meantime, since the primary neighborhood-based algorithms were rating-oriented and a large number of these algorithms fall in this category, the focus in this study is on the similarity functions of this category. However, the meaning of this focus is not to mention that the ranking-oriented algorithms are less worthy and the only reason was to overcome a large number of articles and the original meaning of the neighborhood-based algorithms.

In this article, 33 similarity functions (113 with extension functions) have been studied and tested to obtain a complete reference of similarity functions. It has also attempted to collect all extensions for each function. The functions have been collected from available literature and implemented to evaluate their performance. The rest of this article is organized as follows: the second section deals with the similarity functions. The third section discusses the implementation, datasets, and evaluation criteria. The fourth section discusses the results, and the last section concludes the study.

2. Similarity functions

In this section, the similarity functions used in neighborhood-based collaborative filtering are described. Each function is explained in general and the details of their formula are listed in Table 2. Besides, the related extension of each function is introduced. In Table 1, the most common symbols used in formulas are briefly introduced.

2.1. Pearson correlation coefficient (PCC) and its extensions

The *PCC* similarity function is one of the most commonly used traditional similarity functions in the recommender systems. The similarity between two users is calculated via the function based on the linear dependence of their recorded rates.

$$PCC(u,v) = \frac{\sum_{i\in I_u\cap I_v}\left(R_{ui}-\overline{R_u}\right)\left(R_{vi}-\overline{R_v}\right)}{\sqrt{\sum_{i\in I_u\cap I_v}\left(R_{ui}-\overline{R_u}\right)^2} * \sqrt{\sum_{i\in I_u\cap I_v}\left(R_{vi}-\overline{R_v}\right)^2}} \quad (1)$$

In the earliest definition of this function, PCC_{old} (Resnick, Iacovou, Suchak, Bergstrom, & and Riedl, 1994), $\overline{R_u}$ is obtained from the recorded rates by the user u on the set of co-rated items. However, because of the large number of users and items in recommender systems, its calculation is time-consuming, and thus, a modified definition of the *Pearson* function is used mostly in related literature. In this definition, PCC_{new}, the value of $\overline{R_u}$ is computed using all recorded ratings by the user u (Aggarwal, 2016). In almost all related literature, this new definition of Pearson has been used.

It is clear from the formula that the *Pearson* function is defined on co-rated items. However, in recommender systems due to the sparsity problem (Marinho et al., 2012), the number of co-rated items between two users is low. Therefore, the formula may in some cases is not computable or the obtained value is not reliable (Saranya & Sadasivam, 2017). To overcome this limitation, many extensions of this function have been proposed to cover its weaknesses. For example, in the modified version of the function, *WPCC* proposed by Herlocker, Konstan, Borchers, and and Riedl (1999), the similarity obtained by *Pearson*'s formula is multiplied to a number smaller than one if the number of co-rated items between two users is less than a threshold to blueuce the value calculated by the function. Furthermore, the functions *SPCC* (Liu, Hu, Mian, Tian, & Zhu, 2014; Saranya & Sadasivam, 2017) and *ShrunkPCC* (Knees, Schnitzer, & and Flexer, 2014) use the number of co-rated items to calculate the similarity between all pairs of users, in exponential and fractional form, respectively.

Another modification on the *Pearson* function is to consider the fact that the rate distribution in recommender systems has the long tail property (Aggarwal, 2016; Leskovec, Rajaraman, & Ullman, 2014). Long-tail refers to the fact that some items are more popular and rated by many users, while some others are not popular and receive a limited number of ratings. In overall, the impact of each rate with few rating is much greater than that of each popular item. In this view, *FPC* (Aggarwal, 2016; Ricci, Rokach, Shapira, & Kantor, 2010) and PCC_{IUF} (Weng, Xu, Li, & Nayak, 2005) use the *log* function and the function *weightedPCC* (Zhang et al., 2017) uses the exponential function to obtain a value for the impact (weight) of each rate. Additionally, the *F2PC* (Lathia, Hailes, & and Capra, 2007), *PCCtsim* (Pan, Liu, & and Chang, 2017) and COR_{FRank} (Lee, 2018a) functions calculate the weight of each rate in a slightly different ways. Another modification on *Pearson* is the *NewPCC* (Sheugh & Alizadeh, 2015) function that tries to improve *Pearson* by considering the fact that in recommender systems, some cases force the output of *Pearson* to be *zero*. This *NewPCC* function detects these cases through a modified formula. The *CPCC* (Al-bashiri, Abdulgabber, Romli, & Kahtan, 2018) function is one of the most famous extensions of Pearson. It uses the median of the rates instead of the average in the *Pearson* formula. The goal is to divide the ratings into two categories of interested (positive) and uninterested (negative) (Saranya & Sadasivam, 2017). In addition, the *ModifiedCPCC* function (AL-Bakri & Hashim, 2018) is the edited *CPCC* with the same idea used in the *FPC* and *PCCtsim* functions. In 2019, Mu, Xiao, Tang, Luo, and and Yin (2019) improved the *Pearson* similarity function by introducing the *COPC* function. They used a different value in the *Pearson* formula instead of the mean or the median, similar to the scheme proposed by Pérez-Fernández, Sader, and Baets (2018). This value is differently chosen for each task. In 2019, Ayub et al. (2019) used both user and item average ratings in the *Pearson* formula. The resultant similarity is called the improved *Pearson* similarity measure and denoted by Sim_{IPCC}. Finally, the *GeneralizedDiceCoefficient* function (Luo, Xia, Zhu, & Li, 2013) is the same as the new *Pearson* function with eliminated radical in its formula. This function has been used in incremental topics to perform fewer calculations.

Table 1
The description of symbols used in the paper

Symbol	Description
R	Rating Matrix
Items	set of all items
Users	set of all users
$\|Users\| = n$	total number of all users
$\|Items\| = m$	total number of all items
u, v, w	any user in the system
i, j	any item in the system
U_i	set of all users that have rated the item i
I_u	set of all items that have rated by user u
$I_u \cap I_v$	set of co-rated items of two users u and v
R_{ui}	the submitted rate for item i by user u
$R_{min}, R_{max}, RNG, R_{med}$	in the possible values for a rating, these are minimum rate, maximum rate, maximum-minimum rate, and the median of rates
$\overline{R_i}$	the average rating of item i
$\overline{R_u}$	the average rating of user u
σ_u	the standard deviation of rating of user u
$Nbr_{u,i}$	most similar users for user u and item i
$\|x\|$	if x is a set, then denotes the cardinality of the set; else if x is a number, then is the abs function

2.2. Cosine and its extensions

Another traditional similarity function in recommender systems is the *Cosine* function. This function first transfers inputs to the vector space and then utilizes the angle between two rate vectors as the degree of the similarity of two users (Al-bashiri et al., 2018). The Cosine function is known in some works as the 1-norm (Spertus et al., 2005; Bagchi, 2015) and defined as (Ricci et al., 2010; Jannach, Zanker, Felfernig, & Friedrich, 2010):

$$Cosine\left(u,v\right) = \frac{\sum_{i\in I_u\cap I_v}(R_{ui})(R_{vi})}{\sqrt{\sum_{i\in I_u}(R_{ui})^2}\sqrt{\sum_{i\in I_v}(R_{vi})^2}} \quad (2)$$

Many extensions have been proposed for the *Cosine* function. WUP_u (Boutet et al., 2018) uses all rated items of the first user and the co-rated items of the second user in computing the formula. Thus, the similarity values obtained in this function are asymmetric, unlike the *Cosine* function. The most popular extension of the *Cosine* function is *RawCosine* (Aggarwal, 2016). Since the numerator of *Cosine* formula is deducted from the set of co-rated items, the *RawCosine* function uses this set in the denominator of the fraction as well. *AdjustCosine* and *Adjust2Cosine* (Ricci et al., 2010; Ahn, 2008) are two extensions of the *Cosine* function which are defined similar to *Pearson*. The *AdjustCosine* function calculates the average based on the users rating while the *Adjust2Cosine* function uses the items rating for calculation of the average. Another extension of the function is *CosineUnion* (Feng, Fengs, Zhang, & Peng, 2018) introduced in 2018. *CosineUnion* extends the definition of *Cosine* to a larger set of items (the sum of items rated by at least one of the two users). The aim was to blueuce the impact of Sparsity and maximize usage of the available information. The numerator of the *Cosine* function called $IP-sim$ (Lee, Park, & and Park, 2007) which solves problems of the function such as negative similarities and the denominator of the *zero* fractions as well as the normalization problem in *Cosine*.

2.3. Distance functions

Similarities can be evaluated based on the concept of the user's distance. The idea assumes that the similarity between users is

Table 2
Similarity functions for users u and v, dataset:ML100k, kn neighbours = 50. (See below-mentioned references for further information.)

	Function		Formula	MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
1	pearson correlation coefficient similarities	PCC$_{old}$ (Resnick et al., 1994)	$\frac{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})(R_{vi}-\overline{R_v})}{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v}(R_{vi}-\overline{R_v})^2}}$, $\overline{R_u}$ average rating of user u for all the co-rated items	0.90791	1.1619	0.59368	0.75304	0.44815	0.56184	0.27086
2		PCC$_{new}$ (Aggarwal, 2016)	$\frac{\sum_{i\in I_u\cap I_v}(R_{ui}-R_u)(R_{vi}-R_v)}{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v}(R_{vi}-\overline{R_v})^2}}$, $\overline{R_u}$ average rating of user u for all the items rated by the user	0.92012	1.1782	0.49759	0.75379	0.4351	0.55158	0.077351
3		PCC$_{GDC}$ (Luo et al., 2013)	$\frac{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})(R_{vi}-R_v)}{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})^2*\sum_{i\in I_u\cap I_v}(R_{vi}-\overline{R_v})^2}$, $\overline{R_u}$ aaverage rating of user u for all the items rated by the user	0.96391	1.2284	0.45299	0.73685	0.40714	0.52438	0.077351
4		WPCC (Herlocker et al., 1999)	$\begin{cases} PCC*\frac{\lvert I_u\cap I_v\rvert}{\lvert H\rvert} & \text{, if } \lvert I_u\cap I_v\rvert<\lvert H\rvert \\ PCC & \text{, otherwise}\end{cases}$, $PCC=\text{PCC}_{new}$, |H|=50	0.74454	0.9572	0.98015	0.81241	0.4778	0.6017	0.077351
5		SPCC (Liu et al., 2014)	$PCC*\frac{1}{1+\exp(-\frac{\lvert I_u\cap I_v\rvert}{2})}$, $PCC=\text{PCC}_{new}$	0.85933	1.1037	0.70263	0.76999	0.45266	0.5701	0.077351
6		Shrunk PCC (Knees et al., 2014)	$\frac{\lvert I_u\cap I_v\rvert}{\lvert I_u\cap I_v\rvert+\gamma}*PCC(u,v)$, $PCC=\text{PCC}_{new}$, γ =100	0.74289	0.95552	0.98384	0.81256	0.47956	0.60314	0.077351
7		PCCtsim (tao PAN et al., 2017)	$\frac{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})(R_{vi}-\overline{R_v})\times tr_i^{R_{ui}}\times tr_i^{R_{vi}}}{\sqrt{\sum_{i\in I_u\cap I_v}(tr_i^{R_{ui}}\times(R_{ui}-\overline{R_u}))^2}*\sqrt{\sum_{i\in I_u\cap I_v}(tr_i^{R_{vi}}\times(R_{vi}-\overline{R_v}))^2}}$, $\overline{R_u}$=average of all rated items by u, $tr_i^r=-\frac{\log_2 P_i^r}{\log_2 5}$, $r\in\{1,2,3,4,5\}$ $P_i^r=\frac{d_i^r+t}{d_i^1+d_i^2+d_i^3+d_i^4+d_i^5+5t}$, $t=2$	0.92589	1.1851	0.49695	0.75018	0.42956	0.54611	0.075683
8		CPCC (Shardanand and Maes, 1995)	$\frac{\sum_{i\in I_u\cap I_v}(R_{ui}-R_{med})(R_{vi}-R_{med})}{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-R_{med})^2}*\sqrt{\sum_{i\in I_u\cap I_v}(R_{vi}-R_{med})^2}}$	0.91238	1.1619	0.5606	0.7608	0.39396	0.519	0.17459
9		modified CPCC (AL-Bakri and Hashim, 2018)	$\frac{\sum_{i\in I_u\cap I_v}f_i^2*(R_{ui}-R_{med})(R_{vi}-R_{med})}{\sqrt{\sum_{i\in I_u\cap I_v}f_i^2*(R_{ui}-R_{med})^2}*\sqrt{\sum_{i\in I_u\cap I_v}f_i^2*(R_{vi}-R_{med})^2}}$, $f_i=\log\frac{\lvert Users\rvert}{\lvert U_i\rvert}$,$log10$	0.91866	1.17	0.54892	0.7587	0.39682	0.521	0.139
10		PCC$_{IUF}$ (Weng et al., 2005)	$\frac{\sum_{i\in I_u\cap I_v}f_i^2*(R_{ui}-\overline{R_u})(R_{vi}-\overline{R_v})}{\sqrt{\sum_{i\in I_u\cap I_v}f_i^2*(R_{ui}-\overline{R_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v}f_i^2*(R_{vi}-\overline{R_v})^2}}$, $f_i=\log\frac{\lvert Users\rvert}{\lvert U_i\rvert}$,$log10$	0.92434	1.1837	0.49235	0.75122	0.43641	0.55194	0.075685
11		FPC (Aggarwal, 2016) (Ricci et al., 2010)	$\frac{\sum_{i\in I_u\cap I_v}\lambda_i*(R_{ui}-\overline{R_u})(R_{vi}-\overline{R_v})}{\sqrt{\sum_{i\in I_u\cap I_v}\lambda_i*(R_{ui}-\overline{R_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v}\lambda_i*(R_{vi}-\overline{R_v})^2}}$, $\lambda_i=\log\frac{\lvert Users\rvert}{\lvert U_i\rvert}$, log10, $\overline{R_u}$ average rating of user u for all the items rated by the user	0.92189	1.1815	0.49377	0.75138	0.43429	0.55035	0.075685
12		F2PC (Lathia et al., 2007)	$\frac{\sum_{i\in I_u\cap I_v}d_{uv}*(R_{ui}-\overline{R_u})(R_{vi}-R_v)}{\sqrt{\sum_{i\in I_u\cap I_v}d_{uv}*(R_{ui}-\overline{R_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v}d_{uv}*(R_{vi}-\overline{R_v})^2}}$, $d_{uv}=\frac{\lvert C\rvert-\lvert D\rvert}{\lvert Items\rvert-\lvert E\rvert}$, $f_{ui}=r_{ui}-\overline{r_u}$, $,C=\{i\in Items\lvert(f_{ui}>0andf_{vi}>0)or(f_{ui}<0andf_{vi}<0)\}$, $D=\{i\in Items\lvert(f_{ui}>0andf_{vi}<0)or(f_{ui}<0andf_{vi}>0)\}$, $E=\{i\in Items\lvert f_{ui}=0orf_{vi}=0orf_{ui}=NULLorf_{vi}=NULL)\}$	0.91879	1.177	0.50576	0.75416	0.43688	0.55313	0.19577
13		NewPCC (Sheugh and Alizadeh, 2015)	$\begin{cases}\frac{(-\lvert R_u-R_v\rvert+MaxRate)-MeanRate}{MeanRate} & \text{if flat rating} \\ PCC_{new} & \text{else}\end{cases}$, $\overline{R_u}$=average of all rated items by u , $MeanRate=R_{med}$, flat rating= all of the co-rated items have the same ratings, and the mean of ratings is equal with common rating	0.9202	1.1782	0.49688	0.7405	0.43491	0.5515	0.001702

Continued on next page

	Function		Formula	MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
14		weighted PCC (Zhang et al., 2017)	$\frac{\sum_{i\in I_u\cap I_v} w(i)*(R_{ui}-R_u)(R_{vi}-R_v)}{\sqrt{\sum_{i\in I_u\cap I_v} w(i)*(R_{ui}-\overline{R_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v} w(i)*(R_{vi}-\overline{R_v})^2}}$, $\overline{R_u}$ average rating of user u for all the items rated by the user ,$w(i)=\begin{cases}\exp(-(\lvert i_f\rvert-100)^2/5000) & \lvert i_f\rvert<100\\ \exp(-(\lvert i_f\rvert-100)^2/500000) & o.w.\end{cases}$, i_f=item frequency i	0.91904	1.1768	0.49738	0.75282	0.43369	0.55025	0.075685
15		COR$_{FRank}$ (Lee, 2018a)	$\frac{\sum_{i\in I_u\cap I_v} w(i)*(R_{ui}-\overline{R_u})(R_{vi}-\overline{R_v})}{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v}(R_{vi}-\overline{R_v})^2}}$, $\overline{R_u}$ the average rating of user u of the items in co-rated items , $w(i)$ = fuzzy ranks on items, k_{ui}= the rank of a rating by user u given to item i — Fuzzy Weighted1	0.92637	1.1748	0.85764	0.71122	0.38813	0.50211	0.24298
			Fuzzy Weighted2	0.89986	1.1535	0.63272	0.75485	0.45248	0.56572	0.2596
			Fuzzy Weighted3	0.89255	1.1437	0.66189	0.7569	0.45145	0.5655	0.2595
16		COPC (Mu et al., 2019)	$\frac{\sum_{i\in I_u\cap I_v}(R_{ui}-f(U_u))(R_{vi}-f(U_v))}{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-f(U_u))^2}*\sqrt{\sum_{i\in I_u\cap I_v}(R_{vi}-f(U_v))^2}}$, $f(U_u)=argmin_{L_t\in Label}\sum_{i=1}^{m} M(f_i(U_u),L_t)$, $Label=[1,5]$, $f_i(U_u)$= represnts the rating of item i given by u. $M(f_i(U_u),L_t)$ cost of changing rating into label (zero-one distance function)	0.8977	1.1519	0.60563	0.75229	0.44991	0.56298	0.2789
17		SimIPCC (Ayub et al., 2019)	$\frac{\sum_{i\in I_u\cap I_v}(R_{ui}*R_u-R_{ui}*R_i)(R_{vi}*R_v-R_{vi}*R_i)}{\sqrt{\sum_{i\in I_u}(R_{ui}*\overline{R_u}-R_{ui}*\overline{R_i})^2}*\sqrt{\sum_{i\in I_v}(R_{vi}*\overline{R_v}-R_{vi}*\overline{R_i}))^2}}$, $\overline{R_u}$ the average rating of user u, $\overline{R_i}$ the average rating of item i	0.76699	0.98033	0.98561	0.79842	0.45382	0.57868	0.071588
18	TMJ (Sun et al., 2017)		$Triangle(i,j)*Jaccard(i,j)$, $Triangle(i,j)=1-\frac{\sqrt{\sum_{u\in U_i\cap U_j}(r_{ui}-r_{uj})^2}}{\sqrt{\sum_{u\in U_i\cap U_j}(r_{ui})^2}+\sqrt{\sum_{u\in U_i\cap U_j}(r_{uj})^2}}$	0.75518	0.96467	0.9725	0.81664	0.42376	0.55792	0.07155
19	CTJ (Yan and Tang, 2019)		$CTJ(u,v)=0.5*Jaccard(u,v)*(triangle(u,v)+1)$	0.75712	0.96723	0.97314	0.81413	0.42279	0.55649	0.07155
20	Cosine	Cosine (Ricci et al., 2010; Jannach et al., 2010)	$\frac{\sum_{k\in I_u\cap I_v}(R_{uk}).(R_{vk})}{\sqrt{\sum_{k\in I_u}(R_{uk})^2}\sqrt{\sum_{k\in I_v}(R_{vk})^2}}$	0.75176	0.96268	0.98202	0.81132	0.45027	0.57911	0.07155
21	Cosine	RawCosine (Aggarwal, 2016)	$\frac{\sum_{k\in I_u\cap I_v}(R_{uk}).(R_{vk})}{\sqrt{\sum_{k\in I_u\cap I_v}(R_{uk})^2}\sqrt{\sum_{k\in I_u\cap I_v}(R_{vk})^2}}$	0.94777	1.206	0.44747	0.74685	0.39471	0.51635	0.07155
22	Cosine	ACosine (Ahn, 2008)	$\frac{\sum_{k\in I_u\cap I_v}(R_{uk}-R_u).(R_{vk}-R_v)}{\sqrt{\sum_{k\in I_u\cap I_v}(R_{uk}-\overline{R_u})^2}\sqrt{\sum_{k\in I_u\cap I_v}(R_{vk}-\overline{R_v})^2}}$,$\overline{R_u}$ average rating of user u for all the items rated by the user	0.92012	1.1782	0.49759	0.75379	0.4351	0.55158	0.077351
23	Cosine	A2Cosine (Aggarwal, 2016)	$\frac{\sum_{k\in I_u\cap I_v}(R_{uk}-R_k).(R_{vk}-R_k)}{\sqrt{\sum_{k\in I_u\cap I_v}(R_{uk}-\overline{R_k})^2}\sqrt{\sum_{k\in I_u\cap I_v}(R_{vk}-\overline{R_k})^2}}$,$\overline{R_u}$ average rating of Item k by all the user	0.92539	1.1855	0.50985	0.7453	0.42875	0.54427	0.071733
24	Cosine	WUP-u (Boutet et al., 2018)	$\frac{\sum_{k\in I_u\cap I_v}(R_{uk}).(R_{vk})}{\sqrt{\sum_{k\in I_u}(R_{uk})^2}\sqrt{\sum_{k\in I_u\cap I_v}(R_{vk})^2}}$	0.75415	0.96642	0.99155	0.80538	0.45739	0.58341	0.07155
25	Cosine	CosineUnion (Feng et al., 2018)	$\frac{\sum_{i\in I_u\cap I_v}R_{ui}*R_{vi}+\sum_{i\in I_u-I_u\cap I_v}R_{ui}*\mu_v+\sum_{i\in I_v-I_v\cap I_v}R_{vi}*\mu_u}{\sqrt{\sum_{i\in I_u}R_{ui}^2+\sum_{i\in I_u\cup I_v-I_u}\mu_u^2}.\sqrt{\sum_{i\in I_v}R_{vi}^2+\sum_{i\in I_u\cup I_v-I_v}\mu_v^2}}$	0.80233	1.0112	0.92807	0.79638	0.3131	0.4493	0
26	Cosine	IP-sim (Lee et al., 2007)	$\sum_{i\in I_u\cap I_v}R_{ui}*R_{vi}$	0.75552	0.96851	0.99084	0.80194	0.46709	0.59032	0.07155
27		Euclidean (Huang et al., 2018; Schwarz et al., 2017; Arsan et al., 2016)	$\frac{1}{1+\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-R_{vi})^2}}$	0.9636	1.2273	0.38891	0.7381	0.4025	0.52079	0

