User Profile Correlation-Based Similarity Algorithm in Movie Recommendation System

Triyanna Widiyaningtyas
Universitas Gadjah Mada
Indriana Hidayah
Universitas Gadjah Mada
Teguh Bharata Adji (adji@ugm.ac.id)
Universitas Gadjah Mada https://orcid.org/0000-0001-7856-1498

Research

Keywords: collaborative ltering, user rating value, user behavior value, UPCSim

Posted Date: November 18th, 2020

DOI: https://doi.org/10.21203/rs.3.rs-107401/v1

Version of Record: A version of this preprint was published at Journal of Big Data on March 29th, 2021.
See the published version at https://doi.org/10.1186/s40537-021-00425-x.

RESEARCH

User profile correlation-based similarity algorithm in movie recommendation system

Triyanna Widiyaningtyas[1,2], Indriana Hidayah[1] and Teguh B. Adji[1*]

*Correspondence: adji@ugm.ac.id
[1]Department of Information Technology and Electrical Engineering, Universitas Gadjah Mada, Yogyakarta, Indonesia
Full list of author information is available at the end of the article

Abstract

A recommendation system is a software used in the e-commerce field that provides recommendations for customers to choose the items they like. Several recommendation systems have been proposed; however, collaborative filtering is the most widely used approach. The main issue in collaborative filtering is how to implement a similarity algorithm that can improve performance in the recommendation system. Several similarity algorithms based on user rating value have been developed, and recently a similarity algorithm has been developed that combines the user rating value and the user behavior value. However, the existing research is still based only on a single user behavior value, which is the genre data. Therefore, we propose a new similarity algorithm that considers not only the genre data but also the user profile data (namely age, gender, occupation, and location). The new similarity we are proposing is called User Profile Correlation-based Similarity (UPCSim). The user profile correlation similarity was obtained by calculating the correlation coefficient between the user profile data and the user rating or behavior values. An experiment was done to compare the accuracy of the UPCSim algorithm with that of the previous algorithm. The experiment results showed that the UPCSim algorithm can improve the recommendation performance MAE by 1.64% and RMSE by 1.4% compared to the previous algorithm.

Keywords: collaborative filtering; user rating value; user behavior value; UPCSim

Introduction

The exponential growth of information on the internet causes users can get more information resources to dig up and collect. However, users will get lost in the sea of information and will have difficulty in processing that information [1, 2]. Users have to spend more time and more energy finding the information they want, but users may not necessarily get satisfactory results. Fortunately, user behavior on e-commerce sites and other social networks can be recorded and be tracked, making it easier to analyze user interests [3, 4]. One of the tools used to solve this problem in analyzing user interest is known as a recommendation system.

The recommendation system is software that helps users to get relevant items from millions of items in the database [5, 6]. The recommendation system's main task is to offer users personalized item recommendations through information filtering. This system has become a commercial platform that assists the user in providing suggestions for items to be selected. The provided suggestions are useful to support users in various decision-making processes, such as what books to read, which locations to visit, what news to read, and more [7].

Based on the utilized data source and computation method, the recommendation system is divided into three approaches, namely: collaborative filtering, content-based filtering, and hybrid filtering [6, 8]. The collaborative filtering approach uses the collaborative power of ratings given by all users to make recommendations. The content-based filtering approach uses descriptive attributes of items to make recommendations. Meanwhile, the hybrid filtering approach combines several filtering methods to get a list of items according to user preferences [9].

Of the three recommendation system approaches, collaborative filtering is one of the most popular, successful, and widely used methods in recommendation systems [5, 10, 11]. This recommendation method is widely used because it is simple, efficient, and has an acceptable accuracy level. Based on the advantages of this collaborative filtering, several real-world systems have used this method, such as Amazon, MovieLens, and Netflix [8, 10, 12].

The collaborative filtering approach is categorized into two approaches, viz ranking-oriented collaborative filtering and rating-oriented collaborative filtering [13, 14]. Ranking-oriented collaborative filtering directly provides a preference order for items from users without predicting ratings for items that have not been rated. In contrast, rating-oriented collaborative filtering predicts ratings for items that have not been rated by users based on rating information from other users [14]. The rating-oriented collaborative filtering approach is more widely used because it is faster in generating recommendations than the ranking-oriented collaborative filtering approach.

The rating-oriented collaborative filtering approach is categorized into two methods, viz the model-based method and the memory-based method [8, 15, 16]. The model-based method uses a rating database to build a model and uses that model to predict ratings for unrated items [17].Some of the techniques that are often used to build models include clustering [18, 19], Bayesian network [20], Markovian factorization [21], and Singular Value Decomposition (SVD) [22, 23]. This model-based method has a drawback; namely, the computational complexity is very dependent on the model being built [3].Therefore, the algorithm complexity value cannot be ascertained.

Meanwhile, the memory-based method uses the rating database to calculate the similarity between users or similarity between items [24].In its implementation, this method is divided into two techniques, namely User-Based Collaborative Filtering (UBCF) and Item-Based Collaborative Filtering (IBCF) [1, 2, 25]. The UBCF predicts ratings for all items that have not been rated based on the similarity of users, while the IBCF predicts ratings based on the similarity of items [26].Some of the frequently used traditional similarities are the Cosine (COS), Pearson Correlation Coefficient (PCC), and Jaccard [2, 8].

In recent years, the majority of researchers have focused more on developing a similarity algorithm between users/items, because of its simplicity in computation. For example, Patra et al. [16] proposed a similarity algorithm known as Bhattacharyya similarity. Furthermore, Polatidis et al. [27] proposed a similarity algorithm that is an increase in PCC similarity, known as multi-level collaborative filtering. Their proposed similarity is an increase in the PCC similarity by taking into account the number of items co-rated on several levels. Sun et al. [28] proposed a similarity

algorithm by integrating the similarity of Triangle and Jaccard, known as TMJ similarity. Feng et al. [2] proposed a new similarity algorithm by integrating three factors of similarity impact, namely S_1,S_2, and S_3. With S_1 expressing similarity between users, S_2 is used to calculate the number of items co-rated less than the specified threshold, and S_3 explains the weight of each user's rating preference. The four similarity algorithms that have been developed by previous researchers perform similarity calculations based only on the user's rating value data.

Furthermore, Wu et al. [29] proposed a new similarity by combining similarity based on the user's rating value and similarity based on user behavior value. Wu et al. [29] view that the traditional recommendation algorithm only pays attention to the rating value given by users explicitly, but ignores implicit information on user behavior in assessing the item type (genre) that will also affect the accuracy level. In their research, similarity based on the user's rating value (S_r) was calculated using the UBCF method. Meanwhile, similarity based on user behavior value (S_b) is calculated from the probability of the user score in giving a rating to the genre. The proposed similarity algorithm is known as the User Score Probability Collaborative Filtering (UPCF).

By combining the two similarities, the similarity calculation between users not only require user rating value data but also genre data to calculate the user behavior value. The research results can provide an increase in accuracy compared to the application of the traditional similarity algorithm, which only considers the user rating value. However, there is still a need to improve the recommendations' accuracy by exploring other user behavior affecting user interests.

Therefore, we propose a new similarity algorithm – that is called User Profile Correlation-based Similarity (UPCSim) – which not only pays attention to genre data, but also adds other user behavior data in term of user profile data (namely age, gender, occupation, and location) on giving weight to similarity. In this UPCSim algorithm, the similarity weighting technique is obtained by calculating the correlation coefficient between the user profile data (namely age, gender, occupation, and location) and the user rating value for the similarity weight of S_r, while the similarity weight of S_b is obtained by calculating the correlation coefficient between the user profile data and the user behavior value.

The structure of this paper is as follows. In "Similarity Algorithm" section describes the several similarity algorithm which developed in the collaborative filtering approach. Then, "Research Method" section explains the proposed similarity algorithm in detail. Subsequently, "Experiment" section present the experiment's results using the MovieLens dataset and its discussion. Finally, "Conclusion and Future Work section provides some conclusions and suggestions for further research development".