Continued on next page

	Function		Formula	MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
28	Distance	Hamming (Wang et al., 2015; Baxla, 2014)	$USimi(u,v)=\frac{1}{\lvert I_u\cap I_v\rvert}\sum_{i\in I_u\cap I_v} Hamming(u,v,i),\qquad Hamming(u,v,i)=\lvert\frac{R_{ui}}{R_{ui}+R_{vi}}-0.5\rvert$	1.0319	1.3153	0.82006	0.68786	0.44151	0.53781	0.10961
29	Distance	Manhatan (Candillier et al., 2008)	$1-\frac{1}{R_{max}-R_{min}}\times\frac{\sum_{i\in I_u\cap I_v}\lvert R_{ui}-R_{vi}\rvert}{\lvert I_u\cap I_v\rvert}$	0.87566	1.1202	0.64127	0.76784	0.41106	0.53536	0.073597
30	Distance	simED (Sun et al., 2017)	$1-\frac{1+\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-R_{vi})^2}}{ED_{max}}$, $ED_{max}(u,v)=(R_{max}-R_{min})\sqrt{\lvert I_u\cap I_v\rvert}$	0.90821	1.1605	0.53511	0.75889	0.42189	0.5422	0.077877
31	Distance	MMD (Irish, 2010)	$\frac{1}{1+\frac{1}{r}(\sum_{i=1}^{r}(\theta_u^{\sharp i}-\theta_v^{\sharp i})^2-\frac{1}{\lvert I_u\rvert+0.5}-\frac{1}{\lvert I_v\rvert+0.5})}$, θ_u= a vector represents a vector based in user rating, $m=1,\ldots,M=5\quad rate\in\{m,\ldots,M\}$, $\theta_u=(\theta^{\sharp m},\ldots,\theta^{\sharp M})$, $\theta^{\sharp m}$ = the number of times the rating value m occurs in rating vector I_u	0.8234	1.0529	0.91586	0.77223	0.4011	0.52791	0
32	Distance	NRCF (Sun et al., 2011)	$1-\frac{\sqrt{\sum_{i\in I_u\cap I_v}(\frac{R_{ui}-R_{min-u}}{R_{max-u}-R_{min-u}}-\frac{R_{vi}-R_{min-v}}{R_{max-v}-R_{min-v}})^2}}{\sqrt{\lvert I_u\cap I_v\rvert}}$, 0/0=1	0.88509	1.1339	0.62816	0.76189	0.42462	0.54524	0.076251
33	Distance	Hellinger Distance (Mu et al., 2019)	$SimiHg(u,v)=1-\sqrt{\sum_{h=1}^{5}\sqrt{(P_{uh})(P_{vh})}}$, P_{uh} = represents the percentage, which equals h label's number divides th total number of ratings rated by user u.	1.0728	1.329	0.90086	0.56347	0.2262	0.3227	0
34	Distance	ExpDis (Moghadam et al., 2019)	$ExpDis(u,v)=\exp(-1*\frac{\sqrt{\sum_{i=1}^{m}(R_{ui}-R_{vi})^2}}{\phi\sqrt{\lvert I_u\cap I_v\rvert}})$, $\phi=R_{MAX}-R_{MIN}$	0.75934	0.97241	0.98042	0.80562	0.44865	0.57632	0.07155
35	Distance	Normalized ExpDis (Moghadam et al., 2019)	$NormalizedExpDis(u,v)=ExpDis(u,v)/(\frac{\lvert I_u\rvert}{\lvert I_u\cap I_v\rvert})$	0.75721	0.97119	0.98921	0.80208	0.46848	0.59148	0.07155
36	Distance	Quasi (Jiang et al., 2019)	$SimiQuasi(u,v)=\sum_{i\in I_u\cap I_v}(1-\frac{\lvert R_{ui}-R_{vi}\rvert^p}{((R_{max}-R_{min})/2)^p})$, $p=1/4,1/2,3/4,1$, $TheBest:p=1$	0.74942	0.96207	0.98982	0.80683	0.47103	0.5948	0.11415
37	Jaccard	Jaccard (Leskovec et al., 2014; Al-bashiri et al., 2018)	$\frac{\lvert I_u\cap I_v\rvert}{\lvert I_u\cup I_v\rvert}$	0.75939	0.97009	0.97353	0.81262	0.42301	0.55634	0.07155
38	Jaccard	Extended-Jaccard (Ayub et al., 2018)	$\sum_{j\in I_u\cap I_v}\frac{R_{uj}.R_{vj}}{R_{uj}^2+R_{vj}^2-R_{uj}.R_{vj}}$	0.75631	0.96957	0.99227	0.8007	0.46783	0.59057	0.07155
39	Jaccard	Extended-Jaccard2	$\frac{\sum_{j\in I_u\cap I_v}R_{uj}.R_{vj}}{\sum_{j\in I_u\cap I_v}R_{uj}^2+R_{vj}^2-R_{uj}.R_{vj}}$	0.88439	1.1278	0.61829	0.76935	0.40086	0.52702	0.07155
40	Jaccard	JaccardUOD (Sun et al., 2012)	$\begin{cases}\frac{\lvert I_u\cap I_v\rvert}{\lvert I_u\cup I_v\rvert}\times\frac{\sqrt{\lvert I_u\cap I_v\rvert*(R_{max-u}-R_{min-u})^2}}{\sqrt{\sum_{i\in I_u\cap I_v}(r_{ui}-r_{vi})^2}} & \exists i\in I_u\cap I_v, r_{ui}\neq r_{vi}\\ \frac{\lvert I_u\cap I_v\rvert}{\lvert I_u\cup I_v\rvert}\times\frac{\sqrt{\lvert I_u\cap I_v\rvert*(R_{max-u}-R_{min-u})^2}}{0.9+\sqrt{\sum_{i\in I_u\cap I_v}(r_{ui}-r_{vi})^2}} & O.W.\end{cases}$	0.74809	0.95683	0.97536	0.82114	0.42542	0.56043	0.10961
41	Jaccard	MyJaccard (Ayub et al., 2018)	$\frac{\lvert N_T(u,v)\rvert}{\lvert I_u\cap I_v\rvert}$, $N_T(u,v)=\begin{cases}N_T(u,v)+1 & \text{, if } \sum_{j\in I_u\cap I_v}R_{uj}==R_{vj} \text{ or } \overline{R_u}==\overline{R_v}\\ N_T(u,v) & \text{, otherwise}\end{cases}$	0.88792	1.1341	0.6506	0.76083	0.41297	0.53525	0.26757
42	Jaccard	improved Jaccard (Niu et al., 2019)	$IJ(u,v)=\frac{\sum_{i\in Items}(V_U(u)[i]\cap V_U(v)[i])/\lvert U_i\rvert}{\sum_{i\in Items}(V_U(u)[i]\cup V_U(v)[i])/\lvert U_i\rvert}$, $V_U(u)=(v_1,v_2,\ldots,v_m)$, $v_i=\begin{cases}1, & i\in I_u\\ 0, else\end{cases}$, $V_U(u)[i]$ is the i -th element of the binary user model for user u	0.76066	0.97168	0.97367	0.81333	0.42125	0.55499	0.07155
43	Jaccard	DiceCoefficient (Al-Shamri, 2014)	$\frac{2*\lvert I_u\cap I_v\rvert}{\lvert I_u\rvert\lvert I_v\rvert}$	0.75939	0.97009	0.97353	0.81262	0.42301	0.55634	0.07155
44	Jaccard	Srs (Pirasteh et al., 2015; Rupasingha and Paik, 2019)	$\frac{2(\lvert I_u\cap I_v\rvert)}{\lvert I_u\rvert+\lvert I_v\rvert}$	0.75948	0.97017	0.97353	0.8126	0.42294	0.55627	0.07155

Continued on next page

		Function	Formula	MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
45		Rounding the Data (Leskovec et al., 2014)	Jaccard for $\forall R_{wj} > t$, and $t = 3$	0.75842	0.9711	0.96524	0.81349	0.43273	0.56492	0.22022
46		tanimoto (Ricci et al., 2010; Arsan et al., 2016; Guo, 2014)	$\frac{r_u.r_v}{\lvert r_u\rvert^2 + \lvert r_v\rvert^2 - r_u.r_v} = \frac{\sum_{i=1}^{m} r_{ui}.r_{vi}}{\sum_{i=1}^{m} r_{ui}^2 + \sum_{i=1}^{m} r_{vi}^2 - \sum_{i=1}^{m} r_{ui}.r_{vi}}$, m=total number of items	0.75443	0.96532	0.97442	0.81539	0.43493	0.56724	0.07155
47		like-minded (Saeed and Mansoori, 2017)	$SimiLM(u,v) = \frac{\lvert I_u^- \cap I_v^-\rvert + \lvert I_u^0 \cap I_v^0\rvert + \lvert I_u^+ \cap I_v^+\rvert}{\lvert I_u \cup I_v\rvert}$, $I_u^- = \{i \in I_u : R_{ui} < \overline{R_u}\}$, $I_u^0 = \{i \in I_u : R_{ui} = \overline{R_u}\}$, $I_u^+ = \{i \in I_u : R_{ui} > \overline{R_u}\}$	0.75105	0.9614	0.97399	0.81811	0.43615	0.56896	0.15357
48		Sreepadalike-minded (Sreepada and Patra, 2018)	$SimiLM(u,v) = \frac{\lvert I_u^- \cap I_v^-\rvert + \lvert I_u^+ \cap I_v^+\rvert}{\lvert I_u \cup I_v\rvert}$, $I_u^- = \{i \in I_u : R_{ui} < \theta\}$, $I_u^+ = \{i \in I_u : R_{ui} > \theta\}, \theta = 3$	0.75931	0.97055	0.96856	0.8162	0.40371	0.54017	0.19904
49		CBMR (Kim et al., 2019)	$SimiCBMR(u,v) = \frac{\sum_{i \in I_u \cap I_v} \frac{1}{\log(1+\lvert U_i\rvert)}}{\sqrt{\lvert I_u\rvert * \lvert I_v\rvert}}$	0.75263	0.96336	0.98588	0.81022	0.44629	0.57551	0.07155
50	MSD	MSD (Shardanand, 1994; Shardanand and Maes, 1995; Bobadilla et al., 2012b)	$1 - \frac{\sum_{j \in I_u \cap I_v} (R_{uj} - R_{vj})^2}{I_u \cap I_v}$	0.8874	1.1361	0.60649	0.76174	0.41266	0.53518	0.15748
51		JMSD (Bobadilla et al., 2012c) newmwtric (Bobadilla et al., 2010)	$\frac{\lvert I_u\rvert \cap \lvert I_v\rvert}{\lvert I_u\rvert \cup \lvert I_v\rvert} * MSD(u,v)$	0.79261	1.02	0.83747	0.79097	0.41815	0.54702	0.15748
52		CJMSD (Bobadilla et al., 2012d)	$\frac{\lvert I_u - I_v\rvert \sum_{i \in I_u \cap I_v} (1 - (r_{ui} - r_{vi})^2)}{\lvert I\rvert.\lvert I_u \cup I_v\rvert}$	0.80586	1.0347	0.82161	0.78633	0.40865	0.53773	0.1586
53		BitJMSD (Bobadilla et al., 2013a)	$BitJaccard(u,v) - BitMSD(u,v)$, $BitJaccard(u,v) = \sum_{i \in Items} \alpha_i$, $BitMSD(u,v) = \sum_{i \in Items} \beta_i$, $\alpha_i = \begin{cases} 0 & if rsatr(u,i) = \bullet \vee rsatr(v,i) = \bullet \\ 1 & if rsatr(u,i) \neq \bullet \wedge rsatr(v,i) \neq \bullet \end{cases}$, $\gamma = 4$, $\beta_i = \begin{cases} 1 & if \alpha_i = 1 \wedge rstar(u,i) \neq rstar(v,i) \\ 0 & if \alpha_i = 0 \vee rstar(u,i) = rstar(v,i) \end{cases}$	0.75219	0.96597	0.99199	0.80324	0.47947	0.60048	0.14362
54	Spearman'sRankC	SRCC (Ahn, 2008; Singh et al., 2019b; Bagchi, 2015)	$1 - \frac{6 * \sum_{i \in I} R_{ui}^2 - R_{vi}^2}{\lvert I\rvert.(\lvert I\rvert^2 - 1)}$ $\quad I = I_u \cap I_v$	0.96202	1.2247	0.51726	0.72427	0.4167	0.52892	0.0029965
55		SRC2 (Ahn, 2008; Bagchi, 2015)	$1 - \frac{6 * \sum_{i \in I} (d_i)^2}{\lvert I\rvert.(\lvert I\rvert^2 - 1)}$ $\quad I = I_u \cap I_v$, $d_h = (R_{ui} - R_{vi})^2$	0.83971	1.0825	0.61686	0.77629	0.41649	0.54209	0.031796
56		SRankC2 (Ahn, 2008; Bagchi, 2015)	$1 - \frac{6 * \sum_{i \in I} (d_i)^2}{\lvert I\rvert.(\lvert I\rvert^2 - 1)}$ $\quad I = I_u \cap I_v$, d_h= the difference in the rate for item i by the two users	1.0786	1.3521	0.60995	0.57161	0.26955	0.3662	0
57		SRankC (Ahn, 2008; Bagchi, 2015; Kwon et al., 2009a)	$\frac{6 \sum_{i \in I} (Rank(R_{ui}) - Rnak(R_{vi}))^2}{\lvert I\rvert * (\lvert I\rvert^2 - 1)}$ $\quad I = I_u \cap I_v$	0.97348	1.2446	0.52157	0.72608	0.4619	0.5646	0.16437
58		SC (Ricci et al., 2010; Levinas, 2014; Bobadilla et al., 2012b)	$SC(u,v) = \frac{\sum_{i \in I_u \cap I_v} (rank(u,i) - rank_u) * (rank(v,i) - rank_v)}{\sigma_u * \sigma_v}$	0.91893	1.178	0.50039	0.75456	0.43262	0.54983	0.079006

Continued on next page

	Function		Formula	MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
59	PIP (Ahn, 2008)		$\sum_{i\in I_u\cap I_v} PIP(r_{ui},r_{vi})$, $PIP(r_{ui},r_{vi}) = Proximity(r_{ui},r_{vi}) * Impact(r_{ui},r_{vi}) * Popularity(r_{ui},r_{vi}),\ D(r_1,r_2) =$ $\begin{cases} \lvert r_1-r_2\rvert & , Agreement = 1 \\ 2*\lvert r_1-r_2\rvert & , Agreement = 0 \end{cases}$ $Proximity(r_1,r_2) = \{\{2*(R_{max}-R_{min})+1)\} - D(r_1,r_2)\}^2$, $Impact(r_1,r_2) = \begin{cases} (\lvert r_1-R_{med}\rvert+1)*(\lvert r_2-R_{med}\rvert+1) & , Agreement = 1 \\ \frac{1}{(\lvert r_1-R_{med}\rvert+1)*(\lvert r_2-R_{med}\rvert+1)} & , Agreement = 0 \end{cases}$, $Popularity(r_1,r_2) =$ $\begin{cases} 1+(\frac{r_1+r_2}{2}-\mu_k)^2 & , if(r_1>\mu_k\ and\ r_2>\mu_k) or(r_1<\mu_k\ and\ r_2<\mu_k) \\ 1 & O.W. \end{cases}$ $,R_{med} = \frac{R_{max}+R_{min}}{2}$, μ_k = the average rating of item k by all users	0.75281	0.96793	0.99131	0.79967	0.49115	0.60853	0.07155
60	PSS (Liu et al., 2014)		$\sum_{i\in I_u\cap I_v} PSS(r_{ui},r_{vi})$, $PSS(r_{ui},r_{vi}) = Proximity(r_{ui},r_{vi}) * Significance(r_{ui},r_{vi}) * Singularity(r_{ui},r_{vi})$, $Proximity(r_{ui},r_{vi}) = 1 - \frac{1}{1+\exp(-\lvert r_{ui}-r_{vi}\rvert)}$, $Significance(r_{ui},r_{vi}) = \frac{1}{1+\exp(-\lvert r_{ui}-r_{med}\rvert * \lvert r_{vi}-r_{med}\rvert)}$, $Singularity(r_{ui},r_{vi}) = 1 - \frac{1}{1+\exp(-\lvert \frac{r_{ui}+r_{vi}}{2} - \mu_i\rvert)}$	0.75434	0.96724	0.99091	0.80317	0.46517	0.58913	0.07155
61	NHSM (Liu et al., 2014; Al-bashiri et al., 2018; Son, 2016)		$PSSsim(u,v) * JaccSim(u,v) * URPsim(u,v),\ JaccSim(u,v) = \frac{\lvert I_u \cap I_v\rvert}{\lvert I_u\rvert \cup \lvert I_v\rvert}$, $URPsim(u,v) = 1 - \frac{1}{1+\exp(-\lvert \mu_u-\mu_v\rvert * \lvert \sigma_u-\sigma_v\rvert)}$	0.7501	0.95996	0.98055	0.81578	0.43429	0.56677	0.07155
62	multi-level	multi Level 2016 (Polatidis and Georgiadis, 2016)	$simi(u,v) = \begin{cases} PCC(u,v)+x & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t_1 \text{ and } PCC(u,v)\geq y \\ PCC(u,v)+x & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \leq t_1 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t_2 \text{ and } PCC(u,v)\geq y \\ PCC(u,v)+x & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \leq t_2 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t_3 \text{ and } PCC(u,v)\geq y \\ PCC(u,v)+x & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \leq t_3 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t_4 \text{ and } PCC(u,v)\geq y \\ 0 & \text{otherwise} \end{cases}$, $x1 = 0.5, x2 = 0.375, x3 = 0.25, x4 = 0.125, y = 0.33, t1 = 50, t2 = 20, t3 = 10, t4 = 5, T = 1$	0.76231	0.98043	0.9548	0.80239	0.46723	0.59052	0.83177
63	multi-level	multi Level 2017 (Polatidis and Georgiadis, 2017)	$simi(u,v) = \begin{cases} PCC(u,v)+PCC(u,v) & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t \text{ and } PCC(u,v)\geq y \\ PCC(u,v)*(\frac{1}{1+PCC(u,v)*PCC(u,v)}) & \text{otherwise} \end{cases}$, $y = 0.2, t = 10, T = 1$	0.78168	1.0029	0.9266	0.79677	0.45847	0.58194	0.077351
64	multi-level	Dynamic MultiLevel 2017 (Polatidis and Georgiadis, 2017)	$\begin{cases} PCC(u,v)+PCC(u,v) & \text{if } \frac{\lvert I_u\cap I_v\rvert}{M} \geq m_1 \\ PCC(u,v)+\frac{PCC}{2} & \text{if } \frac{\lvert I_u\cap I_v\rvert}{M} \leq m_1 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{M} \geq m_2 \\ PCC(u,v)+\frac{PCC}{3} & \text{if } \frac{\lvert I_u\cap I_v\rvert}{M} \leq m_2 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{M} \geq m_3 \\ \vdots & \vdots \\ PCC(u,v)+\frac{PCC}{n} & \text{if } \frac{\lvert I_u\cap I_v\rvert}{M} \leq mn_1 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{M} \geq mn_2 \\ PCC(u,v)*(\frac{(\frac{1}{1+PCC(u,v)*PCC(u,v)})}{n}-1) & \text{otherwise} \end{cases}$	0.78079	1.0032	0.92556	0.79894	0.46004	0.5838	0.077351
65	multi-level	IBCF (Alshammari et al., 2018)	$IBCF(u,v) = \begin{cases} TMJ(u,v)+x & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t_1 \text{ and } TMJ(u,v)\geq y \\ TMJ(u,v)+x & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \leq t_1 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t_2 \text{ and } TMJ(u,v)\geq y \\ TMJ(u,v)+x & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \leq t_2 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t_3 \text{ and } TMJ(u,v)\geq y \\ TMJ(u,v)+x & \text{if } \frac{\lvert I_u\cap I_v\rvert}{T} \leq t_3 \text{ and } \frac{\lvert I_u\cap I_v\rvert}{T} \geq t_4 \text{ and } TMJ(u,v)\geq y \\ 0 & \text{otherwise} \end{cases}$, $x1 = 0.5, x2 = 0.375, x3 = 0.25, x4 = 0.125, y = 0.33, t1 = 50, t2 = 20, t3 = 10, t4 = 5, T = 1$	0.64503	0.88117	0.01272	0.82052	0.52306	0.63745	0.99886

Continued on next page

	Function	Formula		MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
66	RJMSD (Jin et al., 2020; Bag et al., 2019)	$RJMSD(u,v) = (\frac{1}{1+(\frac{1}{\lvert I_u \cap I_v\rvert})+(\frac{\lvert \overline{I_u}\rvert}{1+\lvert I_u\rvert})+(\frac{1}{1+\lvert I_v\rvert})}) * (1 - \frac{\sum_{i\in I_u\cap I_v}(R_{ui}-R_{vi})^2}{\lvert I_u \cap I_v\rvert})$		0.91616	1.1737	0.48868	0.75341	0.39724	0.5201	0.085931
67	Discount — Discount1 (Aggarwal, 2016; McLaughlin and Herlocker, 2004)	$Sim(u,v) * \frac{\min\{\{I_u \cap I_v\},\beta\}}{\beta}$, $\beta = 50$	$Sim(u,v) =$ PCC	0.74454	0.9572	0.98015	0.81241	0.4778	0.6017	0.077351
			$Sim(u,v) =$ Cosine	0.75239	0.9646	0.98908	0.80576	0.46531	0.58993	0.07155
	Discount2 (Symeonidis et al., 2006)	$Sim(u,v) * \frac{\max\{\{I_u \cap I_v\},\beta\}}{\beta}$, $\beta = 5$,herlock2002: if n(co-rated) <50 then multiple n/50	$Sim(u,v) =$ PCC	0.74563	0.95978	0.98315	0.81068	0.48446	0.60648	0.077351
			$Sim(u,v) =$ Cosine	0.75416	0.9675	0.99131	0.80205	0.47277	0.59488	0.07155
68	SimiSDFS (Saeed and Mansoori, 2017)	$\mu_u(R) = \begin{cases} \frac{R-R_{min}}{\overline{R_u}-R_{min}} * 0.5 & R_{min} \le R \le \overline{R_u} \\ \frac{R-\overline{R_u}}{R_{mix}-\overline{R_u}} * 0.5 + 0.5 & \overline{R_u} \le R \le R_{max} \\ 0 & \text{otherwise} \end{cases}$, $\mu'_u(R_{ui}) = \begin{cases} \mu_u(R_{ui}) & R_{ui} \ne \bullet \\ \mu_u(\overline{R_u}) & R_{ui} = \bullet \end{cases}$, $SimiDFS(u,v) = \frac{\sum_{i\in \mathbb{I}_{u+v}} T(\mu'_u(R_{ui}),\mu'_v(R_{vi}))}{\sum_{i\in \mathbb{I}_{u+v}} S(\mu'_u(R_{ui}),\mu'_v(R_{vi}))}$, $\mathbb{I}_{u+v}$ shows the items that have been rated by at least one of the users u and v, $SimiSDFS(u,v) = \frac{\min(\sum_{i\in IUV} w_i(u,v),\gamma)}{\gamma} \times SimiDFS(u,v)$, $w_i(u,v) = \begin{cases} 1 & i \in I_u \cap I_v \\ \frac{\min(\lvert U_i\rvert,\beta)}{\beta} & i \notin I_u \cap I_v \end{cases}$, $\beta =$ the average number of ratings given by users to the items for each dataset is chosen, $\gamma = 50$, $L = 2$	$IUV = I_u \cap I_v$	0.75095	0.96423	0.98391	0.80801	0.46593	0.59102	0.073508
			$IUV = \mathbb{I}_{u+v}$,	0.78357	1.0058	0.87853	0.79504	0.44586	0.57131	0.002053
69	MJD (Bobadilla et al., 2012c)	$MJD(u,v) = \frac{1}{6}(w_1 v^0(u,v) + w_2 v^1(u,v) + w_3 v^3(u,v) + w_4 v^4(u,v) + w_5 \mu(u,v) + w_6 Jaccard)$,$\mu_{uv} = 1 - \frac{1}{\lvert I_u \cap I_v\rvert}\sum_{i\in I_u\cap I_v}(\frac{R_{ui}-R_{vi}}{R_{max}-R_{min}})^2$,$v^b$ the number of cases in which users have voted with a difference of b score, $w_1 = 0.66, w_2 = 0.35, w_3 = -0.21, w_4 = -0.43, w_5 = 1.07, w_6 = 0.31$		0.89901	1.1444	0.65323	0.74462	0.40907	0.52797	0.071551
70	SM (Bobadilla et al., 2012b)	$\frac{1}{3}\Big[\frac{1}{\lvert A\rvert}\sum_{i\in A}[1-(R_{ui}-R_{vi})^2(S_P^i)^2] + \frac{1}{\lvert B\rvert}\sum_{i\in B}[1-(R_{ui}-R_{vi})^2(S_N^i)^2]$, $+\frac{1}{\lvert C\rvert}\sum_{i\in C}[1-(R_{ui}-R_{vi})^2 * (S_P^i * S_N^i)]\Big]$, $SM(u,v) \in [0,1]$, $S_N^i = 1 - \frac{\lvert N_i\rvert}{\lvert Users\rvert} \in [0,1]$, $S_P^i = 1 - \frac{\lvert P_i\rvert}{\lvert Users\rvert} \in [0,1]$		0.84064	1.0783	0.73504	0.77267	0.42289	0.54647	0.07155
71	SimiLGGfinal (Jin et al., 2020)	$SimiLGGfinal(u,v) = SimiLocal(u,v) * SimiGlobal1(u,v) * SimiGlobal2(u,v)$, $SimiLocal(u,v) = \frac{1}{3}\big[\frac{1}{\lvert A\rvert}\sum_{i\in A}[(1-\frac{1}{1+\exp(-1 * \lvert R_{ui}-R_{vi}\rvert)}) * (S_P^i)^2] + \frac{1}{\lvert B\rvert}\sum_{i\in B}[(1-\frac{1}{1+\exp(-1 * \lvert R_{ui}-R_{vi}\rvert)}) * (S_N^i)^2] + \frac{1}{\lvert C\rvert}\sum_{i\in C}[(1-\frac{1}{1+\exp(-1 * \lvert R_{ui}-\overline{R_u}\rvert * R_{vi}-\overline{R_v}\rvert)}) * (S_P^i * S_N^i)]\big]$, $SimiGlobal1(u,v) = 1 - \exp(-1 * \frac{\lvert I_u \cap I_v\rvert}{\lvert I_u\rvert})$, $SimiGlobal2(u,v) = \sum_{t=1}^{T}\sqrt{\overleftarrow{V_u(t)} * \overleftarrow{V_v(t)}}$, $\overleftarrow{V_u}$, the user n ratings as a vector. T indicates the user's rating value, $\overleftarrow{V_u(t)}$ indicate the number of users rated the score as T , $T = 5$		0.76433	0.98005	0.99225	0.79608	0.46873	0.59003	0.071557

Continued on next page

		Function	Formula	MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
72		ModifiedHeuristic (Saranya and Sadasivam, 2017)	$ModifiedHeuristic(u,v)=w_1\times Jaccard(u,v)+w_2PSS(u,v)+MBCsimi(u,v)$, $MBCsimi(u,v)=\frac{1}{1+\exp(-\lvert BC(u,v)\rvert)}$, $w_1=w_2=0.5$	0.75415	0.96696	0.99069	0.80396	0.4629	0.58751	0
73		PNR (Wu et al., 2008)	$PNRsim(u,v)=P(u,v)*F(u,v)*G(u,v)$, $P(u,v)=1-\frac{\sum_{i\in I_u\cap I_v}\lvert r_{ui}-r_{vi}\rvert}{n.\delta}\in[0,1]$, $n=\lvert I_u\cap I_v\rvert$, $\delta=R_{max}-R_{min}$, $F(u,v)=\frac{Min(\lvert I_u\cap I_v\rvert,\alpha)}{\alpha}\in[0,1]$, $\alpha\in\mathbb{Z}^+$, $G(u,v)=\frac{Min(Max(Ratio_u,Ratio_v),\beta)}{\beta}$, $Ratio_u=\frac{\lvert I_u\cap I_v\rvert}{\lvert I_u\rvert}$, $\beta\in[0,0.5]$, $\alpha=10,\beta=0.3$, $Rmin=1,Rmax=5$	0.76721	0.98094	0.94314	0.80721	0.42504	0.55674	0.073597
74		SAD (Weng et al., 2005)	$SAD(u,v)=\frac{\sum_{i\in I_u\cap I_v}sv_{u,v,i}}{\lvert I_u\cap I_v\rvert}$, $sv_{u,v,i}=\begin{cases}\sqrt{(R_{ui}-\overline{R_u})\times(R_{vi}-\overline{R_v})}\times(1+\beta\times s_{i,+}) & \text{if } (R_{ui}-\overline{R_u})\geq 0 \text{ and } (R_{ui}-\overline{R_u})\geq 0\\ \sqrt{(R_{ui}-\overline{R_u})\times(R_{vi}-\overline{R_v})}\times(1+\beta\times s_{i,-}) & \text{if } (R_{ui}-\overline{R_u})<0 \text{ and } (R_{ui}-\overline{R_u})<0\\ \sqrt{(R_{ui}-\overline{R_i})\times(R_{vi}-\overline{R_v})} & \text{else}\end{cases}$, $s_{ri}=1-\frac{U_{ir}}{U_i}$, $s_{i,+}(s_{i,-})$=the importance of positive(negative) voting for item i	0.9739	1.2466	0.61416	0.72241	0.48096	0.5774	0.075683
75		RA(RACF) (Wu et al., 2017)	$SimiRA(u,v)=\frac{\sum_{i\in I_u\cap I_v}\frac{min(r_{ui},r_{vi})}{max(r_{ui},r_{vi})}}{\lvert I_u\cap I_v\rvert}$	0.87946	1.122	0.6434	0.76935	0.40222	0.52816	0.07155
76		CosineRec (Jeong et al., 2010)	$simi(u,v)=res(u,v)+av(u,v)$, $res(u,v)=simi(u,v)-Max_{w\neq v}\{av(u,w)+simi(u,w)\}$, $av(u,v)=Min\{0\ ,\ res(v,v)+\sum_{w\neq u,w\neq v}\text{Max}\ \{0\ ,\ res(w,v)\}\}$, $av(v,v)=\sum_{w\neq v}\text{Max}\{0\ ,\ res(w,v)\}$, itr=1000	1.1214	1.3949	0.98276	0.44617	0.18319	0.25968	0
77	combined	ImprovedCosine (Jeong et al., 2010)	$ImprovedCosine(u,v)=(Cosine(u,v))^{\omega(u,v)}$, $\omega(u,v)=(\frac{1}{Jaccard(u,v)})^{\phi}$, $\phi=0.75$	0.7632	0.97725	0.97741	0.80747	0.43895	0.5687	0.071557
78	combined	CJacMD (Suryakant and Mahara, 2016)	$CJacMD(u,v)=Cosine(u,v)+Jaccard(u,v)+MMD(u,v)$	0.75535	0.96639	0.9729	0.81332	0.43348	0.56549	0
79	combined	modifiedSimi (AL-Bakri and Hashim, 2018)	$modifiedCPCC(u,v)\times Jaccard(u,v)$	0.75589	0.96612	0.96515	0.81856	0.40338	0.54038	0.139
80	combined	Weighted PCCJacc (Zhang et al., 2017)	$Simi(u,v)=2*Jaccard(u,v)*WeightedPCC(u,v)$	0.75079	0.9646	0.96637	0.81504	0.44772	0.57793	0.075685
81	combined	PCCJacc (Saranya et al., 2016)	$Simi(u,v)=W_1*PCC(u,v)+W_2*Jaccard(u,v)$, $we=W_1=W_2=0.5$	0.86971	1.1199	0.5919	0.76655	0.44058	0.55951	0.07155
82	combined	COPC-Hg (Mu et al., 2019)	$COPC-HGSimi(u,v)=\alpha COPC(,v)+(1-\alpha)*(Hg(u,v)+Jacc(u,v))$, $\alpha=0.5$	0.76391	0.97966	0.9929	0.79434	0.4692	0.58993	0
83	combined	simiIPWR (Ayub et al., 2019)	$simiIPWR(u,v)=\alpha RPB(u,v)+\beta SimiIPCC(u,v)$, $\alpha=\beta=0.5$, $RPB(u,v)=cos(\lvert\overline{R_u}-\overline{R_v}\rvert*\lvert SD(u)-SD(v)\rvert)$, $SD(u)=\sqrt{\frac{\sum_{i\in I_u}(R_{ui}-\overline{R_u})}{\lvert I_u\rvert}}$	0.76688	0.97979	0.98512	0.79878	0.45159	0.57695	0

Continued on next page

		Function	Formula		MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
84		IPIJ (Liang et al., 2015)	$IPCC(u,v)*IJacc(u,v)$, $S_P^i=1-\frac{\lvert P_i\rvert}{\lvert Users\rvert}$, $S_N^i=1-\frac{\lvert N_i\rvert}{\lvert Users\rvert}$, $S_E^i=1-\frac{\lvert E_i\rvert}{\lvert Users\rvert}$,$IJacc(u,v)=\frac{\sum_{i\in PA}S_P^i+\sum_{i\in NA}S_N^i+\sum_{i\in D}\sqrt{S_P^i*S_N^i}}{\sum_{i\in PA}S_P^i+\sum_{i\in NA}S_N^i+\sum_{i\in D}\sqrt{S_P^i*S_N^i}+\sum_{i\in PO}\sqrt{S_P^i*S_E^i}+\sum_{i\in NO}\sqrt{S_N^i*S_E^i}}$, $IPCC(u,v)=\frac{\sum_{i\in I_u\cap I_v}(r'_{ui}-\overline{r'_u})(r'_{vi}-\overline{r'_v})}{\sqrt{\sum_{i\in I_u\cap I_v}(r'_{ui}-\overline{r'_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v}(r'_{vi}-\overline{r'_v})^2}}$, $r'_{ui}=\begin{cases}\frac{\lvert Ui\rvert}{\lvert H\rvert}*r_{ui} & \lvert U_i\rvert\le H\\ r_{ui} & O.W.\end{cases}$, $H=100$		1.0792	1.3656	0.17743	0.69503	0.5897	0.63769	0.99919
85		IPAIJ (Liang et al., 2015)	$\alpha*IPCC(u,v)+(1-\alpha)*IJacc(u,v)$, $\alpha=0.7$, $S_P^i=1-\frac{\lvert P_i\rvert}{\lvert Users\rvert}$, $S_N^i=1-\frac{\lvert N_i\rvert}{\lvert Users\rvert}$, $S_E^i=1-\frac{\lvert E_i\rvert}{\lvert Users\rvert}$,$IJacc(u,v)=\frac{\sum_{i\in PA}S_P^i+\sum_{i\in NA}S_N^i+\sum_{i\in D}\sqrt{S_P^i*S_N^i}}{\sum_{i\in PA}S_P^i+\sum_{i\in NA}S_N^i+\sum_{i\in D}\sqrt{S_P^i*S_N^i}+\sum_{i\in PO}\sqrt{S_P^i*S_E^i}+\sum_{i\in NO}\sqrt{S_N^i*S_E^i}}$, $IPCC(u,v)=\frac{\sum_{i\in I_u\cap I_v}(r'_{ui}-\overline{r'_u})(r'_{vi}-\overline{r'_v})}{\sqrt{\sum_{i\in I_u\cap I_v}(r'_{ui}-\overline{r'_u})^2}*\sqrt{\sum_{i\in I_u\cap I_v}(r'_{vi}-\overline{r'_v})^2}}$, $r'_{ui}=\begin{cases}\frac{\lvert Ui\rvert}{\lvert H\rvert}*r_{ui} & \lvert U_i\rvert\le H\\ r_{ui} & O.W.\end{cases}$, $H=100$		0.90785	1.1618	0.59365	0.75326	0.44816	0.5619	0.25419
86		kendall (Yang et al., 2009; Sarwar et al., 2001; Liu and Yang, 2008; Xu et al., 2014)	$KendallSimi(u,v)=1-\frac{4\times\sum_{i,j\in\{I_u\cap I_v\}}I^-((R_{ui}-R_{uj})(R_{vi}-R_{vj}))}{\lvert I_u\cap I_v\rvert.(\lvert I_u\cap I_v\rvert-1)}$, $I^-(x)=\begin{cases}1 & \text{if } x<0\\ 0 & \text{O.W}\end{cases}$		0.91594	1.1649	0.69943	0.74146	0.4257	0.5405	0.16329
87	Entropy	SimE (Wei et al., 2015)	$SimE(u,v)=1-\frac{H}{\log_2\lvert I_u\cap I_v\rvert}$, $\Delta d=\sum_{i\in I_u\cap I_v}\Delta d_i$, $p_i=\frac{\Delta d_i}{\Delta d}$, $\sum_i p_i=1$, $H=-\sum_{i\in I_u\cap I_v}p_i\log_2(p_i)$, $\Delta d_i=\lvert(r_{ui}-\overline{r_u})-(r_{vi}-\overline{r_v})\rvert, i\in I_u\cap I_v$, $\overline{r_u}$ =mean rate of all items that rated by user u		0.95101	1.2281	0.41823	0.73037	0.41024	0.52527	0.11851
88		SimEntropy (Wei et al., 2015)	$SimEntropy(u,v)=\alpha*simPCC(u,v)+(1-\alpha)*simE(u,v)$, $\alpha=0.7$		0.91444	1.1785	0.49388	0.75331	0.43576	0.55208	0.0029414
89		Rel (Wei et al., 2015)	$Rel_{ij}=\frac{D_{ij}}{\lvert U_i\cap U_j\rvert}$, $D_{ij}=\sum_{p\in U_i\cap U_j}\Delta r_p*$, $\Delta r_p*=\begin{cases}1 & 0\le\Delta r_i<\frac{(R_{max}-R_{min})}{3}\\ 0.5 & \frac{(R_{max}-R_{min})}{3}\le\Delta r_i<\frac{2(R_{max}-R_{min})}{3}\\ 0 & \frac{2(R_{max}-R_{min})}{3}\le\Delta r_i<(R_{max}-R_{min})\end{cases}$, $\Delta r_p=\lvert r_{pi}-r_{pj}\rvert$		0.88661	1.1327	0.62611	0.76098	0.41629	0.53787	0.082153
90		PROP1 (Lee, 2018b)	$PROP(u,v)=1-sig(\frac{1}{\lvert I_u\cap I_v\rvert}\sum_{i\in I_u\cap I_v}\frac{(r_{ui}-r_{vi})^2}{E(i)})$	$sig(x)=\frac{1}{1+\exp(-x)}$	0.97739	1.2363	0.37251	0.7277	0.32105	0.44416	0
				$sig(x)=\frac{\exp(2x)-1}{\exp(2x)+1}$	0.98108	1.2347	0.48267	0.71295	0.29748	0.41947	0.10742
91		SimEW1 (Kwon et al., 2009b)	$simEW(u,v)=Simi(u,v)*\frac{1}{1+\lvert H(u)-H(v)\rvert}$, $H(u)=-\sum_{i\in I}p(x_i)\log_2(p(x_i))$	$Simi=PCC$	0.89298	1.1508	0.57453	0.7579	0.45738	0.57041	0.077351
				$Simi=EclideanSimi$	0.95584	1.2248	0.3926	0.73897	0.43197	0.54518	0
				$Simi=MSD$	0.88102	1.1314	0.61982	0.76415	0.42656	0.54739	0.15748
92		SimEn (Piao et al., 2007, 2009)	$SimiEn(x,y)=H(x)+H(y)-H(x,y)$, $H(x,y)=H(x)\times H(y)$, $P_{ux}=\frac{\text{number of items rated by user u}}{\text{total number of items}}\times\frac{\text{number of users rating on item x}}{\text{total number of users}}$, $H(x)=-\sum_{u=1}^{\lvert Users\rvert}P_{ux}.\log_2(P_{ux})$		1.094	1.3655	0.71359	0.58302	0.26838	0.36705	0