Similarity Algorithm

In the previous section, it has been explained that the frequently used traditional similarity algorithms in recommendation systems are the Cosine similarity (COS) and the Pearson Correlation Coefficient (PCC) [30]. To explain this similarity calculation, we assume that the user and item sets are defined as $U = \{u_1, u_2, \ldots, u_m\}$ and $I = \{i_1, i_2, \ldots, i_n\}$. The user rating matrix for the item is denoted as

$R = [r_{ui}]^{m \times n}$, where m and n are the number of users and the number of items, respectively, and r_{ui} is the rating given by user u on item i

Cosine similarity measures the angle between two rating vectors (user or item) [2, 3, 16]. The Cosine similarity formula between user u_1 and user u_2 is stated in (1).

$$Sim(u_1, u_2)^{COS} = \frac{\vec{r}_{u_1} . \vec{r}_{u_2}}{\|\vec{r}_{u_1}\| . \|\vec{r}_{u_2}\|} = \frac{\sum_{i \in I_{u_1} \cap I_{u_2}} r_{u_1 i} . r_{u_2 i}}{\sqrt{\sum_{i \in I_{u_1} \cap I_{u_2}} r_{u_1 i}^2} \cdot \sqrt{\sum_{i \in I_{u_1} \cap I_{u_2}} r_{u_2 i}^2}} \quad (1)$$

Pearson similarity measures how two users or items are linearly related to each other [3, 16]. After identifying the items rated jointly between user u_1 and user u_2, Pearson similarity calculates the linear correlation between the two users using the formula specified in (2). Pearson similarity values ranged in the range [-1, +1]. A value of +1 indicates a very high correlation and -1 indicates a negative correlation.

$$Sim(u_1, u_2)^{PCC} = \frac{\sum_{i \in I_{u_1} \cap I_{u_2}} (r_{u_1 i} - \overline{r}_{u_1}) . (r_{u_2 i} - \overline{r}_{u_2})}{\sqrt{\sum_{i \in I_{u_1} \cap I_{u_2}} (r_{u_1 i} - \overline{r}_{u_1})^2} \cdot \sqrt{\sum_{i \in I_{u_1} \cap I_{u_2}} (r_{u_2 i} - \overline{r}_{u_2})^2}} \quad (2)$$

Several similarity improvements are continuously being developed to increase the recommendations' accuracy. Among them are Bhattacharyya's similarity [16], the multi-level collaborative filtering similarity [27], the TMJ similarity [28], and the similarity integrating three impact factors, namely S_1, S_2, dan S_3 [2]. These four similarity algorithms only consider user rating value data explicitly to calculate the similarity between users. These similarity algorithms assume that users who give a high rating on an item indicate that they like the item, while users who give a low rating indicate that they do not like the item.

In the current era of online shopping, users assess items based on their quality, delivery speed, service attitude to customers, and other factors. If the user is not interested in an item, the user will not select/buy the item. The illustration of user behavior in assessing items is shown in Figure 1. The User u likes to watch genre animation and does not like to watch drama and adventure genres. The User u selects the movie title "Toy Story" from a collection of animated films. After watching the movie, the User u gave a low rating to "Toy Story" because the "Toy Story" movie gave unclear sound quality.

In the similarity algorithm, which is only based on the user rating value, if a user gives a low rating, it means that the user does not like the movie title as well as the movie genre. This indicates that the user's preference for similar items will decrease and will affect the resulting recommendations. Therefore, it is necessary to involve user behavior in rating items to calculate similarity as a form of invisible user preferences.

To adopt this user behavior, Wu et al. [29] developed a new similarity algorithm that involves not only explicit rating data but also user behavior data in giving an implicit rating. Wu et al.[29] assumed that users who gave a low rating to an item did not necessarily dislike the item. The similarity algorithm combines similarity

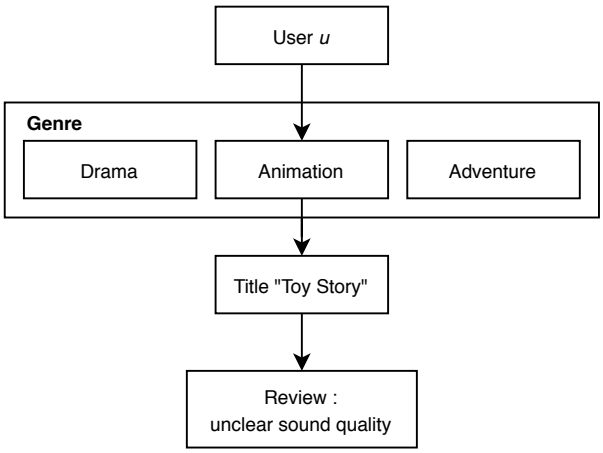

Figure 1 Example of user behavior in rating the item

based on user rating value (S_r) and similarity based on user behavior value (S_b). The formula of this similarity is shown in (3) below.

$$Sim(u_1, u_2)^{UPCF} = \beta S_r(u_1, u_2) + (1 - \beta) S_b(u_1, u_2) \quad (3)$$

S_r (u_1,u_2) is the similarity between user u_1 and user u_2 calculated by the UBCF method, β is the threshold (which is between 0 to 1) that can be set to the average value of the similarity of all the users who are similar to the active user u_1, and S_b is defined in (4).

$$S_b(u_1, u_2) = \frac{\sum_{g \in G_{u_1} \cap G_{u_2}} (P_{u_1 g} - \overline{P}_{u_1}).(P_{u_2 g} - \overline{P}_{u_2})}{\sqrt{\sum_{g \in G_{u_1} \cap G_{u_2}} (P_{u_1 g} - \overline{P}_{u_1})^2} . \sqrt{\sum_{g \in G_{u_1} \cap G_{u_2}} (P_{u_2 g} - \overline{P}_{u_2})^2}} \quad (4)$$

G_{u_1} and G_{u_2} are the set of item types (genres) rated by users u_1 and users u_2 respectively. $P_{u_1 g}$ and $P_{u_2 g}$ are the probability scores for the type of item g from users u_1 and users u_2 respectively. $\overline{P}_{u_1}$ and $\overline{P}_{u_2}$ are the average probability scores for all item types from users u_1 and users u_2, respectively, and g is one type of items rated by both users.

The combination of the two similarities in the research conducted by Wu et al. [29] has several limitations, viz the calculation of similarity based on user behavior value only considers the genre data of the item so that it does not guarantee the resulted recommendations accuracy. Based on the problem, this paper focuses on increasing accuracy by considering other user behavior data that is the user profile data (namely age, gender, occupation, and location) that will influence user behavior in determining the selected item.

We assumed that age, gender, occupation, and location would influence the user's interest in the item. As an illustration, young users would have different preferences from older users. Female users would have different preferences from male users. Users with a technician job would have different preferences from users with a lawyer job, and users living on the coast would have different preferences from

users living in cities. Based on this assumption, the main contribution of our study is the proposed new similarity algorithm that considers not only genre but also age, gender, occupation, and location data to improve accuracy in dealing with items that have not been rated by users (data sparsity). The detail of this algorithm is be described in the next section.

Research Method

The conducted research was more focused on the proposed UPCSim algorithm. This UPCSim algorithm would be compared with the UPCF algorithm [29], because the UPCF already accommodates one user behavior, viz the genre data. Meanwhile, the UPCSim algorithm adds other user behavior data, namely the user profile data. For this comparison, an MBCF system applying the UPCSim algorithm was created. It should be noted that all the pre-processing and processing of MBCF we did were the same as that done by Wu et al. [29]. So that the comparison between the two studies is equal.

The MBCF system that we have developed is a development of the MBCF system that was previously developed by Wu et al. [29]. A detailed illustration of the proposed MBCF system is shown in Figure 2.

The system is divided into four blocks, namely input, data preparation, the MBCF process, and output. The input block is the input dataset used in the MBCF system. The data preparation block consists of the data pre-processing stage, which results in data source ready for use in the MBCF process. The data source includes the rating data and the behavior data. While the behavior data used in Wu et al. [29] only employs the genre data (the green component in data preparation block), our research also accommodated the user profile data (the red component in data preparation block). The MBCF process block is the development of the MBCF method using the weighting similarity. The similarity weighting carried out by Wu et al. used a threshold value ranging from 0 to 1 (the green component in the MBCF process). Whilst, the weighting similarity in our research used the coefficient correlation between the user profile data and the user rating or behavior values (the red component in MBCF process). Furthermore, the final block of the developed system is the output block that provides an evaluation of the UPCSim algorithm application in the MBCF method.

The detail of our proposed UPCSim algorithm is explained as a component of similarity calculation, which is presented in "Similarity Calculation" section. Meanwhile the detail of the developed MBCF system is explained in "Developed MBCF System" section.

Similarity Calculation

In this study, the similarity calculation between users was divided into three components. The first component is the S_r similarity calculation component (shown in the dashed blue box). The second block is the S_b similarity calculation component (shown in the dashed green box). Finally, the UPCSim block (shown in the dashed red box) gives weight to both similarities. Details of the three component in our similarity calculation can be seen in Figure 3.

Each component in Figure 3 is described as follows.

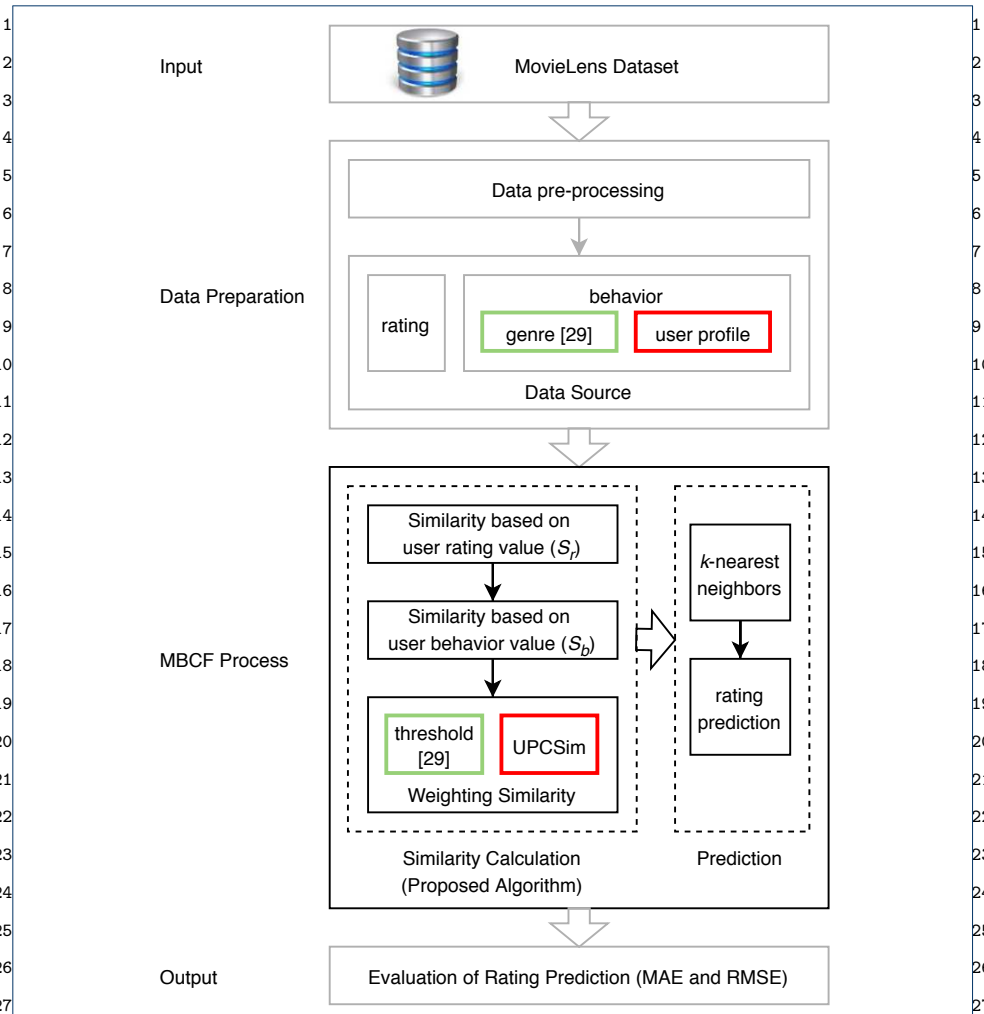

Figure 2 The developed MBCF system

S_r *Similarity*

The S_r similarity component is the similarity calculation based on the user rating value. As an example of the similarity calculation, we used the MovieLens 100K dataset. The initial stage in calculating the S_r similarity was done by reading the rating data from the resulted pre-processing data. Based on the rating data, we obtained a rating matrix of 943×1682. The value of 943 represented the number of users, and the value of 1682 represented the number of movies in the dataset. The rating matrix illustration is shown below.

$$R = \begin{bmatrix} R_{11} & R_{12} & R_{13} & R_{14} & \dots & R_{1\_1682} \\ R_{21} & R_{22} & R_{23} & R_{24} & \dots & R_{2\_1682} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ R_{943\_1} & R_{943\_2} & R_{943\_3} & R_{943\_4} & \dots & R_{943\_1682} \end{bmatrix}$$

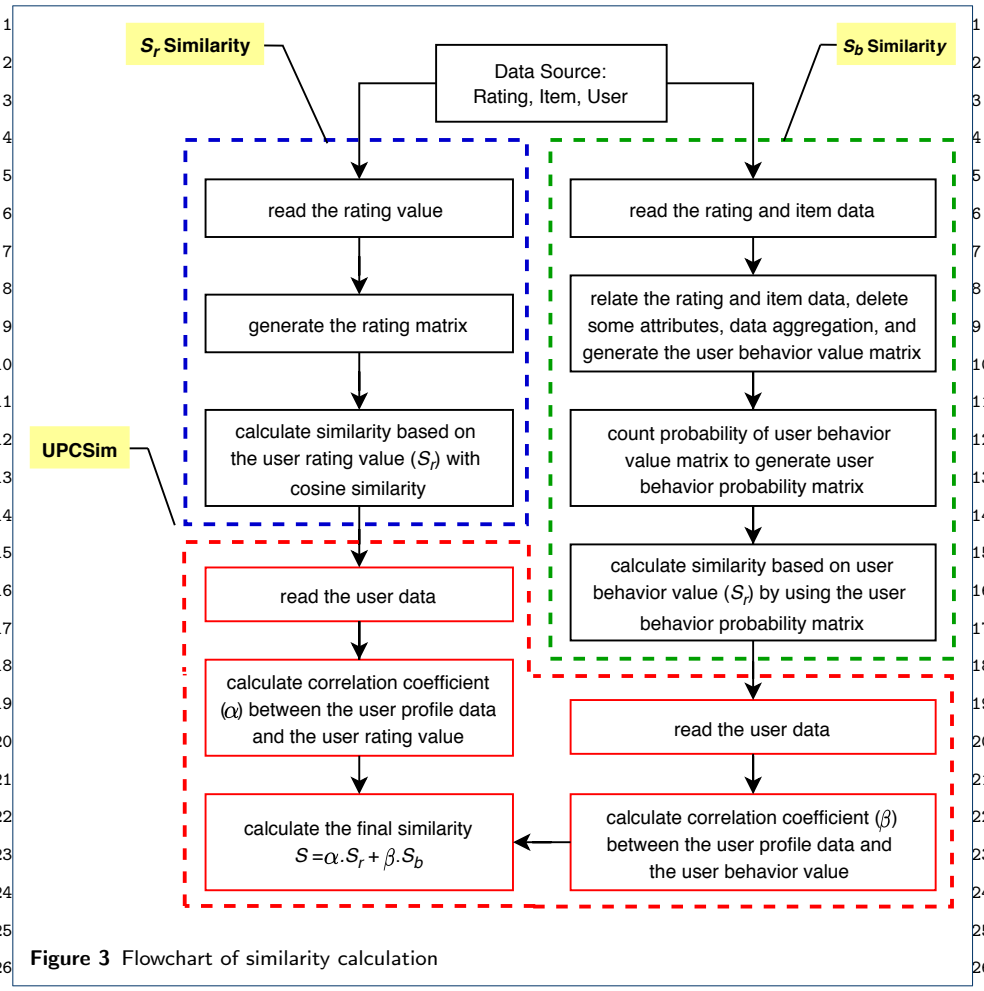

Figure 3 Flowchart of similarity calculation

R_{943_1682} is the rating value given by the 943^{rd} user for the 1682^{nd} item. Values R_{11} to $R_{9431682}$ range from 0 to 5, with a value of 0 indicating that the user unrated the item.