Continued on next page

		Function	Formula		MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
93		NBCE (Li and Zheng, 2015)	$NBCE(u,v) = -1 + \frac{2(2 - BCE(u,v))}{2-(-1)} \in [-1,1]$, $BCE(u,v) = Habit(u,v) + PCC(u,v) \in [-1,2]$, $Habit(u,v) = \begin{cases} En(u,v) & BC(u,v)=0 \\ \sqrt{En(u,v) * BC(u,v)} & BC(u,v) \neq 0 \end{cases} \in [0,1]$, $En(u,v) = \exp(-\vert H(u) - H(v)\vert)$, $H(u) = -\sum_{k\in[R_{min},R_{max}]} P_{uk} * \log(P_{uk})$, $BC(u,v) = \sum_{k\in[R_{min},R_{max}]} \sqrt{P_{uk} * P_{vk}}$		0.98324	1.2422	0.53827	0.7055	0.36688	0.48264	0
94		PCCEntropy (Lee, 2018c)	$\frac{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})(R_{vi}-\overline{R_v}).E(i)}{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})^2} * \sqrt{\sum_{i\in I_u\cap I_v}(R_{vi}-\overline{R_v})^2}}$, $E(i) = -\sum_{k\in[R_{min},R_{max}]} prob(r_i = k)\log_2(prob(r_i = k))$, $prob(r_i = k) = \frac{\vert\{u \in Users \vert r_{ui} = k\}\vert}{\vert\{u \in Users \vert r_{ui} \neq NULL\}\vert}$		0.92011	1.1798	0.47827	0.74735	0.44325	0.55634	0.075683
95		CosineEntropy (Lee, 2018c)	$\frac{\sum_{k\in I_u\cap I_v}(r_{uk}).(r_{vk}).E(k)}{\sqrt{\sum_{k\in I_u}(r_{uk})^2}\sqrt{\sum_{k\in I_v}(r_{vk})^2}}$, $E(i) = -\sum_{k\in[R_{min},R_{max}]} prob(r_i = k)\log_2(prob(r_i = k))$, $prob(r_i = k) = \frac{\vert\{u \in Users \vert r_{ui} = k\}\vert}{\vert\{u \in Users \vert r_{ui} \neq NULL\}\vert}$		0.75153	0.96304	0.98228	0.81004	0.45561	0.58318	0.07155
96		SimiWDE (Kwon et al., 2009a)	$SimiWDE(u,v) = -\sum_{i\in I_u\cap I_v} p(d_i)\log_2(d_i) \times \vert d_i\vert$, $d_i = abs(r_{ui} - r_{vi})$, $p(d_i)$= like simE		0.75486	0.96841	0.99031	0.80348	0.46502	0.58908	0.23568
97	ItemWeighted	INSC (Zhang and Andreae, 2008)	$Simi(u,v,i) = \frac{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})(R_{vi}-\overline{R_v}) * rel(u,v,i,j)}{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-\overline{R_u})^2 * rel(u,v,i,j)} * \sqrt{\sum_{i\in I_u\cap I_v}(R_{vi}-\overline{R_v})^2 * rel(u,v,i,j)}}$	$rel(u,v,i,j) = pcc(i,j)$	1.0712	1.0812-0.0094053i	0.72585	0.70074	0.32059	0.43969	-
				$rel(u,v,i,j) = abs(pcc(i,j))$	0.92283	1.1798	0.48699	0.75209	0.4378	0.55334	-
98		Weighted Distance (Huang and Dai, 2015)	$Simi(u,v,i) = WD(u,v,i) * Jaccard(u,v)$, $WD(u,v,i) = \frac{\sum_{j\in I_u\cap I_v} ISimi(i,j)}{1 + \sum_{j\in I_u\cap I_v}\{ISimi(i,j) * \vert r_{uj} - r_{vj}\vert\}}$	$ISimi = PCC(new)$	0.77734	1.01	0.96925	0.78292	0.51772	0.62322	-
				$ISimi = Cosine$	0.74515	0.9541	0.9864	0.82214	0.43807	0.57155	-
99		EEdited (Choi and Suh, 2013)	$EEdit(u,v,i) = \frac{1}{1 + \sqrt{\sum_{j\in Items} Isimi(i,j)^2 * (r_{uj} - r_{vj})^2}}$	$Isimi = pcc(new)$	0.95789	1.2192	0.39838	0.7412	0.40639	0.52478	-
				$Isimi = Cosine$	0.95486	1.2166	0.40117	0.74331	0.40475	0.52397	-
				$Isimi = EuclideanSim$	0.96364	1.2275	0.38406	0.73641	0.40401	0.52163	-
100		CosineEdited (Choi and Suh, 2013)	$CosineEdit(u,v,i) = \frac{\sum_{j\in Items} Isimi(i,j)^2 * (r_{uj}) * (r_{vj})}{\sqrt{\sum_{j\in Items} Isimi(i,j)^2 * (r_{uj})^2} * \sqrt{\sum_{j\in Items} Isimi(i,j)^2 * (r_{vj})^2}}$	$Isimi = pcc(new)$	0.76378	0.97638	0.97145	0.80978	0.42214	0.55494	-
				$Isimi = Cosine$	0.77217	0.98442	0.96138	0.82743	0.38475	0.52521	-
				$Isimi = EuclideanSim$	0.8171	1.0328	0.94821	0.82196	0.30798	0.44796	-
101		PCCEdited (Choi and Suh, 2013)	$PCCEdit(u,v,i) = \frac{\sum_{j\in Items} Isimi(i,j)^2 * (r_{uj} - \overline{r_u}) * (r_{vj} - \overline{r_v})}{\sqrt{\sum_{j\in Items} Isimi(i,j)^2 * (r_{uj} - \overline{r_u})^2} * \sqrt{\sum_{j\in Items} Isimi(i,j)^2 * (r_{vj} - \overline{r_v})^2}}$	$Isimi = pcc(new)$	0.93006	1.1894	0.47845	0.75421	0.44183	0.55709	-
				$Isimi = Cosine$	0.92322	1.1815	0.47946	0.75565	0.43817	0.55456	-
				$Isimi = EuclideanSim$	0.93678	1.1983	0.48417	0.75282	0.42802	0.54562	-

Continued on next page

	Function	Formula		MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
102	hybrid (Wang et al., 2017)	$S(u,v) = S_2(u,v) * S_3(u,v) * (\sum_{i\in I_u}\sum_{j\in I_v} S_{item}(i,j) * S_1(r_{ui}, r_{vj}))$, $S_{item}(i,j) = \frac{1}{1+D_s(i,j)}$, $D_s(i,j) = \frac{D_s(i\|\|j) + D_s(j\|\|i)}{2}$, $D_s(i,j) = D(p_i\|\|p_j) = \sum_{h=1}^{r} p_{ih}\log_2\frac{p_{ih}}{p_{jh}}$, $p1_{ih} = \frac{\sharp h}{\sharp i}$, $p_{hi} = \frac{\delta + p1_{ih}}{1+\delta * \|D\|}$, $D = 5$, $we: \delta = 0.1$, $S_1(r_{ui}, r_{vj})) = proximity(r_{ui}, r_{vj})) * significance(r_{ui}, r_{vj})) * singularity(r_{ui}, r_{vj}))$, $S_2 = \frac{1}{1+\exp(-\frac{\|I_u\cap I_v\|}{\|I_u\|})}$, $S_3 = 1 - \frac{1}{1+\exp(-\|\mu_u - \mu_v\| * \|\sigma_u - \sigma_v\|)}$		0.77345	0.99097	0.99134	0.79027	0.45752	0.57949	0
103	SimiKL (Deng et al., 2019)	$SimiKL(u,v) = \sum_{i\in I_u}\sum_{j\in I_v} S_{item}(i,j) * S(r_{ui}, r_{vj})$, $S(r_{ui}, r_{vj}) = \frac{1}{1+\exp(-(R_{ui}-\overline{R_u})(R_{vi}-\overline{R_v}))}$, $S_{item}(i,j) = \frac{1}{1+D_s(i,j)}$, $D_s(i,j) = \frac{D'_s(i\|\|j) + D'_s(j\|\|i)}{2}$, $D'_s(i,j) = D(p_i\|\|p_j) = \sum_{h=1}^{r}\lambda p_{ih}\log_2\frac{\lambda p_{ih}}{p_{jh}}$, $p1_{ih} = \frac{\sharp h}{\sharp i}$, $p_{hi} = \frac{\delta + p1_{ih}}{1+\delta * \|V\|}$, $\|V\| = 5$, $\lambda = \sharp i/(\sharp i + \sharp j)$ is the ratio of ratings on item i, $\delta = 0.00001$		0.77987	1.0003	0.9933	0.78517	0.47476	0.5917	0
104	proposedSimi (Feng et al., 2018)	$S(u,v) = S_1(u,v) * S_2(u,v) * S_3(u,v)$, $if\ sparsity\ of\ dataset < \rho\ S_1(u,v) = Cosine(u,v),\ else\ S_1(u,v) = \frac{\sum_{i\in I_u\cap I_v} R_{ui} * R_{vi} + \sum_{i\in I_u - I_u\cap I_v} R_{ui} * \mu_v + \sum_{i\in I_u - I_v\cap I_v} R_{vi} * \mu_u}{\sqrt{\sum_{i\in I_u} R_{ui}^2 + \sum_{i\in I_u\cup I_v - I_u}\mu_u^2}\cdot\sqrt{\sum_{i\in I_v} R_{vi}^2 + \sum_{i\in I_u\cup I_v - I_v}\mu_v^2}}$, $S_2(u,v) = \frac{1}{1+\exp(-\frac{\|I_u\cap I_v\|^2}{\|I_u\|.\|I_v\|})}$, $S_3 = 1 - \frac{1}{1+\exp(-\|\mu_u - \mu_v\| * \|\sigma_u - \sigma_v\|)}$, $sparsity = 93.7$	$\rho = 1$	0.74962	0.96047	0.98142	0.81445	0.44739	0.57751	0.07155
			$\rho = 0$	0.75899	0.97063	0.97595	0.81199	0.41206	0.54666	0
105	CPB (Deshpande and Karypis, 2004)	$CPB(u,v) = \frac{CP(uv)}{Freq(v).(Freq(u))^{\alpha}}$, $we: \alpha = 0.5$	$CP(uv) = Freq(uv)$	0.77283	0.99147	0.99359	0.78758	0.47802	0.59494	0.07155
			$CP(uv) = \frac{\sum_{i\in I_u} R_{ui}}{\|\|R_u\|\|_2}$	0.77985	1.0003	0.9933	0.78509	0.47474	0.59167	0
106	ACSimi (Pirasteh et al., 2015)	$ACSimi(u,v) = A(u,v) * C(u,v) * Simi(u,v)$, $C(u,v) = 1 - \exp(-\frac{\|I_u\cap I_v\|}{\|I_u\|})$, $A(u,v) = \frac{\overrightarrow{V_u}.\overrightarrow{V_v}}{\|\|\overrightarrow{V_u}\|\|.\|\|\overrightarrow{V_v}\|\|}$	$Simi = PCC(new)$	0.74091	0.9534	0.98581	0.81586	0.48255	0.60642	0.077351
			$Simi = Cosine$	0.75222	0.96465	0.98929	0.80611	0.46951	0.5934	0.07155
			$Simi = MSD$	0.77977	1.0058	0.87125	0.79452	0.44221	0.56809	0.15748
			$Simi = Srs$	0.75256	0.96403	0.98722	0.81023	0.45513	0.58283	0.07155
107	RFS (ur Rehman et al., 2013)	$RFS(u,v) = \frac{WSUM}{CORAT}$, $WSUM = \sum_{i\in[1,RNG]} F_i * D_i$, F_j= number of times j occurs in the diefference, j =diffrence of rate of co-rated items, $CORAT = \sum_{i\in[1,RNG]} i/RNG$, $RNG = 5$	$D_i = i/RNG$	0.77038	0.98615	0.99321	0.79103	0.46275	0.58389	0.07155
			$D_i = RNG/i$	0.75397	0.96731	0.99215	0.80221	0.47173	0.5941	0.07155
108	SOFA (Yao et al., 2013)	$\frac{1}{2}(w_1(u,v) + w_2(u,v))$, $w_2(u,v) = \frac{\mu_u * \mu_v + \sigma_u^2 + \sigma_v^2}{\sqrt{(\mu_u + \sigma_u^2) * (\mu_v + \sigma_v^2)}}$, $K = 10$	$w1(u,v) = PCC(u,v)$	0.7917	1.0075	0.9182	0.80007	0.3821	0.51704	0
			$w1(u,v) = Cosine(u,v)$	0.79984	1.0122	0.94809	0.79562	0.33227	0.46852	0
109	RMS (Kwon and Hong, 2011, 2013)	$RMS(u,v) = 1 - \frac{1}{r^k}\frac{1}{n}\sum_{i=1}^{n}(\|R_{ui} - R_{vi}\|)^k$	$k = 0.5, r = 5$	0.9058	1.1584	0.51349	0.75765	0.39532	0.51944	0
			$k = 1.5, r = 5$	0.90866	1.1618	0.49559	0.75605	0.39427	0.51811	0
			$k = 2, r = 5$	0.9108	1.1648	0.48644	0.75537	0.3962	0.51961	0

Continued on next page

	Function	Formula		MAE	RMSE	cvrge	prcnRte	rcllRte	F1Rte	Sim0
110	RES (Physical Resonance Principle) (Tan and He, 2017)	$Sim_{RES}(u,v) = \frac{ArcTan(RES(u,v))}{0.5\pi}$, $RES(u,v) = \ sum_{I_{uv}} C(u,v,k_1) * D(u,v,k_2,k_3) * J(u,v,k_4)$, $C(u,v,k_1) = (\sqrt{0.5 + 0.5 * Cosine(\phi(u) - \phi(v))})^{k_1}$, $\phi(u) = \begin{cases} \frac{\pi}{R_{max} - Rmin}.\phi^{+} & when(r_{ui} - R_{med}) * (\overline{r_u} - \mu) \geq 0 \\ \frac{\pi}{R_{max} - Rmin}.\phi^{-} & when(r_{ui} - R_{med}) * (\overline{r_u} - \mu) < 0 \end{cases}$, $\phi^{+} = \left(1 + \frac{\lvert\overline{r_u} - \mu\rvert}{\frac{R_{max} - R_{min}}{2}}.(r_{ui} - R_{med})\right)$, $\phi^{-} = \left(\frac{1}{1 + \lvert 1 + \lvert\overline{r_u} - \mu\rvert}.(r_{ui} - R_{med})\right)$, $D(u,v,K_2,k_3) =$ $\begin{cases} \exp(0.5 * (\lvert r_{ui} - \overline{r_i}\rvert + \lvert r_{vi} - \overline{r_i}\rvert))_2^k & when(r_{ui} - \overline{r_i})(r_{vi} - \overline{r_i}) \geq 0 \\ \exp(-\lvert r_{ui} - r_{vi}\rvert)_3^k & when(r_{ui} - \overline{r_i})(r_{vi} - \overline{r_i}) < 0 \end{cases}$, $Rmed = (Rmin + Rmax)/2$, $Rmin = 1$, $Rmax = 5$, μ= average rating for all items.	$k_1 = 1$, $k_2 = 1$, $k_3 = 1$, $k_4 = 1$	0.74989	0.96141	0.98808	0.81105	0.4618	0.5885	0.07155
			$k_1 = 22$, $k_2 = 0$, $k_3 = 1.4$, $k_4 = 2$	0.76014	0.97279	0.98205	0.80666	0.44195	0.57101	0.07155
111	significances (Bobadilla et al., 2012a)	$S_i = (\frac{1}{\lvert U_i\rvert} \sum_{u \in U_i} r_{ui}) * (\frac{\lvert U_i\rvert}{\lvert Users\rvert}) = \frac{\sum_{u \in U_i} r_{ui}}{\lvert Users\rvert} \in [0,1], r_{ui} \in [0,1]$, $S_u = (1 - \frac{\lvert D_u\rvert}{\lvert D_u\rvert + \lvert E_u\rvert}) * (\frac{\lvert D_u\rvert + \lvert E_u\rvert}{\lvert Items\rvert}) \in [0,1]$, if R(u,i) has a value:$S_{ui} = r_{ui} * S_i * S_u \ \in [0,1]$,if R(u,i) dose not exist but si and su exist:$S_{ui} = S_u * S_i * \frac{\sum_{j \in N_{ui}} Simi(i,j) * S_j * r_{uj}}{\sum_{j \in N_{ui}} Simi(i,j)}$,$Simi = EuclideanSimi, \lvert N_{ui}\rvert = 50$, else: $S(u,i)$= null	$simiFunc = Cosine(S_{ui})$	0.76537	0.97955	0.97831	0.80045	0.4455	0.57238	0.010659
			$simiFunc = PCC(S_{ui})$	0.81834	1.0497	0.73117	0.76985	0.42722	0.54941	0.010659
			$simiFunc = MSD(S_{ui})$	0.82759	1.054	0.79021	0.77686	0.36422	0.49587	0.010659
112	BCF (Patra et al., 2015, 2014)	$BCF(u,v) = Jaccard(u,v) + \sum_{i \in I_u} \sum_{j \in I_v} BC(i,j) * loc(r_{ui}, r_{vj})$, $loc(r_{ui}, r_{vj}) = \frac{(r_{ui} - R_u)(r_{vj} - R_v)}{\sqrt{\sum_{k \in I_u}(r_{uk} - R_u)^2} * \sqrt{\sum_{k \in I_v}(r_{vk} - R_v)^2}}$, $BC(i,j) = BC(\hat{p_i}, \hat{p_j}) = \sum_{h=1}^{m} \sqrt{\hat{p_{ih}} * \hat{p_{jh}}}$, $\hat{p_{ih}} = \frac{\sharp h}{\sharp i}, \sharp h = numberofuserswhoratedthevaluehtotheitemi, \sharp i = totalnumberofuserswhorateditemi$	$R_u = \overline{r_u}$	0.76351	0.97487	0.96692	0.80952	0.41734	0.55062	0.07155
			$R_u = R_{med} = 3$	0.76312	0.97451	0.96662	0.80909	0.41727	0.55046	0.07155
113	NewUsersimilarity (Shen and Zhou, 2010; Xiaoping, 2015)	, $c(u,v) = \sum_{i \in I_u \cap I_v}(1 - (R_{max} - R_{min})) * abs(r_{ui} - r_{vi})$, $d(u,v) = (\lvert I_u\rvert - \lvert I_u \cap I_v\rvert) + ((\lvert I_v\rvert - \lvert I_u \cap I_v\rvert)$	$Simi(u,v) = \frac{c(u,v)}{c(u,v) + d(u,v)}$	0.94636	1.2058	0.62089	0.71305	0.39667	0.50972	0.11038
			$Simi(u,v) = \frac{e^{(\alpha * c(u,v)} - 1}{e^{\alpha * c(u,v)} + e^{\beta * d(u,v)} - 2}$, $\alpha = \beta = 0.01$	1.201	1.4947	0.49737	0.51079	0.21121	0.29873	0.10961

increasing with decreasing distance between them (Bagchi, 2015). The distance is calculated via the formula:

$$Distance\left(u,v\right) = \left(\sum_{i\in I_u\cap I_v} \left|R_{ui} - R_{vi}\right|^p\right)^{\frac{1}{p}}, \qquad p = 1,2,\ldots \tag{3}$$

where different values of the parameter p yields a different distance function. The distance function has been studied in different literature (Huang et al., 2018; Schwarz, Lobur, & Stekh, 2017; Arsan et al., 2016). The analysis reported by Schwarz et al. (2017) revealed the *Pearson* and *Cosine* functions have faults while the inverse *Euclidean* distance between two vectors increases the accuracy of the similarity. Hence, many distance-based similarity functions have been proposed including *Euclidean* distance (Baxla, 2014) (2-norm), *Hamming* distance (Wang, Zhao, & Hong, 2015), *Manhattan* distance (Candillier, Meyer, & Fessant, 2008) (1-norm), *Quasi* distance (Jiang, Fang, An, & Lavery, 2019), *simED* distance (Sun et al., 2017) and *MMD* distance (Irish, 2010). Among these, *MMD* is the most famous commonly used function for distance calculation in non-metric spaces. *NRCF* (Sun, Zheng, Chen, & Lyu, 2011) is another distance-based similarity function that first normalizes all the rates of each row of the matrix to obtain numbers in the interval [0,1], and then, uses an idea like to the *Euclidean* distance to evaluate the similarity. The *Hellinger* distance function, which is defined in the statistics to calculate the separability of two discrete probability distributions, was used by Mu et al. (2019) to compute the similarity in collaborative filtering. In 2019 Moghadam, Heidari, Moeini, and Kamandi (2019) have introduced another distance-based function that evaluate the similarity using an exponential function.

2.4. Jaccard and its extensions

In 1998, Koutrika and Bercovitz introduced the *Jaccard* function for calculation of the relationship between two users (Al-bashiri et al., 2018). The *Jaccard* function only considers the number of co-rated items between two users without using the actual value of the ratings (Leskovec et al., 2014). This is known as one of the weaknesses of the *Jaccard* function (Saranya & Sadasivam, 2017). The function is defined as:

$$Jacc\left(u,v\right) = \frac{|I_u \cap I_v|}{|I_u \cup I_v|} \tag{4}$$

Many similarity functions have been proposed based on the *Jaccard* idea. *ExtendedJaccard* (Ayub, Ghazanfar, Maqsood, & Saleem, 2018) introduced to involve the value of rates in the similarity calculation. The *JaccardUniformOperatorDistance* (Sun et al., 2012; Saranya et al., 2016) similarity function tries to find an accurate way for calculation of similarity by integrating the rate vector space. In 2018, Ayub et al. (2018) introduced another definition of *Jaccard* called *myJaccard*. In addition to the number of co-rated items, the equality of ratings and average ratings of users are involved in this function for evaluating similarity. In 2019, Niu (Niu et al., 2019) proposed an extended *Jaccard* function for computing the similarity with popularity normalization. The *Dice-Coefcient* function provided by Al-Shammari (Al-Shamri, 2014) is the same as the *Jaccard* function, with a difference in multiplying the number of co-rated items by two. The *RoundingtheData* (Leskovec et al., 2014) function removes the rates less than a certain value in the rating matrix and then keeps all remaining rates the same using the similarity criterion *Jaccard*. The *Tanimoto* (Arsan et al., 2016; Guo, 2014) and *TermSimilarityWeight* (Rupasingha & Paik, 2019) functions like *Jaccard*, do not consider the ratings. They consider the similarity between two sets of items via the ratio of the intersection of two sets. The *Srs* (Pirasteh, Hwang, & Jung, 2015; Rupasingha & Paik, 2019) function measures the degree of correlation between the ratings of co-rated items between two users. Of course, the idea of this function is also derived from the *Jaccard* function. Another function that can be included in this category is the *like−mindedSimi* function that was introduced in 2017 by Saeed and Mansoori (2017). This function first divides the ratings of each user into three categories, and then considers the similary of each two users equal to the total number of intersections in each category to the total number of rated items. Based on the same idea as likeminded, in 2018, Sreepada and Patra (2018) introduced another function with a difference by dividing the rates into two categories. The *CBMR* similarity function is also proposed by Kim, Kim, and Min (2019) with the same idea as the *Jaccard* function.