After the rating matrix is formed, the next step was to calculate the S_r similarity using the Cosine similarity formula referring to (1). The final result of the S_r similarity calculation would form the S_r similarity matrix with the order 943×943. Illustration of the similarity matrix S_r is shown as follows.

$$S_r = \begin{bmatrix} S_{11} & S_{12} & S_{13} & \dots & S_{1\_943} \\ S_{21} & S_{22} & S_{23} & \dots & S_{2\_943} \\ \dots & \dots & \dots & \dots & \dots \\ S_{943\_1} & S_{943\_2} & S_{943\_3} & \dots & S_{943\_943} \end{bmatrix}$$

S_{1_943} is the similarity value based on the rating between the 1^{st} user and the 943^{rd} user.

S_b Similarity

The S_b similarity component is the similarity calculation based on the user behavior value. The initial stage in calculating the S_b similarity was done by reading the

rating and item data. In the MovieLens 100K dataset, the rating data stated the user's rating value for the watched movies, and the item data stated the movie title data containing the genre information of each movie. Each movie title can include several genres. For example, the movie "Toy Story" has the genre of animation, children, and comedy. The rating and item data are used to calculate the user bahavior value.

The user behavior value could be calculated by relating the rating data and the item data, removing some unused attributes from the results of the relationship between the rating data and the item data, and performing data aggregation using the sum function grouped by user. This data aggregation results are illustrated in the user behavior value matrix with the order matrix of 943×19, , where the value 943 represents the number of users, and the value 19 represents the number of genres that exist. The illustration of user behavior value matrix is shown as follows.

$$B = \begin{bmatrix} B_{11} & B_{12} & B_{13} & ... & B_{1\_19} \\ B_{21} & B_{22} & B_{23} & ... & B_{2\_19} \\ ... & ... & ... & ... & ... \\ B_{943\_1} & B_{943\_2} & B_{943\_3} & ... & B_{943\_19} \end{bmatrix}$$

B_{943_19} is the 943^{rd} user behavior value for the 19^{th} genre, representing the total number of 19^{th} genre watched by 943^{rd} user. After the user behavior value matrix was formed, the next stage was to calculate the probability of genres occurrence from the user behavior value matrix to produce a probability matrix of user behavior value using (5) below.

$$P = \frac{B(g)}{N} \tag{5}$$

$B(g)$ is the value of user behavior for the target genre g, and N is the total number of users who gave rate to the target genre g. The illustration of the probability matrix of user behavior value is shown as follows.

$$B = \begin{bmatrix} P_{11} & P_{12} & P_{13} & ... & P_{1\_19} \\ P_{21} & P_{22} & P_{23} & ... & P_{2\_19} \\ ... & ... & ... & ... & ... \\ P_{943\_1} & P_{943\_2} & P_{943\_3} & ... & P_{943\_19} \end{bmatrix}$$

P_{943_19} is the probability value of the 943^{rd} user behavior for the 19^{th} genre. The probability matrix of user behavior value was used as the basis for calculating the S_b similarity referring to (4). The results of the S_b similarity calculation would form a matrix with the order 943×943. The illustration of the S_b similarity matrix is shown as follows.

$$S_b = \begin{bmatrix} S_{11} & S_{12} & S_{13} & ... & S_{1\_943} \\ S_{21} & S_{22} & S_{23} & ... & S_{2\_943} \\ ... & ... & ... & ... & ... \\ S_{943\_1} & S_{943\_2} & S_{943\_3} & ... & S_{943\_943} \end{bmatrix}$$

S_{1_943} is the similarity value based on the user behavior value the 1^{st} user and the 943^{rd} user.

UPCSim

The UPCSim component is the component of the similarity calculation using the UPCSim algorithm, which calculates the weights of both similarities (S_r and S_b) based on user profile. The initial stage in calculating the weights was done by reading the user data. In the MovieLens 100K dataset, the user data stated the user profile data that consisted of age, gender, occupation, and location. The user profile data was used to calculate the weights of S_r and S_b similarities. The weights of these two similarities were calculated based on the correlation coefficient (R) using multiple linear regression. The weight of S_r similarity was obtained by calculating the correlation coefficient between user profile data (age, gender, occupation, and location) and the user rating value, which was then symbolized by α. While the weight of S_b similarity was obtained by calculating the correlation coefficient between user profile data (age, sex, occupation, and location) and the user behavior value, which was then symbolized by β.

After weighting the two similarities, the next stage was to calculate the final similarity matrix by combining the weighted S_r and S_b similarities. By combining these two similarities, we could obtain the final similarity matrix S with the order matrix 943×943. The formula of the final similarity between user u and user v is defined in (6) below.

$$S(u,v) = \alpha S_r(u,v) + \beta S_b(u,v) \tag{6}$$

$S(u,v)$ is the final similarity between user u and user v. $S_r(u,v)$ is the similarity based on user rating value between user u and user v. $S_b(u,v)$ is the similarity based on user behavior value between users u and user v. α is the weight of the similarity S_r, and β is the weight of the similarity S_b.

Developed MBCF System

Based on the illustration shown in Figure 2, this section describes each block of the developed MBCF system.

Input

The first block of the developed MBCF system was the input dataset. In this paper, we used the MovieLens dataset (https://grouplens.org/datasets/movielens/). This dataset was collected by the "GroupLens Study Group of the University of Minnesota" [31]. There were several versions of the dataset, including ml-100K, ml-1M, ml-10M, ml-20M, etc. In this experiment, we chose the dataset used in previous studies [29], namely ml-100K. This ml-100K dataset contained several data files. Our study used 3 data files, namely rating data, item data, and user data..

The rating data consisted of 100,000 ratings as rated by 943 users on 1682 movies. Each user has rated at least 20 movies. The rating value given by the user ranged from 1 to 5. A score of 1 stated that the user did not like the watched movie. A score of 5 stated that the user liked the watched movie. This rating data had a

sparsity of 93.7% and a density of 6.3%. This rating data structure consisted of user-id, movie-id, rating, and timestamp.

Item data contains information about items (movies). This item's data structure consisted of 24 attributes, namely, movie-id, movie title, release date, video release date, IMDb URL, and 19 attributes of movie type (genre). Each item (movies) can have several genres.

User data contains information about the user's profile. This user data structure consisted of 5 attributes, namely user-id, age, gender, occupation, and zip code (which states the user's location).

Data Preparation

The second block was data preparation. In this section, data pre-processing was carried out. The purpose of this process was to prepare data to obtain a quality dataset. In this study, data pre-processing was done by reducing attributes that were not relevant to data processing.

Based on the existing file data structure, several attributes were not needed in data processing. These attributes included the timestamp on the rating data, movie title, release date, video release date, and IMDb URL on the item data. At the dataset preparation stage, these attributes would be deleted so that the data processing stage ran more effectively.

MBCF Process

The third block of the MBCF system was the MBCF process. The MBCF process was divided into two sub blocks, namely the similarity calculation and the prediction.

The similarity calculation was the initial process used in the information filtering process using the MBCF approach. In this study, the similarity calculation was divided into three components, namely S_r similarity calculation, S_b similarity calculation, and the UPCSim algorithm. In detail, these three components have been described in "Similarity Calculation" section.

The prediction was carried out to provide a predicted rating for items that had not been rated by active users. The initial stage taken in this prediction was to determine the number (k) of the nearest neighbors of an active users. k was an integer number representing the number of neighbors, ranging from 10 to 100 [2, 27–29]. After the k value was determined, the next stage was to determine the rating prediction for some unrated items.

The formula used to determine the predicted rating for an item (i) unrated by active users (u) is shown in (7) below [2, 16].

$$p_{ui} = \bar{r}_u + \frac{\sum_{v \in NNu} S(u,v).(r_{vi} - \bar{r}_v)}{\sum_{v \in NNu} |S(u,v)|}, v \neq u \quad (7)$$

p_{ui} represents the predicted rating value of user u to item i. $\bar{r}_u$ dan $\bar{r}_v$ is the average rating of user u and user v respectively. r_{vi} is the rating value given by user v to item i. $S(u,v)$ is the final similarity between user u and user v. NNu is the set of nearest neighbors to user u.

Output

The fourth block of the MBCF system was the output block. This block was used to evaluate the performance of the UPCSim Algorithm in predicting ratings for items that had not been rated by an active user.

To measure the performance of the recommendation system, mean absolute error (MAE), rooted mean squared error (RMSE), precision, and recall were the most popular measures. According to Jalili et al. [32], the metrics for evaluating recommendation systems can be classified into two categories, namely prediction metrics and classification metrics. The MAE and RMSE are primarily used to evaluate prediction metrics [33, 34], whereas precision and recall are used to evaluate classification metrics, namely evaluation of the quality of top-N recommendations [3].

In this study, we adopted the MAE and RMSE metrics to measure the prediction metrics of the UPCSim Algorithm. The MAE is the most widely used metric in recommendation systems with a collaborative filtering approach. It is used to estimate the average absolute deviation between the actual and the predicted rating values. A lower MAE provides good recommendation quality [35]. The formula for calculating MAE is defined in (8).

$$MAE = \frac{1}{TN} \sum_{u \in U, i \in I} |p_{ui} - r_{ui}| \tag{8}$$

Meanwhile, RMSE reflects the degree of deviation between the predicted rating and the actual rating. A lower RSME is associated with higher prediction metrics [36]. The RMSE formula is defined in (9).

$$RMSE = \sqrt{\frac{1}{TN} \sum_{u \in U, i \in I} (p_{ui} - r_{ui})^2} \tag{9}$$

TN is the total number of predicted items. p_{ui} and r_{ui} represent the predicted rating and actual rating of the user u to item i, respectively.

Experiment

This section begins with the experiment design described in "Experiment Design" section. Next, the "Experiment Result and Analysis" section explain the comparison between the proposed UPCSim algorithm and the previous similarity algorithms, namely the Cosine similarity algorithm and the UPCF similarity algorithm. The comparison utilized the MAE and RMSE values of the MBCF system that had been made. Finally, "Discussion" section provides conclusions from the experiment results.

Experiment Design

To evaluate the performance of the proposed UPCSim algorithm, the experiment design in this study implemented the following four steps:

The first step was to divide the dataset. The dataset would be divided into two parts, viz training data, and testing data. The k-fold cross-validation method was used in dividing the dataset. In this experiment, the chosen k was 5. It meant

that 80% of the dataset was used for training data and the remaining 20% was for testing data. The training data were train1, train2, train3, train4, and train5, and the testing data are called test1, test2, test3, test4, and test5.

The second step was to calculate the similarity matrix between users. The S_r similarity was obtained based on the user rating value matrix, and the S_b similarity was obtained based on the user behavior value matrix. Then, we calculated the weights of the two similarities using the correlation coefficient (R), with multiple linear regression analysis. The final similarity matrix was obtained by combining the two weighted similarities.

The third step was to calculate the predicted ratings for the testing data. The nearest neighboring k was selected based on the final similarity matrix. In this experiment, the k values varied from 10 to 100, with an increase in the k value by 10.

The fourth step was to measure the proposed UPCSim algorithm's performance using the MAE and RMSE prediction metrics.

Experiment Results and Analysis

This section aims to compare the proposed UPCSim algorithm's performance results with the traditional Cosine similarity and the UPCF similarity. The MAE and RMSE values were obtained by comparing the three algorithms using variations in the number of different neighbors. Experiments were carried out in five iterations. The first iteration was done using the train1 and test1 dataset. The second iteration was done using the train2 and test2 datasets, and so on. The number of nearest neighbors used in this experiment ranged from 10 to 100, which were used in each train1, train2, train3, train4, and train5 data. The three similarity algorithms (Cosine, UPCF, and UPCSim) were applied to the five-training data. The average MAE performance for the three algorithms is shown in Table 1.

In Table 1, MAE_c is the average MAE value of the UBCF experiment using Cosine similarity. The MAE_p is the average MAE value of the UBCF experiment using UPCF similarity. The MAE_ps is the average MAE value of the UBCF experiment using the proposed UPCSim similarity. The MAE_ps-c is the difference between the MAE value using the proposed UPCSim similarity and the MAE value using the Cosine similarity. And the MAE_ps-p is the difference between the MAE values using the proposed UPCSim similarity and the MAE values using the UPCF similarity.

Table 1 Comparison of average MAE values of the three algorithms

Number of Neighbors	MAE_c	MAE_p	MAE_ps	MAE_ps-c	MAE_ps-p
10	0.8227	0.7792	0.7669	0.0558	0.0123
20	0.8099	0.7631	0.7483	0.0616	0.0148
30	0.8051	0.7565	0.7410	0.0641	0.0155
40	0.8048	0.7551	0.7387	0.0661	0.0164
50	0.8043	0.7544	0.7369	0.0674	0.0175
60	**0.8041**	0.7535	0.7364	0.0677	0.0171
70	0.8049	0.7529	0.7359	0.0690	0.0170
80	0.8051	0.7526	0.7355	0.0696	0.0171
90	0.8056	0.7525	0.7347	0.0709	0.0178
100	0.8072	**0.7521**	**0.7337**	**0.0735**	**0.0184**
average	0.8074	0.7572	0.7408	0.0666	0.0164

Based on Table 1, an increase in the number of nearest neighbors results in a decreased MAE value in the UPCF similarity algorithm and the UPCSim similarity

algorithm. For the Cosine algorithm, the MAE value decreases only to $k = 60$. It shows that the number of nearest neighbors influenced MAE value; in other words, the number of nearest neighbors affects the algorithm's performance. The smallest MAE value is obtained in the UPCSim algorithm, which means that this algorithm's error prediction is the smallest. So, it can be said that UPCSim's algorithm is more reliable than others. Compared with the Cosine similarity, the UPCSim algorithm has an increase in MAE values ranging from 5.58% to 7.35%, with the average MAE of 6.66% for all k nearest neighbors. Compared with the UPCF similarity, the UPCSim algorithm has an increase in MAE values ranging from 1.23% to 1.84%, with the average MAE of 1.64% for all k nearest neighbors. By referring to the data in Table 1, the average MAE value of the three algorithms can be illustrated graphically, as in Figure 4.

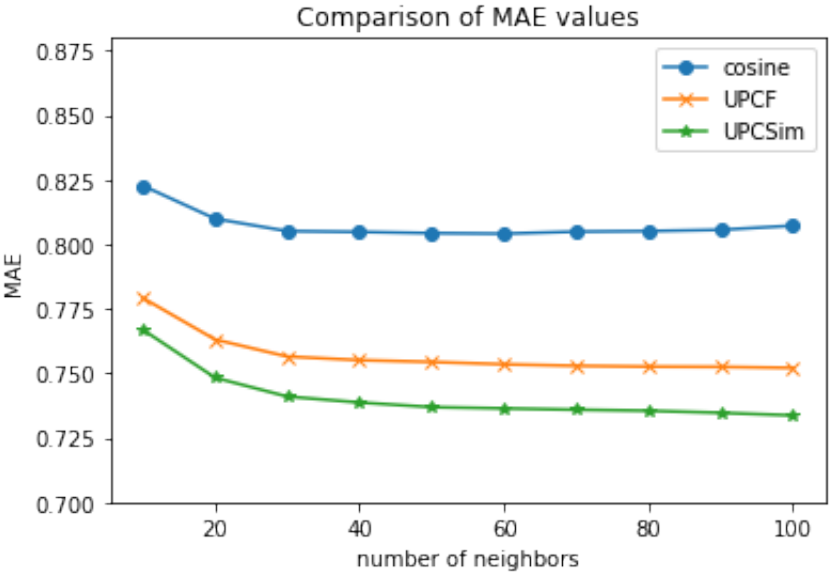

Figure 4 Comparison of the average MAE value of the three similarity algorithms using Movielens 100k dataset

Figure 4 shows that the three algorithms' MAE value decreases with an increasing number of nearest neighbors. At the beginning of the curve, it can be seen that the decline in MAE value is very sharp with the increase in the number of nearest neighbors, while at the end of the curve, the greater the number of nearest neighbors, the MAE value tends to be stable. It can be said that the number of nearest neighbor variables affects the MAE value, where the greater the number of nearest neighbors, the smaller the MAE value. With the same number of nearest neighbors, the MAE value of the UPCSim algorithm is always smaller than that of the other recommendation algorithms. In other words, the error between the actual rating and the predicted rating of the proposed UPCSim algorithm is the smallest with a more accurate rating prediction.