2.5. Improved Pearson and Jaccard correlation (IPIJ)

In 2015, Liang, Ma, and Yuan (2015) introduced the *IPCC* (Improved Pearson Correlation Coefficient) and *IJacc* (Improved Jaccard) similarity functions. To calculate the *IPCC* function, first, changes are made on the rates and then *Pearson* is applied to them. Changes on the rates are done for the items having several users who rated this item less than a threshold, and thus, these ratings will be worthless. The threshold value is evaluated experimentally. Then, for both users the number of items is enumerated including the positively rated item by both users, the negative rated item by both users, a positive and a negative rated item by these two users, and finally, a positive or negative rated item by a user without rating by the other user. Estimation of the similarities between two users is calculated based on the *Jaccard*'s idea on these numbers. The final *IPIJ* and *IPAIJ* functions are obtained via the following formulas:

$$IPIJ(u,v) = IPCC(u,v)^{*}IJacc(u,v)$$

$$IPAIJ(u,v) = \alpha^{*}IPCC(u,v) + (1-\alpha)^{*}IJacc(u,v), \quad \alpha = 0.7$$

$$S_P^i = 1 - \frac{|P_i|}{|Users|},\ S_N^i = 1 - \frac{|N_i|}{|Users|},\ S_E^i = 1 - \frac{|E_i|}{|Users|}$$

$$IJacc\left(u,v\right) = \frac{\sum_{i\in PA} S_P^i + \sum_{i\in NA} S_N^i + \sum_{i\in D}\sqrt{S_P^{i*}S_N^i}}{\sum_{i\in PA} S_P^i + \sum_{i\in NA} S_N^i + \sum_{i\in D}\sqrt{S_P^{i*}S_N^i} + \sum_{i\in PO}\sqrt{S_P^{i*}S_E^i} + \sum_{i\in NO}\sqrt{S_N^{i*}S_E^i}}$$

$$IPCC\left(u,v\right) = \frac{\sum_{i\in I_u\cap I_v}\left(r'_{ui} - \overline{r'_u}\right)\left(r'_{vi} - \overline{r'_v}\right)}{\sqrt{\sum_{i\in I_u\cap I_v}\left(r'_{ui} - \overline{r'_u}\right)^2}^{*}\sqrt{\sum_{i\in I_u\cap I_v}\left(r'_{vi} - \overline{r'_v}\right)^2}}$$

$$r'_{ui} = \begin{cases} \frac{|Ui|}{|H|}^{*}R_{ui} & \text{if } |U_i|\leqslant H \\ R_{ui} & O.W. \end{cases} \tag{5}$$

2.6. Triangle Multiplying Jaccard (TMJ)

In 2017, Sun et al. (2017) introduced a similarity function made from the combination of similarities of *Triangle* and *Jaccard*. The *Triangle* function uses the length and angle between two rating vectors while the *Jaccard* function considers the number of co-rated items.

$$TMJ(u,v) = Jaccard(u,v)*\left(1-\frac{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui}-R_{vi})^2}}{\sqrt{\sum_{i\in I_u\cap I_v}(R_{ui})^2}+\sqrt{\sum_{i\in I_u\cap I_v}(R_{vi})^2}}\right) \quad (6)$$

In 2019, Yan and Tang (2019) proposed an algorithm similar to this idea. The algorithm first carries out a clustering on users and items, and then, computes the similarity by combining the *Jaccard* and *Triangle* functions in a different way.

2.7. Mean Squablue Difference (MSD) and its extensions

The *MSD* function measures the similarity using the value of rates (Shardanand, 1994; Hassanieh et al., 2018; Sivaramakrishnan, Subramaniyaswamy, Arunkumar, Renugadevi, & Ashikamai, 2018).

$$MSD(u,v) = 1-\frac{\sum_{j\in I_u\cap I_v}(R_{uj}-R_{vj})^2}{|I_u\cap I_v|} \quad (7)$$

A special type of *MSD* is called *JMSD* (Bobadilla, Ortega, Hernando, & Bernal, 2012; Bobadilla, Serradilla, & Bernal, 2010) that combines the *MSD* and *Jaccard* function to slightly cover the disadvantages of each other using all the rates that are recorded by two users. However this function is still suffering from the cold start problem (Saranya & Sadasivam, 2017; Bobadilla et al., 2012). *CJMSD* is another function proposed by Bobadilla, Ortega, Hernando, and Arroyo (2012) face with the problem of long computation time because it has to use the dependent and independent ratings for each pair of users. Note that this function is asymmetric. In another work, Bobadilla, Ortega, Hernando, and de Rivera (2013) proposed the *BitJMSD* function similar to the *JMSD*. Regarding the existing common problem in long processing time for computing similarity and selecting *n* neighborhoods, Bobadilla et al. tried to blueuce the computational time by considering the problem in a binary view. The *BitJMSD* function consists of the *BitJaccard* and *BitMSD* functions. the *BitJaccard* function first marks unrecorded rates as unknown, and then, converts the rates more or less than a threshold γ to *one* or *zero*, respectively. Then, it computes the *Jaccard* function (This idea is used in the *Jaccard* function as *Roundthedata*). In the sequel, the *BitMSD* function calculates the similarity value by checking whether the ratings of two users are identical or not.

2.8. Spearmans rank correlation (SRC) and its extensions

The *SRC* function has been introduced in literature (Ahn, 2008; Bagchi, 2015; Sivaramakrishnan et al., 2018) with some differences with the following formula:

$$SRC(u,v) = 1-\frac{6*\sum_{i\in I_u\cap I_v} R_{ui}^2-R_{vi}^2}{|I_u\cap I_v|.\left(|I_u\cap I_v|^2-1\right)} \quad (8)$$

SrankC is an extension of this function used in some studies (Ahn, 2008; Bagchi, 2015; Singh, Setta, & and Rajput, 2019; Kwon, Lee, & and Hong, 2009). In this function, the rates are first sorted, and then, their rank is used in the above formula instead of the rated value. *SC* (Ricci et al., 2010; Levinas, 2014; Bobadilla, Ortega, & Hernando, 2012) is another function in this category that first sorts the rates in descending order and assigns a rank for each rate. Then, it computes the *Pearson* function using the obtained ranks. This function acts well when the size of the dataset is small and the requiblue time for computation is low.

2.9. New heuristic similarity model (NHSM)

The *PIP* (Ahn, 2008) function computes three mathematical factors from the rate of co-rated items for both users to measure the similarity between them. The factors are the distance between two rates, the popularity degree of the ratings (higher rating, more interest) and the difference between them. The obtained values from this function are very large as well as the processing time and complexity of the calculations are also high. This heuristic similarity measure is composed of three factors of similarity including proximity, impact, and popularity, and hence, the measure is named PIP.

$$Simi(u,v) = \sum_{i\in I_u\cap I_v} PIP(R_{ui},R_{vi})$$
$$PIP(R_{ui},R_{vi}) = Proximity(R_{ui},R_{vi})*Impact(R_{ui},R_{vi})*Popularity(R_{ui},R_{vi}) \quad (9)$$

Liu et al. (2014) proposed a heuristic similarity function called *NHSM* that computes similarities based on local information between users and items as well as global information. The *NHSM* function is as famous as the *Cosine* and *Pearson* similarity functions. Liu et al. also introduced the *PSS* function that is similar in idea to the *PIP* function, except for using the nonlinear sigmoid function to penalize and encourage instead of the linear function used by PIP for this purpose. *URPsim* (Liu et al., 2014) is another function introduced in this work. This function supposes that one user may tend to rate high while the other tends to rate low. Furthermore, the *NHSM* function is derived from the combination of *URPsim*, *PSS*, and *Jaccard*. In this function, the obtained values are very small while the computation time and its complexity are also high. The key point is that the function is not exclusively dependent on the co-rated items and the similarity evaluation is done with the global view. The PSS measure is also composed of three factors of similarity including proximity, significance, and singularity, and hence, the measure is named PSS.

$$PSSsimi(u,v) = \sum_{i\in I_u\cap I_v} PSS(R_{ui},R_{vi})$$
$$PSS(R_{ui},R_{vi}) = Proximity(R_{ui},R_{vi})*Significance(R_{ui},R_{vi})*Singularity(R_{ui},R_{vi}) \quad (10)$$

$$NHSM(u,v) = PSSsimi(u,v)*JaccSim(u,v)*URPsim(u,v)$$
$$URPsim\left(u,v\right) = 1-\frac{1}{1+\exp\left(-\left|\overline{R_u}-\overline{R_v}\right|*\left|\sigma_u-\sigma_v\right|\right)} \quad (11)$$

Son (Son, 2016) selected some typical algorithms from three main categories algorithms for recommender systems to compare their performance. In this comparative study, the *NHSM* similarity function was selected as the representative for the collaborative filtering methods. As it was reported in this article, the *NHSM* function yields higher accuracy and lower computational time, while other methods use additional information.

2.10. MultiLevel

Polatidis and Georgiadis have introduced the *multilevel* algorithms (Polatidis & Georgiadis, 2016; Polatidis & Georgiadis, 2017) in the field of the collaborative filtering approach. The basic idea behind these functions is to divide the similarity between two users into multiple levels based on the existing constraints. Then, the similarity is computed at each level using the *Pearson* similarity function. In the first algorithm

(Polatidis & Georgiadis, 2016), the number of levels is fixed, while in the second algorithm, published a year later in 2017, the number of levels is dynamically determined.

$$simi(u,v) = \begin{cases} Pearson(u,v)+x & \text{if } \frac{|I_u \cap I_v|}{T} \geqslant t_1 \text{and} Pearson\left(u,v\right) \geqslant y \\ Pearson(u,v)+x & \text{if } \frac{|I_u \cap I_v|}{T} \leqslant t_1 \text{and} \frac{|I_u \cap I_v|}{T} \geqslant t_2 \text{ and } Pearson\left(u,v\right) \geqslant y \\ Pearson(u,v)+x & \text{if } \frac{|I_u \cap I_v|}{T} \leqslant t_2 \text{and} \frac{|I_u \cap I_v|}{T} \geqslant t_3 \text{ and } Pearson\left(u,v\right) \geqslant y \\ Pearson(u,v)+x & \text{if } \frac{|I_u \cap I_v|}{T} \leqslant t_3 \text{and} \frac{|I_u \cap I_v|}{T} \geqslant t_4 \text{ and } Pearson\left(u,v\right) \geqslant y \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

where t_1, t_2, t_3, t_4 and T is the pre-specified thresholds. In 2018, Alshammari, Kapetanakis, Polatidis, and and Petridis (2018) proposed the *IBCF* similarity function based on the *Triangle* similarity function and the *Multilevel* function. This function uses the *Triangle* function based on the *Multilevel* (Polatidis & Georgiadis, 2016) scheme instead of the *Pearson* function. This function uses the same values of the hyperparameters.

2.11. Significance, default values, and fuzzy set (SDFS)

The *SDFS* similarity function was proposed by Saeed and Mansoori (2017) in 2017. They first introduced a function namely *DFS* using the fuzzy *T −norm* and *S −norm* operators. This function does not necessarily depend on the co-rated items. The final similarity is called *SDFS* and calculated by using the *DFS* function.

$$SimiSDFS\left(u,v\right) = \frac{Min\left(\sum_i \in I_{UV} w_i\left(u,v\right), \gamma\right)}{\gamma} * SimiDFS\left(u,v\right) \tag{13}$$

where I_{UV} is intersection or union of I_u and I_v.}

2.12. Mean, Jaccard, and differences (MJD)

The *Pearson*'s similarity functions and other traditional similarity functions used in recommender system only use the numerical values of rates. The idea of the *MJD* similarity function is to enable extraction of additional information from the ratings such as rate distribution, the number of each rate occurrence, and so on. Thus, the similarity between two users is computed by using both the numerical values and additional non-numerical information. The *MJD* similarity function introduced by Bobadilla et al. (2012) obtained from the linear combination of several similarity functions as:

$$MJD\left(u,v\right) = \frac{1}{6}\left(w_1 v^0\left(u,v\right) + w_2 v^1\left(u,v\right) + w_3 v^3\left(u,v\right) + w_4 v^4\left(u,v\right) + w_5 \mu\left(u,v\right) + w_6 Jaccard\right) \tag{14}$$

whereas v^k is the number of items whose rate difference is k and w_i obtained in the neural network learning process. In addition, the average and standard deviation of difference of co-rated items' rates are calculated after normalization of the rates. Finally, the weight of each segment is obtained using an artificial neural network.

2.13. Singularity measure (SM)

Since the traditional similarity functions in recommender systems calculate the similarity between two users based on the rates recorded for the co-rated items without considering the concept created by the user, their output has usually an error. Suppose two users share a similar rating for an item, but this rating differs from other user's ratings. This fact is strong evidence for the high similarity between two users. However, if the ratings recorded by these two users are the same as those rated by others, there is no evidence to obtain any similarities between them. The *SM* (Bobadilla et al., 2012) function tries to accurately calculate the similarity between users by extracting this kind of hidden information among the ratings for the items. The function divides the ratings for the MovieLens dataset into two categories including the positive and negative ratings. First, it computes the singularity of the high rates (S_P^i) and low rates (S_N^i) for each item. Then, for both user u and v, it assumes A as the set of all items rated high, B as the set of all items rated low, and C as the set of all items rated differently by two users. Finally, the similarity is calculated via the formula:

$$SM(u,v) = \frac{1}{3}\left[\frac{1}{|A|}\sum_{i \in A}\left[1-(R_{ui}-R_{vi})^2\left(S_P^i\right)^2\right] + \frac{1}{|B|}\sum_{i \in B}\left[1-(R_{ui}-R_{vi})^2\left(S_N^i\right)^2\right] + \frac{1}{|C|}\sum_{i \in C}\left[1-(R_{ui}-R_{vi})^2\left(S_P^i * S_N^i\right)\right]\right] \tag{15}$$

In 2020, A nonlinear function (Jin, Zhang, Cai, & Zhang, 2020) was introduced to calculate the similarity whereas the singularity factor was used to weight the similarity. The proposed algorithm not only applies the user's co-rating items information but also takes into account the overall rating data effectively using context information. The singularity factor used in this article (*SimiLocal*) slightly is similar to the *SM* function.

$$SimiLGGfinal(u,v) = SimiLocal(u,v) * SimiGlobal1(u,v) * SimiGlobal2(u,v) \tag{16}$$

$$SimiGlobal1\left(u,v\right) = 1 - \exp\left(-1 * \frac{|I_u \cap I_v|}{|I_u|}\right) \tag{17}$$

$$SimiGlobal2\left(u,v\right) = \sum_{t=1}^{T} \sqrt{\overrightarrow{V_u(t)} * \overrightarrow{V_v(t)}}, \quad T = 5 \tag{18}$$

They defined the user u ratings as a vector $\overrightarrow{V_u} = (f_1, f_2, \ldots, f_T)$, the user v ratings as a vector $\overrightarrow{V_v} = (l_1, l_2, \ldots l_T)$, and f_t, l_t indicate the number of users rated the score as t.

2.14. Discount

There is a belief that when the number of co-rated items is lower than a threshold, the computed similarity is less reliable. Thus, the similarity functions calculate a low similarity for these users (Aggarwal, 2016; McLaughlin & Herlocker, 2004).

$$Discount\left(u,v\right) = Simi\left(u,v\right)*\frac{Min\{|I_u \cap I_v|, \beta\}}{\beta} \tag{19}$$

In these articles, the *Pearson* and *Cosine* similarity functions are used. However, in another article (Symeonidis, Nanopoulos, Papadopoulos, & Manolopoulos, 2006), the same function definition is referblue to as the weighted similarity function, and the *Max* function is used instead of *Min* in the above formula.

$$SAD\left(u,v\right) = \frac{\sum_{i\in I_u\cap I_v} sv_{u,v,i}}{|I_u \cap I_v|},$$
$$sv_{u,v,i} = \begin{cases} \sqrt{\left(R_{ui}-\overline{R_u}\right)*\left(R_{vi}-\overline{R_v}\right)}*\left(1+\beta*s_{i,+}\right) & \text{if}\left(R_{ui}-\overline{R_u}\right)\geqslant 0\text{and}\left(R_{ui}-\overline{R_u}\right)\geqslant 0 \\ \sqrt{\left(R_{ui}-\overline{R_u}\right)*\left(R_{vi}-\overline{R_v}\right)}*\left(1+\beta*s_{i,-}\right) & \text{if}\left(R_{ui}-\overline{R_u}\right) < 0\text{and}\left(R_{ui}-\overline{R_u}\right) < 0 \\ \sqrt{\left(R_{ui}-\overline{R_i}\right)*\left(R_{vi}-\overline{R_v}\right)} & \text{else} \end{cases} \tag{22}$$

2.15. Statistic-based cOllaborative Filtering Algorithm (SOFA)

SOFA (Yao, Yuan, Xie, & Chen, 2013) computes the similarity using statistical information. It first uses the *Pearson* or *Cosine* similarity function to find the similarity between two users. If the number of co-rated items is lower than the threshold, it does not consider the similarity to be reliable and assigns zero to this similarity. Through another way, the function re-measures the similarity by using variance and average of the ratings. Eventually, the ultimate similarity is evaluated by combining these two values.

$$SOFA\left(u,v\right) = \frac{1}{2}\left(Simi\left(u,v\right) + \frac{\overline{R_u}*\overline{R_v}+\sigma_u^2*\sigma_v^2}{\sqrt{\left(\overline{R_u}+\sigma_u^2\right)*\left(\overline{R_v}+\sigma_v^2\right)}}\right) \tag{20}$$

2.16. Proximity, number, and ratio (PNR)

Wu, He, Ren, and Xia (2008) improved the urban block distance (Manhattan or boxcar) and introduced the *PNR* function. They try to provide a function that accurately calculates the similarity based on co-rated items, including the number of co-rated items in calculations, and also has a low computational time. As they reported in their article, the *PNR* function has obtained more accurate results than other functions. The function works based on three factors including proximity and similarity between co-rated items, number of co-rated items, and the ratio of the users' co-rated items to all rated items.

$$PNRsim(u,v) = P(u,v)*F(u,v)*G(u,v) \tag{21}$$

The $P(u,v)$ factor is the similarity of rates of co-rated items calculated by using the *Manhattan* distance. The value of $P(u,v)$ alone does not represent the similarity between two users correctly, because when the number of co-rated items are low, the formula may calculate a large number. The $F(u,v)$ and $G(u,v)$ are used to influence the number of co-rated items and the ratio of co-rated items, respectively.

2.17. Statistical attribute distance (SAD)

The *SAD* similarity function was presented by Weng et al. (2005). This function integrates the statistical information of rates with the calculated similarity. Suppose a particular item *i* is liked by most users. The idea behind this function is that if two users' opinions are negative against that item, those two users are more alike than other pairs of users whose opinion is positive.

2.18. RAtio-based (RA)

In 2017, Wu, Cheng, and Chen (2017) introduced the *RA* similarity function based on the ratio of rates. This function sometimes yields better results than *Pearson*, *Cosine*, and *NRCF*.

$$RA\left(u,v\right) = \frac{\sum_{i\in I_u\cap I_v}\frac{Min(R_{ui},R_{vi})}{Max(R_{ui},R_{vi})}}{|I_u \cap I_v|} \tag{23}$$

2.19. CosineRec

The *CosineRec* (Jeong, Lee, & Cho, 2010) function was introduced in 2012 to improve the accuracy by updating traditional functions. *CosineRec* is the first function based on the repeated message sending. In other words, the function calculation is repeated recursively until its convergence. It should be noted that the function has a long computation time.

$$\begin{aligned} &simi(u,v) = res(u,v) + av(u,v) \\ &res\left(u,v\right) = simi\left(u,v\right) - \mathbf{Max}_{w\in Users,\, w\neq v}\left\{av\left(u,w\right) + simi\left(u,w\right)\right\} \\ &av\left(u,v\right) = \mathbf{Min}\left\{0, res\left(v,v\right) + \sum_{w\in Users,\, w\neq u, w\neq v} \text{Max}\left\{0, res\left(w,v\right)\right\}\right\} \\ &av\left(v,v\right) = \sum_{w\in Users,\, w\neq v} \text{Max}\left\{0, res\left(w,v\right)\right\} \end{aligned} \tag{24}$$

2.20. Conditional probability-based (CPB)

The *CPB* function which was presented by Deshpande and Karypis (2004) uses the conditional probability to compute similarity. It is an asymmetric similarity function that is computed via the formula:

$$CPB\left(u,v\right) = \frac{Freq(uv)}{Freq(v)(Freq(u))^{\alpha}} \tag{25}$$

2.21. Relevant Jaccard mean square distance (RJMSD)

RJMSD (Jin et al., 2020; Bag, Kumar, & Tiwari, 2019) makes full use of all the scoring information to get the relevant neighbors of the user. The proposed similarity calculation model can make pblueictions easily and efficiently via the formula:

$$RJMSD\left(u,v\right) = \left(\frac{1}{1+\left(\frac{1}{|I_u\cap I_v|}\right)+\left(\frac{|\overline{I_u}|}{1+|\overline{I_u}|}\right)+\left(\frac{1}{1+|\overline{I_v}|}\right)}\right)*\left(1-\frac{\sum_{i\in I_u\cap I_v}(R_{ui}-R_{vi})^2}{|I_u\cap I_v|}\right) \quad (26)$$

2.22. Rating frequency based similarity (RFS)

The *RFS* function is presented by ur Rehman, Hussain, and and Hussain (2013). The function calculates the similarity between two users based on their ratings and the number of times the discrepancy occurs.

$$RFS\left(u,v\right) = \frac{\sum_{i\in[1,RNG]}F_i{*}i/RNG}{\sum_{i\in[1,RNG]}i/RNG} \quad (27)$$

2.23. Raw Moment Similarity (RMS)

The *RMS* similarity function was proposed by Kwon and Hong (2011); Kwon and Hong, 2013, this function tries to calculate the similarity in a probable manner using the rate differences.

$$RMS\left(u,v\right) = 1-\frac{1}{r^k}\frac{1}{n}\sum_{i=1}^{n}(|R_{ui}-R_{vi}|)^k = 1-\frac{1}{r^k}\sum_{z=1}^{m}\Pr\left(D=d_z\right)d_z^k \quad (28)$$

2.24. REsonance Similarity (RES)

In 2017, Tan and He presented the *RES* (Tan & He, 2017) similarity function inspiblue by the Physical Resonance Principle. This function consists of three parts. The first part refers to the consistency of two user's ratings, which is measublue by the angle between them. The second part is the distance criterion, which is calculated based on an exponential function. The third part is the *Jaccard* function. The *RES* function is generally used to overcome the existing problems with the *Pearson* and *Cosine* functions.

$$\begin{aligned}
&SimiRES\left(u,v\right)=\frac{ArcTan(RES(u,v))}{0.5\pi}\\
&RES\left(u,v\right)=\sum_{I_u\cap I_v}C\left(u,v,k_1\right)*D\left(u,v,k_2,k_3\right)*J\left(u,v,k_4\right)\\
&C\left(u,v,k_1\right)=\left(\sqrt{0.5+0.5*Cosine(\phi(u)-\phi(v))}\right)^{k_1}\\
&\phi(u)=\begin{cases}\dfrac{\pi}{R_{max}-R_{min}}*\phi^{+} & ,\text{when}\left(R_{ui}-R_{med}\right)*\left(\overline{R_u}-\mu\right)\geqslant 0\\[2ex] \dfrac{\pi}{R_{max}-R_{min}}*\phi^{-} & ,\text{when}\left(R_{ui}-R_{med}\right)*\left(\overline{R_u}-\mu\right)<0\end{cases}\\
&D\left(u,v,k_2,k_3\right)=\begin{cases}\exp\left(0.5*\left(\left|R_{ui}-\overline{R_i}\right|+\left|R_{vi}-\overline{R_i}\right|\right)\right)^{k_2} & ,\text{when}\left(R_{ui}-\overline{R_i}\right)\left(R_{vi}-\overline{R_i}\right)\geqslant 0\\ \exp(-|R_{ui}-R_{vi}|)^{k_3} & ,\text{when}\left(R_{ui}-\overline{R_i}\right)\left(R_{vi}-\overline{R_i}\right)<0\end{cases}
\end{aligned} \quad (29)$$

The function is optimized over the values k_1,k_2,k_3, and k_4 to obtain the optimal values of these variables and minimize the minimum square error (MSE).