Furthermore, the comparison of the average RMSE values of the three recommendation algorithms is shown in Table 2. The RMSE_c is the average RMSE value from the UBCF experiment using Cosine similarity. The RMSE_p is the average RMSE

value of the UBCF experiment using UPCF similarity, and RMSE_ps is the average RMSE value of the UBCF experiment using the proposed UPCSim similarity. The RMSE_ps-c is the difference between the RMSE value based on the proposed UPCSim similarity and the RMSE value based on the Cosine similarity. The RMSE_ps-p is the difference between the RMSE value based on the proposed UPCSim similarity and RMSE value based on UPCF similarity.

Table 2 Comparison of average RMSE values of the three algorithms

Number of Neighbors	RMSE_c	RMSE_p	RMSE_ps	RMSE_ps-c	RMSE_ps-p
10	1.0417	0.9835	0.9793	0.0624	0.0042
20	1.0248	0.9643	0.9541	0.0707	0.0102
30	1.0189	0.9574	0.9453	0.0736	0.0121
40	1.0163	0.9558	0.9427	0.0736	0.0131
50	1.0162	0.9556	0.9393	0.0769	0.0163
60	**1.0154**	0.9547	0.9389	0.0765	0.0158
70	1.0162	0.9544	0.9383	0.0779	0.0161
80	1.0170	0.9543	0.9381	0.0789	0.0162
90	1.0173	0.9542	0.9364	0.0809	0.0178
100	1.0176	**0.9538**	**0.9359**	**0.0817**	**0.0179**
average	1.0201	0.9588	0.9448	0.0753	0.0140

As can be seen in Table 2, when the number of nearest neighbors is the same, the average RMSE value on the UPCSim similarity is always smaller than the other similarities. Meanwhile, an increase in the number of nearest neighbors results in a decreased RMSE value in the three algorithms. This shows that the number of nearest neighbors influences the RMSE value. The UPCSim algorithm produces the smallest RMSE value, which means that the proposed algorithm's prediction error is the smallest. So, it can be said that the UPCSim algorithm is superior. Compared to the Cosine similarity, the UPCSim algorithm has an increase in RMSE values ranging from 6.24% to 8.17%, with the average RMSE of 7.53% for all k nearest neighbors. Compared with the UPCF similarity, the UPCSim algorithm has an increase in the RMSE value ranging from 0.42% to 1.79%, with the average RMSE of 1.4% for all k nearest neighbors. Based on the data in Table 2, the three algorithms' average RMSE value can be illustrated graphically, as shown in Figure 5.

Figure 5 illustrates the effect of changes in the number of nearest neighbors on the RMSE value. Three algorithms show a decrease in the RMSE value first and tend to be stable when the neighbors' number is greater than 50. The RMSE value in the UPCSim algorithm always shows the smallest value for each different number of neighbors. It shows that the UPCSim algorithm has the lowest error rate compared to the other two algorithms. So, it can be said that the UPCSim algorithm is superior.

The superiority of the UPCSim similarity algorithm can be obtained because the algorithm considers the calculation of similarity weighting involving more complete user behavior (genre and user profile) than that used in Wu et al. [29], which only involved genre. So, the prediction metric is closer to the actual value. However, the UPCSim algorithm is considered computationally more complex than the previous recommendation algorithms so that if it is implemented in a larger dataset, it will take a longer time to produce recommendations.

Conclusion and Future Work

Our experiment results on the MovieLens 100K dataset show that the UPCSim algorithm can reduce MAE and RMSE values by 1.64% and 1.4%, respectively,

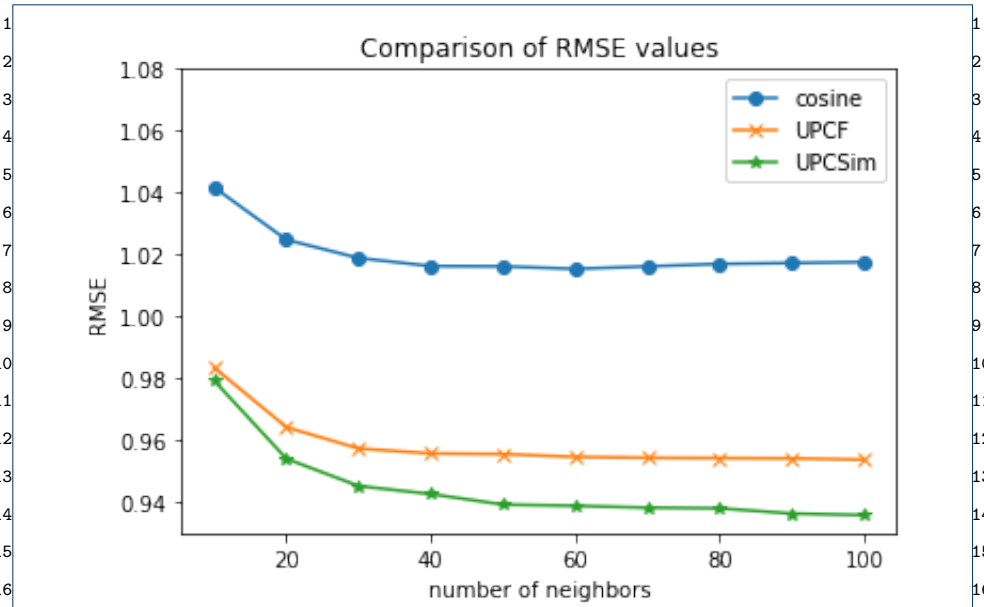

Figure 5 Comparison of the average RMSE value of the three similarity algorithms using Movielens 100k dataset

compared to UPCF algorithm. The strength of our algorithm is the accomodation of the user profile for calculating the similarity weighting in order to capture the user interest more accurately. Although the UPCSim similarity algorithm can improve the accuracy of prediction metrics, this study still has some limitations. The UPCSim algorithm is more complex than the previous algorithms. Thus, it will be time consuming if it is applied in a larger dataset. Besides, the UPCSim similarity algorithm still needs to improve the resulting prediction metrics. Therefore, in future studies, a clustering method can be considered to overcome scalability problems due to the larger number of datasets and reduce computation time.

Acknowledgements
Not applicable

Funding
This research was funded by Indonesia Endowment Fund for Education (LPDP), Ministry of Finance of Republic of Indonesia: *Beasiswa Unggulan Dosen Indonesia - Dalam Negeri* ($BUDI - DN$) contract number 20200421211035.

Abbreviations
UPCSim: User Profile Correlation-based Similarity; MAE: Mean Absolute Error; RMSE: Root Mean Squared Error; SVD: Singular Value Decomposition; MBCF: Memory-Based Collaborative Filtering; UBCF: User-Based Collaborative Filtering; IBCF: Item-Based Collaborative Filtering; COS: Cosine similarity; PCC: Pearson Correlation Coefficient; TMJ: Triangle Multipliying Jaccard; UPCF: User Probability score Collaborative Filtering

Availability of data and materials
The datasets generated and analysed during the current study are available in the MovieLens dataset (https://grouplens.org/datasets/movielens/)

Ethics approval and consent to participate
Not applicable

Competing interests
The authors declare that they have no competing interests.

Consent for publication
Not applicable

Authors' contributions

All authors contributed both the concepts and contents of this study. TW provided the manuscript under supervised by TBA and IH. All authors also performed discussion intensively for contents improvement. All authors have read and approved the final manuscript.

Author details

[1]Department of Information Technology and Electrical Engineering, Universitas Gadjah Mada, Yogyakarta, Indonesia. [2]Departement of Electrical Engineering, Universitas Negeri Malang, Malang, Indonesia.

References

1. Xu, G., Tang, Z., Ma, C., Liu, Y., Daneshmand, M.: A collaborative filtering recommendation algorithm based on user confidence and time context. Journal of Electrical and Computer Engineering **2019**, 1–12 (2019). doi:10.1155/2019/7070487
2. Feng, J., Fengs, X., Zhang, N., Peng, J.: An improved collaborative filtering method based on similarity. Plos One **13**(9), 1–18 (2018). doi:10.1371/journal.pone.0204003
3. Liu, H., Hu, Z., Mian, A., Tian, H., Zhu, X.: A new user similarity model to improve the accuracy of collaborative filtering. Knowledge-Based Systems **56**, 156–166 (2014). doi:10.1016/j.knosys.2013.11.006
4. Camacho, L.A., Alves-Souza, S.N.: Social network data to alleviate cold-start in recommender system: A systematic review. Information Processing and Management **54**, 529–544 (2018). doi:10.1016/j.ipm.2018.03.004
5. Sahu, A.K., Dwivedi, P.: User profile as a bridge in cross-domain recommender systems for sparsity reduction. Applied Intelligence **49**, 2461–2481 (2019). doi:10.1007/s10489-018-01402-3
6. Kumar, P., Kumar, V., Thakur, R.S.: A new approach for rating prediction system using collaborative filtering. Iran Journal of Computer Science **2**, 81–87 (2019). doi:10.1007/s42044-018-00028-5
7. Alonso, S., Bobadilla, J., Ortega, F., Moya, R.: Robust model-based reliability approach to tackle shilling attacks in collaborative filtering recommender systems. IEEE Access **7**, 41782–41798 (2019). doi:10.1109/ACCESS.2019.2905862
8. Salah, A., Rogovschi, N., Nadif, M.: A dynamic collaborative filtering system via a weighted clustering approach. Neurocomputing **175**, 206–215 (2015). doi:10.1016/j.neucom.2015.10.050
9. Aggarwal, C.C.: Recommender Systems. Springer, New York (2016). doi:10.1007/978-3-319-29659-3
10. Zhang, J., Lin, Y., Lin, M., Liu, J.: An effective collaborative filtering algorithm based on user preference clustering. Applied Intelligence **45**, 230–240 (2016). doi:10.1007/s10489-015-0756-9
11. Laishram, A., Padmanabhan, V., Lal, R.P.: Analysis of similarity measures in user-item subgroup based collaborative filtering via genetic algorithm. International Journal of Information Technology **10**(4), 523–527 (2018). doi:10.1007/s41870-018-0195-z
12. Bagher, R., Cami Hassanpour, H., Mashayekhi, H.: User trends modeling for a content-based recommender system. Expert Systems with Applications **87**, 209–219 (2017). doi:10.1016/j.eswa.2017.06.020
13. Li, G., Zhang, Z., Wang, L., Chen, Q., Pan, J.: One-class collaborative filtering based on rating prediction and ranking prediction. Knowledge-Based Systems **124**, 46–54 (2017). doi:10.1016/j.knosys.2017.02.034
14. Wang, S., Huang, S., Liu, T.-Y., Ma, J., Chen, Z., Veijalainen, J.: Ranking-oriented collaborative filtering: A listwise approach. ACM Transactions on Information Systems **35**(2), 1–28 (2016). doi:10.1145/2960408
15. Karabadji, N.E.I., Beldjoudi, S., Seridi, H., Aridhi, S., Dhifli, W.: Improving memory-based user collaborative filtering with evolutionary multi-objective optimization. Expert Systems with Applications **98**, 153–165 (2018). doi:10.1016/j.eswa.2018.01.015
16. Patra, B.K., Launonen, R., Ollikainen, V., Nandi, S.: A new similarity measure using bhattacharyya coefficient for collaborative filtering in sparse data. Knowledge-Based Systems **82**, 163–177 (2015). doi:10.1016/j.knosys.2015.03.001
17. Ocepeka, U., Rugeljb, J., Bosnića, Z.: Improving matrix factorization recommendations for examples in cold start. Expert Systems with Applications **42**(19), 6784–6794 (2015). doi:10.1016/j.eswa.2015.04.071
18. Tran, C., Kim, J.Y., Shin, W.Y., Kim, S.W.: Clustering-based collaborative filtering using an incentivized/penalized user model. IEEE Access **7**, 62115–62125 (2019). doi:10.1109/ACCESS.2019.2914556
19. Bobadilla, J., Bojorque, R., Esteban, A.H., Hurtado, R.: Recommender systems clustering using bayesian non negative matrix factorization. IEEE Access **6**, 3549–3564 (2018). doi:10.1109/ACCESS.2017.2788138
20. Vander Aa, T., Chakroun, I., Haber, T.: Distributed bayesian probabilistic matrix factorization. In: Procedia of International Conference on Computational Science, ICCS, 12-14 June 2017, Zurich, Switzerland, pp. 1030–1039 (2017). doi:10.1016/j.procs.2017.05.009
21. Zhang, R., Mao, Y.: Movie recommendation via markovian factorization of matrix processes. IEEE Access **7**, 13189–13199 (2019). doi:10.1109/ACCESS.2019.2892289
22. Xian, Z., Li, Q., Li, G., Li, L.: New collaborative filtering algorithms based on svd++ and differential privacy. Mathematical Problems in Engineering **2017**, 1–14 (2017). doi:10.1155/2017/1975719
23. Guan, X., Li, C.T., Guan, Y.: Matrix factorization with rating completion: An enhanced svd model for collaborative filtering recommender systems. IEEE Access **5**, 27668–27678 (2017). doi:10.1109/ACCESS.2017.2772226
24. Yue, L., Sun, X.X., Gao, W.Z., Feng, G.Z., Zhang, B.Z.: Multiple auxiliary information based deep model for collaborative filtering. Journal of Computer Science and Technology **33**(4), 668–681 (2018). doi:10.1007/s11390-018-1848-x
25. Shams, B., Haratizadeh, S.: Item-based collaborative ranking. Knowle dge-Base d Systems journal **152**, 172–185 (2018). doi:10.1016/j.knosys.2018.04.012
26. Park, Y., Park, S., Jung, W., Lee, S.G.: Reversed cf: A fast collaborative filtering algorithm using a k-nearest neighbor graph. Expert Systems with Applications **42**(8), 4022–4028 (2015). doi:10.1016/j.eswa.2015.01.001
27. Polatidis, N., Georgiadis, C.K.: A multi-level collaborative filtering method that improves recommendations. Expert Systems with Applications **48**, 100–110 (2016). doi:10.1016/j.eswa.2015.11.023

28. Sun, S.B., Zhang, Z.H., Dong, X.L., Zhang, H.R., Li, T.J., Zhang, L., Min, F.: Integrating triangle and jaccard similarities for recommendation. Plos One **12**(8), 1–11 (2017). doi:10.1371/journal.pone.0183570
29. Wu, C., Wu, J., Luo, C., Wu, Q., Liu, C., Wu, Y., Yang, F.: Recommendation algorithm based on user score probability and project type. Eurasip Journal on Wireless Communications and Networking **2019**(80), 1–13 (2019). doi:10.1186/s13638-019-1385-5
30. Yu, P.: Collaborative filtering recommendation algorithm based on both user and item. In: Proceedings of 2015 4th International Conference on Computer Science and Network Technology, ICCSNT 2015, pp. 239–243 (2015). doi:10.1109/ICCSNT.2015.7490744
31. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. ACM Transactions on Interactive Intelligent Systems **5**(4) (2015). doi:10.1145/2827872
32. Jalili, M., Ahmadian, S., Izadi, M., Moradi, P., Salehi, M.: Evaluating collaborative filtering recommender algorithms: A survey. IEEE Access **6**, 74003–74024 (2018). doi:10.1109/ACCESS.2018.2883742
33. Zhang, F., Gong, T., Lee, V.E., Zhao, G., Rong, C., Qu, G.: Fast algorithms to evaluate collaborative filtering recommender systems. Knowledge-Based Systems **96**, 96–103 (2016). doi:10.1016/j.knosys.2015.12.025
34. Zheng, M., Min, F., Zhang, H.R., Chen, W.B.: Fast recommendations with the m-distance. IEEE Access **4**, 1464–1468 (2016). doi:10.1109/ACCESS.2016.2549182
35. Vellaichamy, V., Kalimuthu, V.: Hybrid collaborative movie recommender system using clustering and bat optimization. International Journal of Intelligent Engineering and Systems **10**(5), 38–47 (2017). doi:10.22266/ijies2017.1031.05
36. Fan, X., Chen, Z., Zhu, L., Liao, Z., Fu, B.: A novel hybrid similarity calculation model. Scientific Programming **2017**, 1–9 (2017). doi:10.1155/2017/4379141