2.25. New User Similarity

The *NewUserSimilarity* (Shen & Zhou, 2010; Xiaoping, 2015) function is based on the idea that when two users submit a rating for an item, they have the same opinion and are similar. For this purpose, the following two criteria are calculated, and then, the final similarity value is obtained from the linear or exponential combination of these two criteria.

$$\begin{aligned}
&c\left(u,v\right)=\sum_{i\in I_u\cap I_v}\left(1-\left(R_{max}-R_{min}\right)\right)*|R_{ui}-R_{vi}|\\
&d(u,v)=(|I_u|-|I_u\cap I_v|)+(|I_v|-|I_u\cap I_v|)
\end{aligned} \quad (30)$$

2.26. Bhattacharyya Coefficient Function (BCF)

Because similarity functions have been mostly defined on co-rated items, they do not work well on sparse datasets. Patra, Launonen, Ollikainen, and Nandi (2015) and Patra, Launonen, Ollikainen, and Nandi (2014) proposed the *BCF* similarity function based on the Bhattacharyya Coefficient, which uses all recorded ratings to measure the similarity between two users. The Bhattacharyya criterion is used in different research works such as image processing, signal, and pattern recognition. However, here it measures the similarity between two users by computing two probability distributions via the formula:

$$BCF\left(u,v\right)=Jaccard\left(u,v\right)+\sum_{i\in I_u}\sum_{j\in I_v}BC\left(i,j\right)*loc\left(R_{ui},R_{vj}\right) \quad (31)$$

where *BC* and *loc* calculate the similarity between two users based on global and local information, respectively. The *Jaccard* function was used to increase the effect of the number of co-rated items. The disadvantage of this function is that if two users rate a small set of items or there is no common rated items, it is not able to calculate the similarity. Besides, it cannot overcome the scalability problem and its computation is complex (Saranya & Sadasivam, 2017; Nadine, Cao, & Deng, 2016).

2.27. *ModifiedHeuristic*

The *ModifiedHeuristic* (Saranya & Sadasivam, 2017) function is designed based on the *Jaccard*, *PSS*, and *BC* similarity functions. The main motivation in defining this function was to overcome the low coverage and accuracy of the existing functions due to the sparsity and scalability problems. It therefore uses the *Jaccard* function to exploit the global and local information. On the other hand, *ModifiedHeuristic* uses a modified *BC* function to cover the weakness of the *BC* function in producing *zero* at its output when two users differently rate the same item. This function also considers the divergence in rating by two users in the computations. Additionally, it employs the *PSS* function to incorporate the values of two users' rates into the calculated similarity.

$$ModifiedHeuristic\left(u,v\right) = w_1{*}Jaccard\left(u,v\right) + w_2{*}PSS\left(u,v\right) + \frac{1}{1+\exp(-|BC(u,v)|)} \tag{32}$$

2.28. *Hybrid*

If the similarity between two users depend only on co-rated items, employing symmetric function can be useful. Nevertheless, the items and the rate value of two users are different, and thus, the similarity function is dependent on all the rates submitted by users, and an asymmetric function will work properly. For this purpose, in 2017, *hybrid* (Wang et al., 2017) similarity function was proposed based on the combination of the *kullback* −*leibler* and *PSS* similarity functions. The main aim of designing this function was to use all item ratings.

$$INSC\left(u,v,i\right) = \frac{\sum_{i\in I_u\cap I_v}\left(R_{ui}-\overline{R_u}\right)\left(R_{vi}-\overline{R_v}\right){*}rel\left(u,v,i,j\right)}{\sqrt{\sum_{i\in I_u\cap I_v}\left(R_{ui}-\overline{R_u}\right)^2{*}rel\left(u,v,i,j\right)}{*}\sqrt{\sum_{i\in I_u\cap I_v}\left(R_{vi}-\overline{R_v}\right)^2{*}rel\left(u,v,i,j\right)}} \tag{36}$$

$$hybrid\left(u,v\right) = S_2\left(u,v\right){*}S_3\left(u,v\right){*}\left(\sum_{i\in I_u}\sum_{j\in I_v}S_{item}\left(i,j\right){*}S_1\left(R_{ui},R_{vj}\right)\right) \tag{33}$$

The nonlinear similarity function S_1 is slightly similar to the *PSS* similarity function. The S_{item} function is used to prevent the effect of large amounts of S_1 in different input cases. The S_{item} has a substantial impact on the final similarity value. Thus, it should be defined in a way without exclusive dependence on co-rated items and considering all ratings in its calculations. These desiblue properties are obtained using the *kullback* −*leibler* function. The S_2 and S_3 functions are defined asymmetrically using the number of co-rated items, mean and standard deviation.

In 2018, Deng et al. (2019) proposed another function based on a similar idea with a minor difference. The function calculates the similarity via the formula:

$$SimiKL\left(u,v\right) = \sum_{i\in I_u}\sum_{j\in I_v}S_{item}\left(i,j\right){*}S\left(R_{ui},R_{vj}\right) \tag{34}$$

where S_{item} is calculated from the *kullback* −*leibler* criterion. Its difference with the above hybrid function is in applying the λ criterion. Also, $S(R_{ui},R_{vi})$ is calculated using an exponential function.

proposedSimi is another function introduced by Feng et al. (2018) in 2018 based on a similar idea used in the hybrid function:

$$proposedSimi(u,v) = S_1(u,v){*}S_2(u,v){*}S_3(u,v) \tag{35}$$

where the S_2 and S_3 functions are calculated similar to S_2 and S_3 in the hybrid function with a slight difference, while the S_1 function is calculated via the *CosineUnion* function.

2.29. *ItemWeighted*

In general, there are many implicit assumptions in the proposition of algorithms in the collaborative filtering approach (Bobadilla, Hernando, Ortega, & Gutiérrez, 2012): (1) there is no difference between users, (2) there is no difference between the items in the system, and (3) There is no difference between the ratings that users submitted. Some researchers emphasize on the idea that all items used in the computation of similarity should not have the same weights. Based on this idea, the items with high similarity to the target item should have bigger weight (Bobadilla et al., 2012; Zhang & Andreae, 2008; Choi & Suh, 2013). Several similarity functions have been suggested based on this idea including *WeightedDistance* (Huang & Dai, 2015), *INSC* (Zhang & Andreae, 2008), *PCCEdited*, *CosineEdited*, and *EEdited* (Choi & Suh, 2013). In these functions, first, the similarity between the target item and other items is measublue using a function (usually using the *Pearson* or *Cosine* function). Then, for each target item, the similarity between two users is calculated using an extension of the *Cosine* or *Pearson* functions. These functions generally identify a different set of neighbors for each target item of the active user. It should be noted, however, that computation in these similarity functions is highly time-consuming.

2.30. *Similarity functions based on significance*

Regarding the aforementioned three implicit assumptions of the algorithms in collaborative filtering, Bobadilla et al. (2012) introduced the *significance* function based on a new approach. Using the function, the importance of an item (S_i) is measublue first based on the amount and number of rates it has earned. In addition, the importance of a user (S_u) is measublue by the number of high and low submitted ratings. Then, the value of an item per user (S_{ui}) is calculated based on two obtained values in the previous step, and stoblue in a new matrix. Finally, the *Pearson*, *Cosine*, and *MSD* similarity functions are employed to calculate the similarity between each pair of users using this new matrix.

$$S_i = \left(\frac{1}{|U_i|}\sum_{u\in U_i} R_{ui}\right)*\left(\frac{|U_i|}{|Users|}\right)$$

$$S_u = \left(1-\frac{|D_u|}{|D_u|+|E_u|}\right)*\left(\frac{|D_u|+|E_u|}{|Items|}\right)$$

$$S_{ui} = \begin{cases} S_u{}^*S_i{}^*R_{ui} & ,\text{if} R_{ui}\text{exists} \\ S_u{}^*S_i{}^*\dfrac{\sum_{j\in Nbr_{u,i}} Simi\left(i,j\right){}^*S_j{}^*R_{uj}}{\sum_{j\in Nbr_{u,i}} Simi\left(i,j\right)} & ,\text{if} R_{ui}\text{notexists, but} S_i \text{and} S_u \text{exist} \end{cases} \tag{37}$$

2.31. Accordance, Compromise and Similarity (ACsimi)

In 2015, Pirasteh et al. (2015) presented *ACSimi* which is a weighted similarity function. *ACSimi* was constructed asymmetrically using the traditional similarity functions and the number of items that are not co-rated. In the definition of this function, there are two factors and a similarity function. The first factor, named accordance, measures the impact of each user on their neighbor and vice versa, while the second factor, named compromise, measures the similarity of ratings regardless of the items. The similarity functions such as *Pearson*, *Cosine*, *MSD*, and *Srs* are used in this function.

$$ACSimi\left(u,v\right) = \frac{\overrightarrow{V_u}\cdot\overrightarrow{V_v}}{\left\|\overrightarrow{V_u}\right\|\cdot\left\|\overrightarrow{V_v}\right\|}*\left(1-\exp\left(-\frac{|I_u\cap I_v|}{|I_u|}\right)\right)*Simi\left(u,v\right) \tag{38}$$

where $\overrightarrow{V_u}$ represents a vector based on user u ratings.

2.32. Entropy based

In 2015, Wei, Guangquan, and Jie (2015) presented a similarity function based on the concept of entropy and *Manhattan* distance. In this function, the entropy concept was used to identify neighbors properly while the *Manhattan* distance was used to overcome the long tail problem. The function is calculated as follows:

$$H = -\sum_{i\in I_u\cap I_v} p_i \log_2(p_i),\quad SimiE\left(u,v\right) = 1-\frac{H}{\log_2|I_u\cap I_v|} \tag{39}$$
$$SimEntropy(u,v) = \alpha^*Pearson(u,v)+(1-\alpha)^*SimiE(u,v)$$

In another attempt in 2015, Li and Zheng (2015) presented a similarity function by combining the entropy and Bhattacharyya functions. They believed that each user shows a different behavior in rating. Someone has interested in high ratings and some others prefer to rate low. The *Bhattacharyya* function is a relevant criterion for measuring the overlap between two users' ratings, while entropy is a proper criterion for measuring the rating differences. The function calculates the similarity via the formula:

$$En(u,v) = \exp(-|H(u)-H(v)|)$$
$$Habit\left(u,v\right) = \begin{cases} En(u,v) & ,\text{if} BC(u,v)=0 \\ \sqrt{En(u,v)^*BC(u,v)} & ,\text{if} BC(u,v)\neq 0 \end{cases} \tag{40}$$
$$BCE(u,v) = Habit(u,v)+Pearson(u,v)$$
$$NBCE\left(u,v\right) = -1+\frac{2^*(2-BCE(u,v))}{2-(-1)}$$

On the other hand, the majority of the similarity functions presented for collaborative filtering evaluate each pair of rated items individually. As a result, the global rating behavior of users is neglected. As an example, the discrepancy in ratings of an item by two users causes a great similarity drop down, even if ratings of other items are the same. Regarding this fact, in 2018 and 2019, Lee (2018b) and Lee (2019) introduced a similarity function to model the user rating behavior based on the entropy concept via the formula:

$$PROP\left(u,v\right) = 1-sig\left(\frac{1}{|I_u\cap I_v|}\sum_{i\in I_u\cap I_v}\frac{(R_{ui}-R_{vi})^2}{E(i)}\right) \tag{41}$$

The role of the *sig* function was to produce an output between *zero* and *one*. In 2018, Lee (2018c) also used the entropy measure as the weight in the *Cosine* and *Pearson* similarity functions:

$$E(i) = -\sum_{k\in[R_{min},R_{max}]} prob\left(r_i=k\right)\log_2\left(prob\left(r_i=k\right)\right)$$
$$COSEntropy\left(u,v\right) = \frac{\sum_{i\in I_u\cap I_v} R_{ui}{}^*R_{vi}{}^*E(i)}{\sqrt{\sum_{i\in I_u}(R_{ui})^2}\sqrt{\sum_{i\in I_v}(R_{vi})^2}} \tag{42}$$
$$PCCEntropy\left(u,v\right) = \frac{\sum_{i\in I_u\cap I_v}\left(R_{ui}-\overline{R_u}\right)^*\left(R_{vi}-\overline{R_v}\right)^*E(i)}{\sqrt{\sum_{i\in I_u\cap I_v}\left(R_{ui}-\overline{R_u}\right)^2}^*\sqrt{\sum_{i\in I_u\cap I_v}\left(R_{vi}-\overline{R_v}\right)^2}}$$

The *WeightedDifferenceentropy* function is another similarity function that uses the entropy measure (Kwon et al., 2009). This function calculates the similarity between two users via the formula:

$$SimiWDE\left(u,v\right) = -\sum_{i\in I_u\cap I_v} p(d_i)\log_2(d_i)\times|d_i| \tag{43}$$

Kwon, Lee, Kim, and Hong (2009) presented another function based on entropy to calculate similarity:

$$simEW\left(u,v\right) = Simi\left(u,v\right)^*\frac{1}{1+|H(u)-H(v)|} \tag{44}$$

Furthermore, the functions represented in Piao, Zhao, and Zheng (2009) and Piao, Zhao, and and Feng (2007) calculate the similarity using the entropy measure in a different form as follows:

$$Simi(u,v) = H(u)+H(v)-H(u,v) \tag{45}$$

2.33. Combined functions

Among the algorithms presented for neighborhood-based collaborative filtering, there are many algorithms that were designed through integrating other algorithms (El Alami, Nfaoui, & El Beqqali, 2015). In 2017, Shen, Liu, and Zhang (2017) presented the *ImprovedCosine* similarity function, which is an improved form of the *Cosine* function. This function is presented based on the idea that the *Cosine* function only uses local rating information without considering global information, causing an error in identifying similar users to the target user.

$$\omega\left(u,v\right) = \left(\frac{1}{Jaccard(u,v)}\right)^{\phi},\qquad \phi\in\left[0,\infty\right]^{***} \tag{46}$$
$$ImprovedCosine(u,v) = (Cosine(u,v))^{\omega(u,v)}$$

In 2018, Duong et al. (2018) introduced the *squablueCosine*, *squabluePearson*, *cubedCosine* and *cubedPearson* functions with a slightly

Table 3
Selected Extentions of each similarity function.

Function	Selected Extensions
PCC	WPCC, ShrunckPCC
Cosine	Cosine, WUP$_u$, IPSim
Distance	Quasi
Jaccard	JaccardUOD, ExtendedJacc
MSD	BitMSD
Entropy	CosineEntropy, SimWDE
ItemWeighted	WeightedDistance(Cosine, PCC)
ACSimi	simi = PCC
Combined	weightedPCCJacc
Discount	simi = PCC

different idea. They found that at 97% of cases, the *Cosine* function produces the similarity between two arbitrary items equal to a number in the range $[0.85, 1]$ with a coefficient of variation equal to 0.9%. Such a small coefficient of variation causes an error in identifying similarities between similar and dissimilar items. The value of the coefficient is better for the *Pearson* function, and thus, to overcome the problem and take advantage of both *Cosine* and *Pearson* functions, they combined both functions to calculate similarity.

In 2018, Suryakant and Mahara introduced the *CjacMD* (Suryakant & Mahara, 2016) similarity function. Since the *Cosine* function considers the angle between two rate vectors without taking into account any user behavioral information, its accuracy is low. Thus, by combining the *Cosine*, *MMD* and *Jaccard* functions, each user's taste for voting (some high-rating and some low-rating) is involved in the similarity calculations.

$$CJacMD(u,v) = Cosine(u,v) + Jaccard(u,v) + MMD(u,v) \tag{47}$$

Other extensions of the *Pearson* function includes its combination with the *Jaccard* function (AL-Bakri & Hashim, 2018; Zhang et al., 2017; Saranya et al., 2016) as follows:

$$modifiedSimi(u,v) = modifiedCPCC(u,v)*Jaccard(u,v) \tag{48}$$

$$Simi(u,v) = 2*Jaccard(u,v)*WeightedPCC(u,v) \tag{49}$$

$$Simi(u,v) = w_1*Pearson(u,v) + w_2*Jaccard(u,v) \qquad w_1 = w_2 = 0.5 \tag{50}$$

The *COPCHg* function was developed by Mu et al. (2019) with

Table 4
results for dataset: ML1M and kn neighbours:100.

	function	MAE	RMSE	coverage	precision	recall	F1-measure	Simi = 0
1	*WPCC*	0.7175	0.92411	0.98516	0.82358	0.55612	0.66392	0.081685
2	*ShrunckPCC*	0.71034	0.916	0.99601	0.8225	0.57701	0.67822	0.081685
3	*Discount1(PCC)*	0.7175	0.92411	0.98516	0.82358	0.55612	0.66392	0.081685
4	*Discount2(PCC)*	0.7147	0.92193	0.99748	0.81801	0.58474	0.68198	0.081685
5	*ACSimi(PCC)*	0.70938	0.9146	0.99679	0.82152	0.57942	0.67956	0.081685
6	*Quasi*	0.7239	0.93344	0.9984	0.81156	0.57268	0.67151	0.1113
7	*SimiSDFS*	0.73266	0.94141	0.98102	0.81931	0.51354	0.63135	0.078838

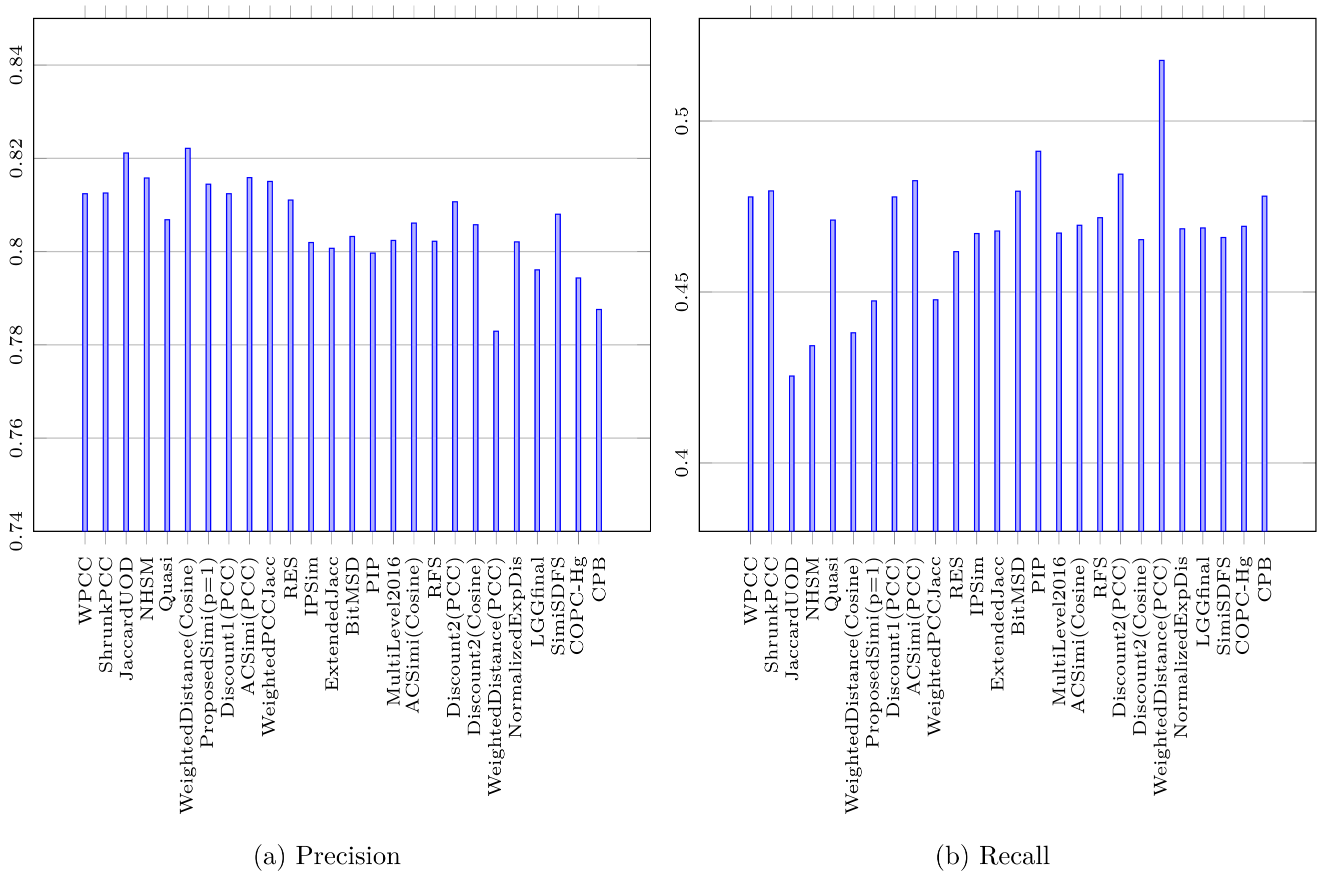

Fig. 1. Precision and Recall of simi functions.

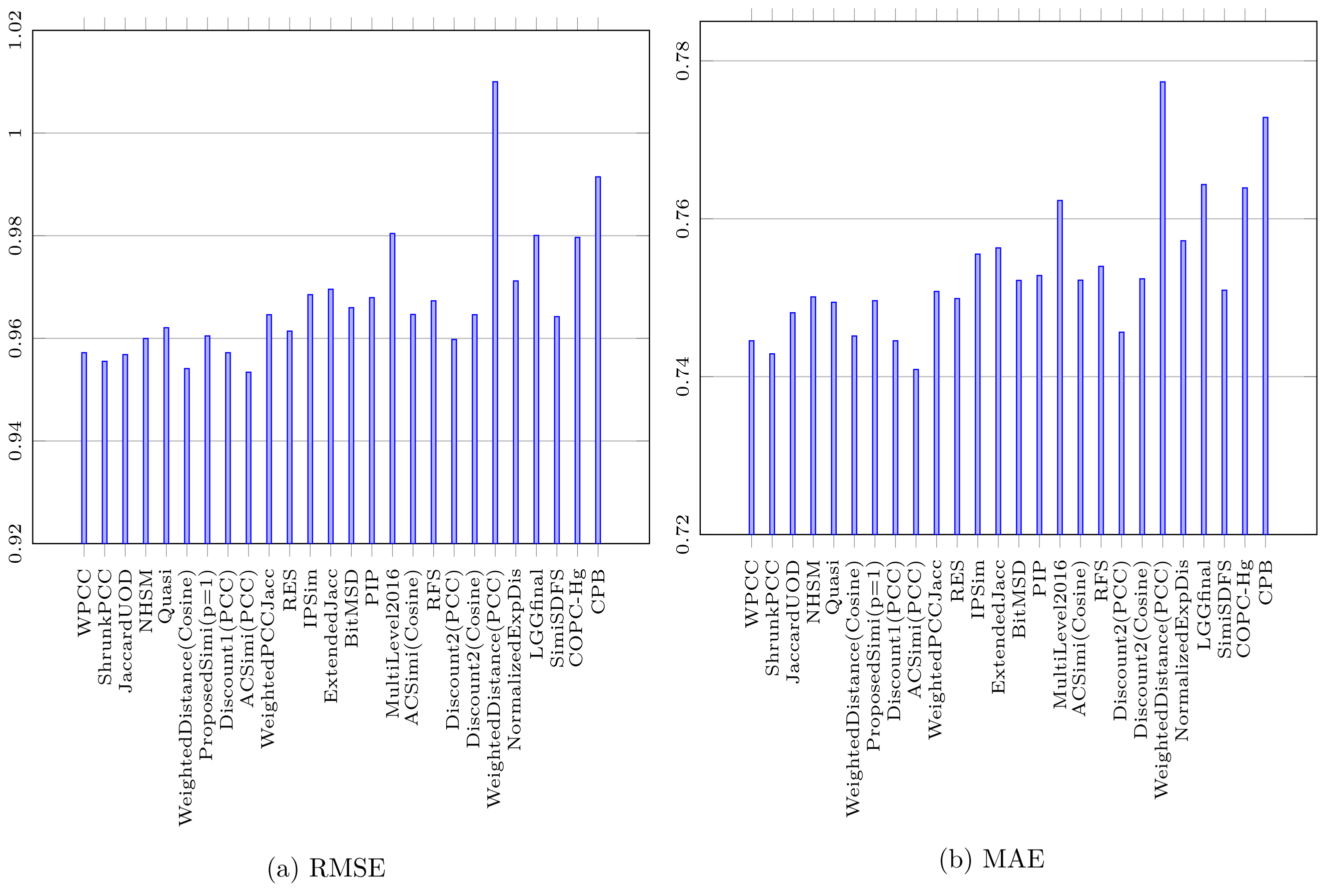

Fig. 2. RMSE and MAE of simi functions.

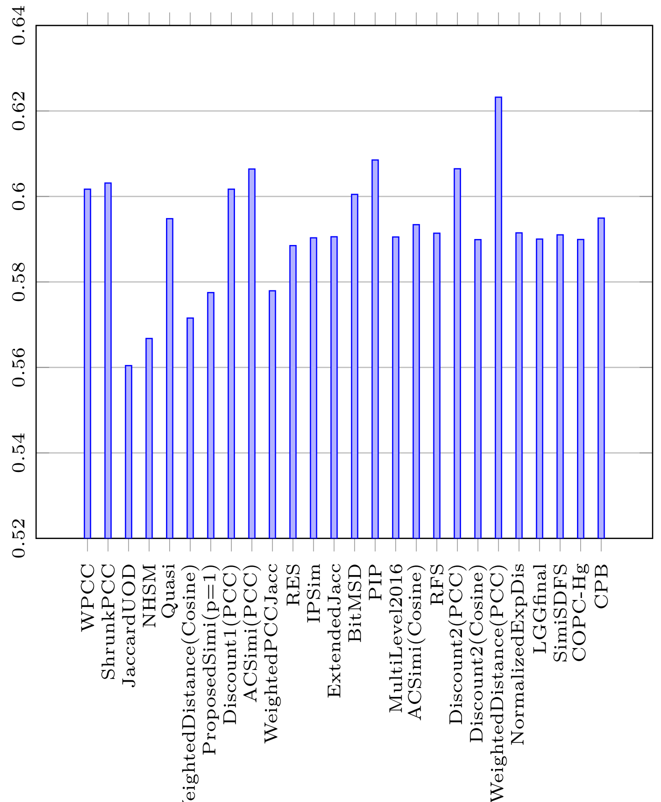

Fig. 3. F1-measure of simi functions.

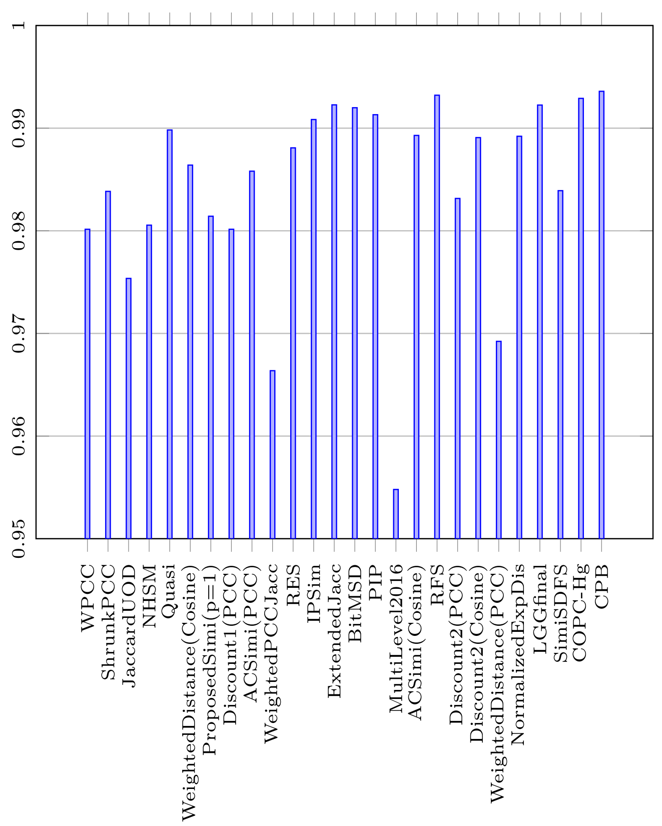

Fig. 4. Coverage of simi functions.

combining the *Jaccard*, *COPC* and *HgDistance* functions. The *COPC* function was also introduced by Mu et al. in the same article. However, it is described in the *pearson* subsection because it is an improvement of the *Pearson* function.

$$COPC - HG(u, v) = \alpha^* COPC(, v) + (1-\alpha)^*(Hg(u, v) + Jacc(u, v)) \quad (51)$$

There are many users whose rating preference behavior is different from normal users. They tend to rate items according to their behavior. Some users generally rate the items in low ranges regardless of their goodness or badness while some others mostly rate the items in high ranges. These diverse form of rating are called as rating preference behavior (*RPB*). To handle such behaviors, Ayub et al. (2019) proposed the IPWR similarity measure using the standard deviation (*SD*) of each user via the following formula. The *simiIPWR* similarity measure considers both *RPB* and Sim_{IPCC} by combining both factors using an adaptive weighting scheme.

$$simiIPWR(u, v) = \alpha^* RPB(u, v) + \beta^* SimiIPCC(u, v), \quad (52)$$

$$RPB\left(u, v\right) = Cosine\left(|\overline{R_u} - \overline{R_v}|^*|SD(u) - SD(v)|\right), SD(u) = \sqrt{\frac{\sum_{i\in I_u}\left(R_{ui} - \overline{R_u}\right)}{|I_u|}} \quad (53)$$

3. Experiments and Evaluations

Since the primary neighborhood-based algorithms have been designed as user-based (Kluver et al., 2018), all the reviewed algorithms in this survey were implemented based on the user-based approach. To implement a neighborhood-based algorithm, it is necessary to specify the aggregation function, the number of neighbors, datasets, and evaluation metrics. In this section, the experimental setup and the obtained results by each algorithm are presented.

3.1. The aggregation function and number of neighbors

To pblueict the rating of an item by an active user, the following formula (Ahn, 2008; Aggarwal, 2016) was used:

$$\widehat{R}_{ui} = \overline{R_u} + \frac{\sum_{v\in Nbr_{u,i}} simi\left(u, v\right)^* \left(R_{vi} - \overline{R_v}\right)}{\sum_{Nbr_{u,i}} \left|simi\left(u, v\right)\right|} \quad (54)$$

where *Nbr* indicates neighbors of the most similar active user. To find the number of neighbors in the presented algorithms, the Elbow method (Thorndike, 1953) is mostly used to examine different numbers in an appropriate range and estimate the optimal number of neighbors. In this study, the approximate number of neighbors for each algorithm was obtained from its related article. In most of these studies, approximately, a number of 50 neighbors used on the ML100k dataset. Accordingly, in the experiments conducted in this study, a number of 50 users was consideblue as neighbors of a user to evaluate and compare the similarity functions.

3.2. Evaluation Metrics

The accuracy metrics for evaluation of recommender systems can be categorized into three classes including pblueictive accuracy metrics, classification accuracy metrics, and rank accuracy metrics (Zhang et al., 2016; Herlocker, Konstan, Terveen, & Riedl, 2004). Pblueictive accuracy metrics are generally used to compare the quantity of similarity between pblueicted ratings and real ratings. Mean Absolute Error (MAE) and Root Mean Squablue Error (RMSE) are two common metrics that are mostly used in the evaluation of recommender systems. Classification accuracy metrics have been adopted from the information retrieval research and consist of precision, recall, F1-measure, and some other related metrics. These metrics are used to calculate the fraction of pblueiction, and the quality of recommendations and search results. Unlike pblueictive accuracy metrics and classification accuracy metrics in directly measuring the quality of the pblueicted items, the focus of rank accuracy metrics is on the ordering quality of the recommended items. Some commonly used rank accuracy metrics are the half-life utility metric, the Pearson product-moment correlation coefficient, and the Normalized Distance-based Performance Measure (NDPM). Regarding these three categories of evaluation metrics, none of the cited articles here use the rank accuracy metrics and they employ one of pblueictive accuracy metrics or classification accuracy metrics for evaluation purposes.

Since one of the purposes of this study was the evaluation of the similarity functions under the same fair and standard condition, the similarity functions investigated hereby are evaluated using the metrics used in both categories. The MAE and RMSE evaluation metrics (Peng et al., 2017) are used with the precision, recall and F1-measure metrics. The precision, recall, and F1-measure metrics require an Interested-Value, which is assigned by different values in different articles. In this article, the precision-recall definition used by Yao et al. (2013) was utilized while Interested-Value was assigned three. In addition, the coverage and the number of zero values calculated by the similarity function are measublue. Some authors believe that zero values calculated by the similarity functions affect the accuracy because the function is not able to get the actual similarities between users (Sheugh & Alizadeh, 2015; Ahn, 2008). To simplify the calculation of the coverage metric (Aggarwal, 2016), it was consideblue as the percentage of the test data that their value was estimated by the proposed model. The coverage along with the MAE metric provides a proper evaluation of the algorithms. The preference of an algorithm is indicated by a lower value of MAE, RMSE, and sim0 as well as a higher value of coverage, precision, recall, and F1-measure.

3.3. Dataset

Regarding the studies on collaborative filtering-based methods, the MovieLens (ML) dataset is the most popular set for performance investigation. Accordingly, the ML100k and ML1M datasets[1] were to evaluate similarity functions. The ratings in this dataset are integer values ranging from 1 to 5. The dataset was divided into two groups including training set (80%) and testing set (20%). In this study, the similarity functions were utilized on the ML100K dataset consisting of 100000 rates recorded by 943 users for 1682 movies having a sparsity of 93.7%. The 5-fold cross-validation technique was used over the ML100k dataset to calculate the average of the criteria shown in Table 2. The functions obtained high performance on the ML100k dataset were chosen to be examined additionally on the ML1M dataset. The ML1M dataset contains 1000209 ratings, 6040 users, and 3952 Movies with a sparsity of 95.8%. The results of applying these algorithms on the ML1M dataset are shown in Table 4.

4. Discussion

Generally, recommender systems face three major challenges including sparsity, scalability, and cold start. Many similarity functions have been developed to deal with these challenges and blueuce the rate of errors caused by them in the recommender systems. In this review, the neighborhood-based collaborative filtering similarity functions have been collected and their performance has been investigated under equal conditions based on the evaluation metrics. However, comparing the ability of these functions to overcome the abovementioned three challenges in recommender systems is not consideblue in this study and it may be the subject of another article.

[1] http://grouplens.org/datasets/movielens/

It is clear from the initial results that the coverage of the $IBCF, IPIJ, SimEW(Euclidean), PROP1, EEdited$ functions is very low (less than 40%), and comparing their performance with those of other similarity functions is not fair. Table 3 shows the selected extensions of each similarity function family which obtained a comparable performance with other functions. For example, among different extensions of the *Pearson* function, *WPCC* and *ShrunkPCC* obtained the highest performance and selected for further comparison. Besides, the MAE, RMSE, and coverage results of these two functions are significantly different from other extensions of the *Pearson* function. It should be noted, however, that both functions obtained the highest values in terms of precision, recall, and F1-measure within the Pearson extensions. These functions have been proposed regarding that for the number of co-rated items less than a threshold, the similarity accuracy will be low, and thus, it is necessary to decrease their effects. This is while the attempts by similarity functions to model the long-tail property in similarity computation have failed in improving accuracy.

Figs. 1–4 comparatively show the scores obtained by the investigated functions. Performance of these functions is higher than all other functions discussed in this survey. It can be seen from Fig. 2,
$WPCC, ShrunckPCC, JaccardUOD, NHSM, Quasi, WeightedDistance$

$$(Cosine), ProposedSimi(\rho = 1), Discount(PCC),$$

$$ACSimi(PCC), WeightedPCCJacc, RES$$

and *SimiSDFS* are the best in pblueictive accuracy metrics in terms of MAE and RMSE (the lower, the better), where they obtained values less than 0.75 and 0.96, respectively. These threshold values were chosen regarding that the best obtained results for MAE and RMSE were 0.74 and 0.955, respectively. These functions obtained lower MAE and RMSE errors in the estimation of ratings. Regarding the ability of these functions in classification accuracy metrics,
$WPCC, ShrunckPCC, IPSim, ExtendedJacc,$

$$BitMSD, PIP, MultiLevel2016, Quasi, WeightedDistance(PCC), CPB,$$

$$ACSim(PCC), ACSim(Cosine), RFS, Discount, LGGfinal, NormalizedExpDis,$$

$$COPC - Hg$$

, and *SimiSDFS* yielded favorable performance in terms of F1-measure. Regarding that, the precision and recall metrics are not informative enough for evaluation of the functions, the F1-measure was used to combine them and calculate a more informative metric. The F1-measure of these functions were greater than 0.59 (The higher values obtained in the evaluation). In overall, investigation of the functions to determine their capability in both points of view revealed that the $WPCC, ShrunckPCC, Discount(PCC), ACSimi(PCC), Quasi$ and *SimiSDFS* functions obtain the best performance.

An important finding in these comparisons is that the *ItemWeighted* functions, which require a great deal of time to perform calculations, could not achieve a rank in the set of final functions. Besides, the *entropy*-based functions have not also yielded significant results (Have higher MAE error and lower F1-measure). These are while simple functions having low complexity and reasonable computation time yield better results. As an example, the $IP - sim$ function is a simple function for calculating similarity which produces highly accurate results in our evaluation.

The set of selected similarity functions was further examined on the ML1M dataset. Table 4 represents the results obtained by different functions on this dataset. In this evaluation, the number of optimal neighbors for each similarity function were determined and their intersection was consideblue. Additionally, the optimal number of neighbors was defined 100 in this experiment. The outcome of this investigation indicates that the examined functions do not have a particular superiority to each other. This observation is due to a little difference in the values of the evaluation metrics. However, regardless of this point, $ACSimi(PCC), Discount(PCC)$, and *ShrunckPCC* obtained the best performance among these functions with an epsilon difference. In this view, this difference is so small, and thus, these similarity functions (except *SimiSDFS*) have approximately high accuracy. In other words, these functions obtained more accurate results on the MovieLens datasets than other functions.

5. Conclusions and Future work

In this paper, the main effort was to provide a comprehensive review of the neighborhood-based similarity functions with a rating oriented perspective. The investigated functions in this study were developed and examined on the ML dataset to provide fairly evaluation and comparison of their behavior. After investigation of the functions based on the precision, recall, coverage, F1-measure, RMSE, and MAE metrics, the results of experiments indicated that $WPCC, ShrunckPCC, Discount(PCC), ACSimi(PCC)$, and *Quasi* show remarkable performance and outperform other similarity functions. It is expected that the results presented in this paper to be a complete reference for the future studies in analysis and evaluation of novel similarity functions.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering, 17*(6), 734–749.

Agarwal, A., & Chauhan, M. (2017). Similarity measures used in rcommender systems: A study. *International Journal of Engineering Technology Science and Research, 4*(6), 619–626.

Aggarwal, C. C. (2016). *Recommender systems: The textbook*. Springer International Publishing.

Aghdam, M. H., Analoui, M., & Kabiri, P. (2015). A novel non-negative matrix factorization method for recommender systems. *Applied Mathematics And Information Sciences, 9*(5), 2721–2732.

Aghdam, M. H., Analoui, M., & Kabiri, P. (2016). Collaborative filtering using non-negative matrix factorization. *Journal of Information Science, 43*, 567–579.

Aghdam, M. H., Analoui, M., & Kabiri, P. (2016). Modelling trust network using resistive circuits for trust-aware recommender system. *Journal of Information Science, 43*, 135–144.

Ahmadian, S. (2017). A social recommendation system based on reliable virtual ratings (Ph.D. thesis). University of Zanjan.

Ahn, H. J. (2008). new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences, 178*(1), 37–51.

AL-Bakri, N. F. and Hashim, S. H. (2018). A modified similarity measure for improving accuracy of user-based collaborative filtering. Iraqi Journal of Science 59(2B): 934–945.

Al-bashiri, H., Abdulgabber, M. A., Romli, A., & Kahtan, H. (2018). An improved memory-based collaborative filtering method based on the topsis technique. *PLOS ONE, 13*(10), 1–26.

Al-Shamri, M. Y. H. (2014). Power coefficient as a similarity measure for memory-based collaborative recommender systems. *Expert Systems with Applications, 41*(13), 5680–5688.

Alshammari, G., Kapetanakis, S., Polatidis, N., and Petridis, M. (2018). A triangle multi-level item-based collaborative filtering method that improves recommendations. In 19th International Conference, EANN 2018, Bristol, UK.

Arsan, T., Koksal, E., & Bozkus, Z. (2016). comparison of collaborative filtering algorithm with varous similarity measures for movie recommendeation. *International Journal of Computer Science, Engineering and Applications, 6*(3).

Ayub, M., Ghazanfar, M. A., Maqsood, M., & Saleem, A. (2018). A jaccard base similarity measure to improve performance of cf based recommender systems. In *2018 International Conference on Information Networking (ICOIN)* (pp. 1–6).

Ayub, M., Ghazanfar, M. A., Mehmood, Z., Saba, T., Alharbey, R., Munshi, A. M., & Alrige, M. A. (2019). Modeling user rating preference behavior to improve the performance of the collaborative filtering based recommender systems. *PloS one, 14*(8).

Bag, S., Kumar, S. K., & Tiwari, M. K. (2019). An efficient recommendation generation using relevant jaccard similarity. *Information Sciences, 483*, 53–64.

Bagchi, S. (2015). Performance and quality assessment of similarity measures in collaborative filtering using mahout. Procedia Computer Science, 50:229 – 234. Big Data, Cloud and Computing Challenges.

Baxla, M. A. (2014). *Comparative study of similarity measures for item based top n recommendation. Master's thesis, A thesis submitted in partial requirements for the degree of Bachelor of Technology in Computer Science and Engineering.* Rourkela-769008, Orissa: National Institute of Technology Rourkela.

Bobadilla, J., Hernando, A., Ortega, F., & Gutiérrez, A. (2012). Collaborative filtering based on significances. *Information Science, 185*(1), 1–17.

Bobadilla, J., Ortega, F., & Hernando, A. (2012). A collaborative filtering similarity measure based on singularities. *Information Processing & Management, 48*(2), 204–217.

Bobadilla, J., Ortega, F., Hernando, A., & Bernal, J. (2012). A collaborative filtering approach to mitigate the new user cold start problem. *Knowledge-Based Systems, 26*, 225–238.

Bobadilla, J., Ortega, F., Hernando, A., & de Rivera, G. G. (2013). A similarity metric designed to speed up, using hardware, the recommender systems k-nearest neighbors algorithm. *Knowledge-Based Systems, 51*, 27–34.

Bobadilla, J., Ortega, F., Hernando, A., & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems, 46*, 109–132.

Bobadilla, J., Ortega, F., Hernando, A., & Arroyo, Ángel (2012). A balanced memory-based collaborative filtering similarity measure. *International Journal of Intelligent Systems, 27*(10), 939–946.

Bobadilla, J., Serradilla, F., & Bernal, J. (2010). A new collaborative filtering metric that improves the behavior of recommender systems. *Knowledge-Based Systems, 23*(6), 520–528.

Boutet, A., Moor, F. D., Frey, D., Guerraoui, R., Kermarrec, A., & Rault, A. (2018). Collaborative filtering under a sybil attack: Similarity metrics do matter!. In *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)* (pp. 466–477).

Breese, J. S., Heckerman, D., & Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. *Technical Report*. MSR-TR-98-12.

Cacheda, F., Carneiro, V., Fernández, D., & Formoso, V. (2011). Comparison of collaborative filtering algorithms: Limitations of current techniques and proposals for scalable, high-performance recommender systems. *ACM Transacations on the Web, 5(1):2:1–2:33*.

Candillier, L., Meyer, F., & Fessant, F. (2008). Designing specific weighted similarity measures to improve collaborative filtering systems. In *Industrial Conference on Data Mining* (pp. 242–255). Springer.

Chen, R., Hua, Q., Chang, Y.-S., Wang, B., Zhang, L., & Kong, X. (2018). A survey of collaborative filtering-based recommender systems: From traditional methods to hybrid methods based on social networks. *IEEE Access, 6*, 64301–64320.

Choi, K., & Suh, Y. (2013). A new similarity function for selecting neighbors for each target item in collaborative filtering. *Knowledge-Based Systems, 37*, 146–153.

Deng, J., Wang, Y., Guo, J., Deng, Y., Gao, J., & Park, Y. (2019). A similarity measure based on kullback-leibler divergence for collaborative filtering in sparse data. *Journal of Information Science, 45*(5), 656–675.

Deshpande, M., & Karypis, G. (2004). Item-based top-n recommendation algorithms. *ACM Transactions on the Web, 22*(1), 143–177.

Duong, T. N., Than, V. D., Tran, T. H., Dang, Q. H., Nguyen, D. M., & Pham, H. M. (2018). An effective similarity measure for neighborhood-based collaborative filtering. In *2018 5th NAFOSTED Conference on Information and Computer Science (NICS)* (pp. 250–254).

El Alami, Y. E. M., Nfaoui, E. H., & El Beqqali, O. (2015). Improving neighborhood-based collaborative filtering by a heuristic approach and an adjusted similarity measure. *BDCA*, 16–22.

Feng, J., Fengs, X., Zhang, N., & Peng, J. (2018). An improved collaborative filtering method based on similarity. *PLOS ONE, 13*(9), 1–18.

Guo, S. (2014). *Analysis and evaluation of similarity metrics in collaborative filtering recommender system* (Master's thesis). TORNIO: Bachelor's thesis of the Degree Programme in Business Information Technology, Bachelor of Business Administration.

Hassanieh, L. A., Jaoudeh, C. A., Abdo, J. B., & Demerjian, J. (2018). Similarity measures for collaborative filtering recommender systems. In *2018 IEEE Middle East and North Africa Communications Conference (MENACOMM)* (pp. 1–5).

Herlocker, J., Konstan, J. A., & Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval, 5*(4), 287–310.

Herlocker, J., Webster, J., Jung, S., Dragunov, A., Holt, T., Culter, T., & Haerer, S. (2002). A framework for collaborative information environments and unified access to distributed digital content. In *Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries* (p. 378).

Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. (1999). 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, pages 230–237, New York, NY, USA. ACM.

Herlocker, J. L., Konstan, J. A., Terveen, L. G., & Riedl, J. T. (2004). Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems, 22*(1), 5–53.

Hofmann, T. (2004). Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems, 22*(1), 89–115.

Huang, B. and Dai, B. (2015). A weighted distance similarity model to improve the accuracy of collaborative recommender system. In 2015 16th IEEE International Conference on Mobile Data Management, volume 2, pages 104–109.

Huang, H., Yu, G., & Wang, X. (2018). Collaborative filtering algorithm based on rating difference and user interest. In *2018 IEEE 3rd International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)* (pp. 25–31).

Irish, J. D. (2010). The mean measure of divergence: Its utility in model-free and model-bound analyses relative to the mahalanobis d2 distance for nonmetric traits. *American Journal of Human Biology, 22*(3), 378–395.

Isinkaye, F., Folajimi, Y., & Ojokoh, B. (2015). Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal, 16*(3), 261–273.

Jannach, D., Zanker, M., Felfernig, A., & Friedrich, G. (2010). *Recommender systems an introduction*. Cambridge University Press.

Jeong, B., Lee, J., & Cho, H. (2010). Improving memory-based collaborative filtering via similarity updating and prediction modulation. *Information Sciences, 180*(5), 602–612.

Jiang, S., Fang, S.-C., An, Q., & Lavery, J. E. (2019). A sub-one quasi-norm-based similarity measure for collaborative filtering in recommender systems. *Information Sciences, 487*, 142–155.

Jin, Q., Zhang, Y., Cai, W., & Zhang, Y. (2020). A new similarity computing model of collaborative filtering. *IEEE Access, 8*, 17594–17604.

Katpara, H., & Vaghela, V. B. (2016). Similarity measures for collaborative filtering to alleviate the new user cold start problem. *International Journal of Research and Scientific Innovation-IJRSI, 4*(1), 233–238.

Kim, S., Kim, H., & Min, J.-K. (2019). An efficient parallel similarity matrix construction on mapreduce for collaborative filtering. *The Journal of Supercomputing, 75*(1), 123–141.

Kluver, D., Ekstrand, M. D., & Konstan, J. A. (2018). Rating-based collaborative filtering: algorithms and evaluation. In *Social Information Access* (pp. 344–390). Springer.

Knees, P., Schnitzer, D., and Flexer, A. (2014). Improving neighborhood-based collaborative filtering by reducing hubness. In Proceedings of International Conference on Multimedia Retrieval, ICMR '14, pages 161:161–161:168, New York, NY, USA. ACM.

Kwon, H.-J., & Hong, K.-S. (2011). Personalized smart tv program recommender based on collaborative filtering and a novel similarity method. *IEEE Transactions on Consumer Electronics, 57*(3).

Kwon, H.-J., & Hong, K. S. (2013). Novel neighbor selection method to improve data sparsity problem in collaborative filtering. *International Journal of Distributed Sensor Networks, 9*(8), Article 847965.

Kwon, H.-J., Lee, T.-H., and Hong, K.-S. (2009). Improved memory-based collaborative filtering using entropy-based similarity measures. In Proceedings of the 2009 International Symposium on Web Information Systems and Applications (WISA'09), pages 029–034.

Kwon, H. J., Lee, T. H., Kim, J. H., & Hong, K. S. (2009). Improving prediction accuracy using entropy weighting in collaborative filtering. In *2009 Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing* (pp. 40–45).

Lathia, N., Hailes, S., and Capra, L. (2007). Private distributed collaborative filtering using estimated concordance measures. In Proceedings of the 2007 ACM Conference on Recommender Systems, RecSys '07, pages 1–8, New York, NY, USA. ACM.

Lee, S. (2018a). Collaborative filtering using fuzzy rank-based similarity measures. In *2018 International Conference on Control* (pp. 84–89). Artificial Intelligence, Robotics Optimization (ICCAIRO).

Lee, S. (2018b). An entropy-based similarity measure for collaborative filtering. In Y. Tan, Y. Shi, & Q. Tang (Eds.), *Data Mining and Big Data* (pp. 129–137). Cham: Springer International Publishing.

Lee, S. (2018). Entropy-weighted similarity measures for collaborative recommender systems. volume 1982.

Lee, S. (2019). Using entropy for similarity measures in collaborative filtering. *Journal of Ambient Intelligence and Humanized Computing*.

Lee, T. Q., Park, Y., and Park, Y. -T. (2007). A similarity measure for collaborative filtering with implicit feedback. In Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence, pages 385–397, Berlin, Heidelberg. Springer, Berlin Heidelberg.

Leskovec, J., Rajaraman, A., & Ullman, J. D. (2014). *Mining of massive datasets* (2nd ed.). New York, NY, USA: Cambridge University Press.

Levinas, C. A. (2014). An analysis of memory based collaborative filtering recommender systems with improvement proposals. Master's thesis.

Li, M. and Zheng, K. (2015). A collaborative filtering algorithm combined with user habits for rating. In International Conference on Logistics Engineering, Management and Computer Science(LEMCS 2015).

Liang, S., Ma, L., & Yuan, F. (2015). A singularity-based user similarity measure for recommender systems. *International journal of Innovative Computing, Information and Control, 11*(5), 1629–1638.

Liu, H., Hu, Z., Mian, A., Tian, H., & Zhu, X. (2014). A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems, 56*, 156–166.

Liu, M., Pan, W., Liu, M., Chen, Y., Peng, X., and Ming, Z. (2017). Mixed similarity learning for recommendation with implicit feedback. Know-Based System 119(C): 178–185.

Liu, N. N. and Yang, Q. (2008). Eigenrank: A ranking-oriented approach to collaborative filtering. In Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '08, pages 83–90, New York, NY, USA. ACM.

Luo, X., Xia, Y., Zhu, Q., & Li, Y. (2013). Boosting the k-nearest-neighborhood based incremental collaborative filtering. *Knowledge-Based System, 53*, 90–99.

Marinho, L., Hotho, A., Jäschke, R., Nanopoulos, A., Rendle, S., Schmidt-Thieme, L., Stumme, G., and Symeonidis, P. (2012). Recommender Systems for Social Tagging Systems. SpringerBriefs in Electrical and Computer Engineering. Springer New York.

McLaughlin, M. R. and Herlocker, J. L. (2004). A collaborative filtering algorithm and evaluation metric that accurately model the user experience. In Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '04, pages 329–336, New York, NY, USA. ACM.

McNee, S. M., Riedl, J., & Konstan, J. A. (2006). Being accurate is not enough: how accuracy metrics have hurt recommender systems. In G. M. Olson, & R. Jeffries (Eds.), *Extended Abstracts Proceedings of the 2006 Conference on Human Factors in Computing Systems, CHI 2006* (pp. 1097–1101). ACM.

Moghadam, P. H., Heidari, V., Moeini, A., & Kamandi, A. (2019). An exponential similarity measure for collaborative filtering. *SN Applied Sciences, 1*(10), 1172.

Mu, Y., Xiao, N., Tang, R., Luo, L., and Yin, X. (2019). An efficient similarity measure for collaborative filtering. Procedia Computer Science, 147:416–421. 2018 International Conference on Identification, Information and Knowledge in the Internet of Things.

Nadine, U., Cao, H., & Deng, J. (2016). Competitive recommendation algortithm for e-commerce. In *Internationa Conference on Natural Computation, Fuzzy Sysytems and Knowledge Discovery ICNC-FSKD, IEEE*.

Niu, K., Zhao, X., Li, F., Li, N., Peng, X., & Chen, W. (2019). Utsp: User-based two-step recommendation with popularity normalization towards diversity and novelty. *IEEE Access, 7*, 145426–145434.

Patra, B. K., Launonen, R., Ollikainen, V., & Nandi, S. (2014). *Exploiting bhattacharyya similarity measure to diminish user cold-start problem in sparse data* (pp. 252–263). Discovery Science, Springer International Publishing.

Patra, B. K., Launonen, R., Ollikainen, V., & Nandi, S. (2015). a new similarity measure using bhattacharryya coefficient for collaborative filtering in sparse data. *Knowledge-Based System, 82*, 163–177.

Peng, M., Zeng, G., Sun, Z., Huang, J., Wang, H., & Tian, G. (2017). Personalized app recommendation based on app permissions. *World Wide Web, 21*, 1–16.

Piao, C., Zhao, J., and Feng, J. (2007). Research on entropy-based collaborative filtering algorithm. In IEEE International Conference on e-Business Engineering (ICEBE'07), pages 213–220.

Piao, C.-H., Zhao, J., & Zheng, L.-J. (2009). Research on entropy-based collaborative filtering algorithm and personalized recommendation in e-commerce. *Service Oriented Computing and Applications, 3*(2), 147–157.

Pirasteh, P., Hwang, D., & Jung, J. E. (2015). Weighted similarity schemes for high scalability in user-based collaborative filtering. *Mobile Networks and Applications, Springer Nature, 20*(4), 497–507.

Polatidis, N., & Georgiadis, C. K. (2016). A multi-level collaborative filtering method that improves recommendations. *Expert Systems Application, 48*(C), 100–110.

Polatidis, N., & Georgiadis, C. K. (2017). A dynamic multi-level collaborative filtering method for improved recommendations. *Computer Standards and Interfaces, 51*, 14–21.

Pérez-Fernández, R., Sader, M., & Baets, B. D. (2018). Joint consensus evaluation of multiple objects on an ordinal scale: An approach driven by monotonicity. *Information Fusion, 42*, 64–74.

Reddy, Y. S., & Govindarajulu, P. (2017). A survey on data mining and machine learning techniques for internet voting and product/service selection. *International Journal of Computer Science and Network Security (IJCSNS), 17*(9), 261–273.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., and Riedl, J. (1994). Grouplens: An open architecture for collaborative filtering of netnews. In Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work, CSCW '94, pages 175–186, New York, NY, USA. ACM.

Ricci, F., Rokach, L., Shapira, B., & Kantor, P. B. (2010). *Recommender Systems Handbook* (1st ed.). New York, NY, USA: Springer-Verlag New York Inc.

Rupasingha, R. A., & Paik, I. (2019). Alleviating sparsity by specificity-aware ontology-based clustering for improving web service recommendation. *IEEJ Transactions on Electrical and Electronic Engineering, 14*(10), 1507–1517.

Saeed, M., & Mansoori, E. G. (2017). A novel fuzzy-based similarity measure for collaborative filtering to alleviate the sparsity problem. *Iranian Journal of Fuzzy Systems, 14*(5), 1–18.

Saeed, M. (2017). Alleviating the sparsity problem in recommender systems. PhD thesis, Shiraz University (Iran).

Saranya, K., & Sadasivam, G. S. (2017). Modified heuristic similarity measure for personalization using collaborative filtering technique. *Applied Mathematics, 11*(1), 307–315.

Saranya, K. G., Sadasivam, G. S., & Chandralekha, M. (2016). Performance comparison of different similarity measures for collaborative filtering technique. *Indian Journal of Science and Technology, 9*(29).

Sarwar, B., Karypis, G., Konstan, J., and Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. In Proceedings of the 10th International Conference on World Wide Web, WWW '01, pages 285–295, New York, NY, USA. ACM.

Schwarz, M., Lobur, M., & Stekh, Y. (2017). Analysis of the effectiveness of similarity measures for recommender systems. In *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)* (pp. 275–277).

Shams, B. (2018). Network-oriented approach to neighbor-based collaborative ranking (Ph.D. thesis). University of Tehran, Tehran, Iran.

Shardanand, U. (1994). Social information filtering for music recommendation (Master's thesis). In *Massachusetts Institute of Technology Dept. of Electrical Engineering and Computer Science*. Massachusetts Institute of Technology.

Shardanand, U. and Maes, P. (1995). Social information filtering: Algorithms for automating 'word of mouth'. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '95, pages 210–217, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

Shen, K., Liu, Y., & Zhang, Z. (2017). Modified similarity algorithm for collaborative filtering. In L. Uden, W. Lu, & I.-H. Ting (Eds.), *Knowledge Management in Organizations* (pp. 378–385). Cham: Springer International Publishing.

Shen, L. and Zhou, Y. (2010). A new user similarity measure for collaborative filtering algorithm. In 2010 Second International Conference on Computer Modeling and Simulation, Vol. 2, pages 375–379.

Sheugh, L., & Alizadeh, S. H. (2015). A note on pearson correlation coefficient as a metric of similarity in recommender system. In *2015 AI Robotics (IRANOPEN)* (pp. 1–6).

Singh, P. K., Pramanik, P. K. D., & Choudhury, P. (2019). A comparative study of different similarity metrics in highly sparse rating dataset. In *Data Management, Analytics and Innovation* (pp. 45–60). Springer.

Singh, P. K., Setta, S., and Rajput, I. S. (2019). A modified spearman's rank correlation coefficient for an efficient method of similarity calculation in collaborative filtering-based recommendation. Available at SSRN 3368728.

Sivaramakrishnan, N., Subramaniyaswamy, V., Arunkumar, S., Renugadevi, A., & Ashikamai, K. (2018). Neighborhood-based approach of collaborative filtering techniques for book recommendation system. *International Journal of Pure and Applied Mathematics, 119*(12), 13241–13250.

Son, L. H. (2016). Dealing with the new user cold-start problem in recommender systems: A comparative review. *Information Systems, 58*, 87–104.

Spertus, E., Sahami, M., and Buyukkokten, O. (2005). Evaluating similarity measures: A large-scale study in the orkut social network. In Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, KDD '05, pages 678–684, New York, NY, USA. ACM.

Sreepada, R. S., & Patra, B. K. (2018). An incremental approach for collaborative filtering in streaming scenarios. In G. Pasi, B. Piwowarski, L. Azzopardi, & A. Hanbury (Eds.), *Advances in Information Retrieval* (pp. 632–637). Cham: Springer International Publishing.

Stephen, S. C., Xie, H., and Rai, S. (2017). Measures of similarity in memory-based collaborative filtering recommender system: A comparison. In Proceedings of the 4th Multidisciplinary International Social Networks Conference, MISNC '17, pages 32:1–32:8, New York, NY, USA. ACM.

Su, X., & Khoshgoftaar, T. M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence, 17*(9), 261–273.

Sun, H., Zheng, Z., Chen, J., & Lyu, M. R. (2011). Nrcf: A novel collaborative filtering method for service recommendation. In *2011 IEEE International Conference on Web Services* (pp. 702–703).

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. *Journal of Computer Science and Technology, 27*(6), 1252–1260.

Sun, S.-B., Zhang, Z.-H., Dong, X.-L., Zhang, H.-R., Li, T.-J., Zhang, L., & Min, F. (2017). Integrating triangle and jaccard similarities for recommendation. *PLOS ONE, 12*(8), 1–16.

Suryakant and Mahara, T. (2016). A new similarity measure based on mean measure of divergence for collaborative filtering in sparse environment. Procedia Computer Science, 89(C):450–456.

Symeonidis, P., Nanopoulos, A., Papadopoulos, A. N., and Manolopoulos, Y. (2006). Collaborative filtering: Fallacies and insights in measuring similarity. Universitaet Kassel.

Tan, Z., & He, L. (2017). An efficient similarity measure for user-based collaborative filtering recommender systems inspired by the physical resonance principle. *IEEE Access, 5*, 27211–27228.

Pan, T., Liu, Q., and Chang, L. (2017). Ratings distribution recommendation model-based collaborative filtering recommendation algorithm. DEStech Transactions on Computer Science and Engineering, (smce).

Thorndike, R. L. (1953). Who belongs in the family? *Psychometrika, 18*(4), 267–276.

Tsuchiya, Y. and Nobuhara, H. (2018). Improved listwise collaborative filtering with high-rating-based similarity and temporary ratings. In IUI Workshops.

Tsuchiya, Y., & Nobuhara, H. (2019). Listwise collaborative filtering with high-rating-based similarity and simple missing value estimation. *Journal of Japan Society for Fuzzy Theory and Intelligent Informatics, 31*, 501–507.

ur Rehman, Z., Hussain, F. K., and Hussain, O. K. (2013). Frequency-based similarity measure for multimedia recommender systems. Multimedia Systems, 19(2):95–102.

Wang, S., Zhao, Z., & Hong, X. (2015). The research on collaborative filtering recommendation algorithm based on improved clustering processing. In *Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on* (pp. 1012–1015). IEEE.

Wang, Y., Deng, J., Gao, J., & Zhang, P. (2017). A hybrid user similarity model for collaborative filtering. *Information Sciences, 418–419*, 102–118.

Wei, W., Guangquan, Z., & Jie, L. (2015). Collaborative filtering with entropy-driven user similarity in recommender systems. *International Journal of Intelligent Systems, 30*(8), 854–870.

Weng, L. T., Xu, Y., Li, Y., & Nayak, R. (2005). An improvement to collaborative filtering for recommender systems. In *Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on* (Vol. 1, pp. 792–795). IEEE.

Wu, F., He, L., Ren, L., & Xia, W. (2008). An effective similarity measure for collaborative filtering. In *2008 IEEE International Conference on Granular Computing* (pp. 659–664).

Wu, X., Cheng, B., & Chen, J. (2017). Collaborative filtering service recommendation based on a novel similarity computation method. *IEEE Transactions on Services Computing, 10*(3), 352–365.

Xiaoping, L. (2015). User similarity measure method based on the comparison model of psychology. In *2015 Seventh International Conference on Measuring Technology and Mechatronics Automation* (pp. 1386–1389).

Xu, R., Wang, S., Zheng, X., & Chen, Y. (2014). Distributed collaborative filtering with singular ratings for large scale recommendation. *Journal of Systems and Software, 95*, 231–241.

Yan, H., & Tang, Y. (2019). Collaborative filtering based on gaussian mixture model and improved jaccard similarity. *IEEE Access, 7*, 118690–118701.

Yang, C., Wei, B., Wu, J., Zhang, Y., and Zhang, L. (2009). Cares: A ranking-oriented cadal recommender system. In Proceedings of the 9th ACM/IEEE-CS Joint Conference on Digital Libraries, JCDL '09, pages 203–212, New York, NY, USA. ACM.

Yao, Y., Yuan, H., Xie, F., & Chen, Z. (2013). Sofa: Statistic based collaborative filtering algorithm. In *2013 Fourth International Conference on Networking and Distributed Computing* (pp. 136–140).

Zhang, F., Gong, T., Lee, V. E., Zhao, G., Rong, C., & Qu, G. (2016). Fast algorithms to evaluate collaborative filtering recommender systems. *Knowledge-Based System, 96 (C):96–103*.

Zhang, F., Zhou, W., Sun, L., Lin, X., Liu, H., & He, Z. (2017). Improvement of pearson similarity coefficient based on item frequency. In *2017 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR)* (pp. 248–253).

Zhang, Y. and Andreae, P. (2008). Iterative neighbourhood similarity computation for collaborative filtering. In 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Vol. 1, pages 806–809.