Figures

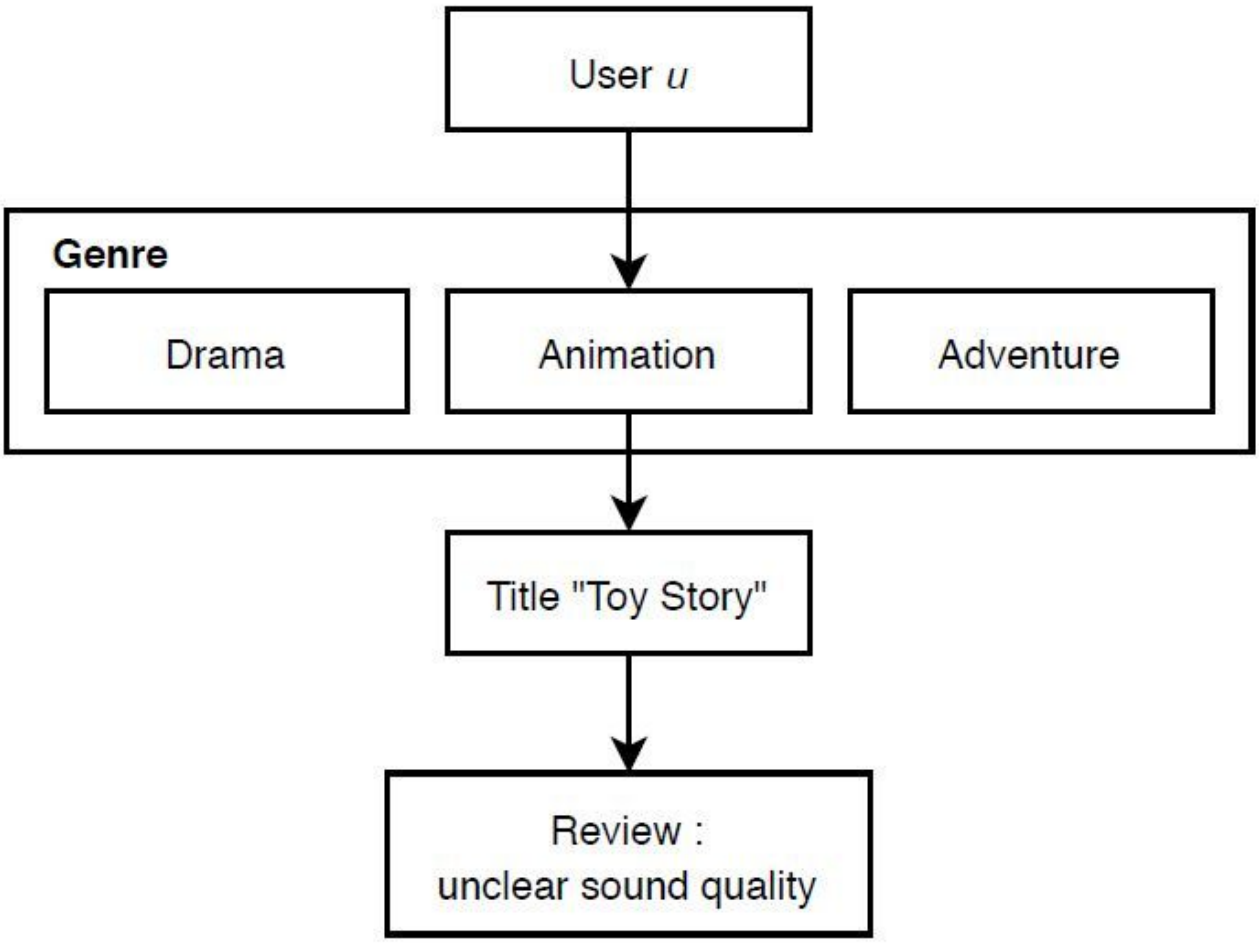

Figure 1

Example of user behavior in rating the item

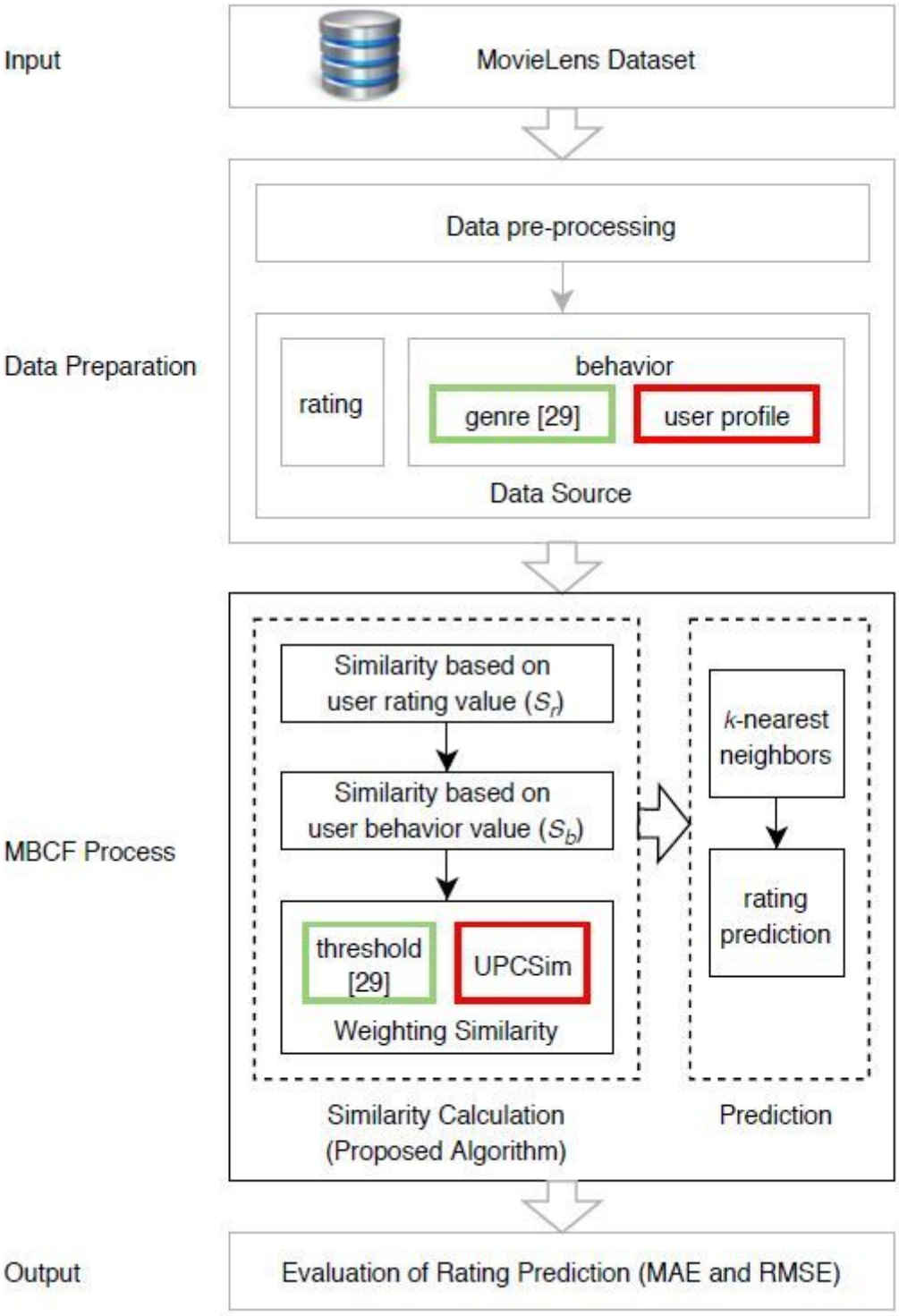

Figure 2

The developed MBCF system

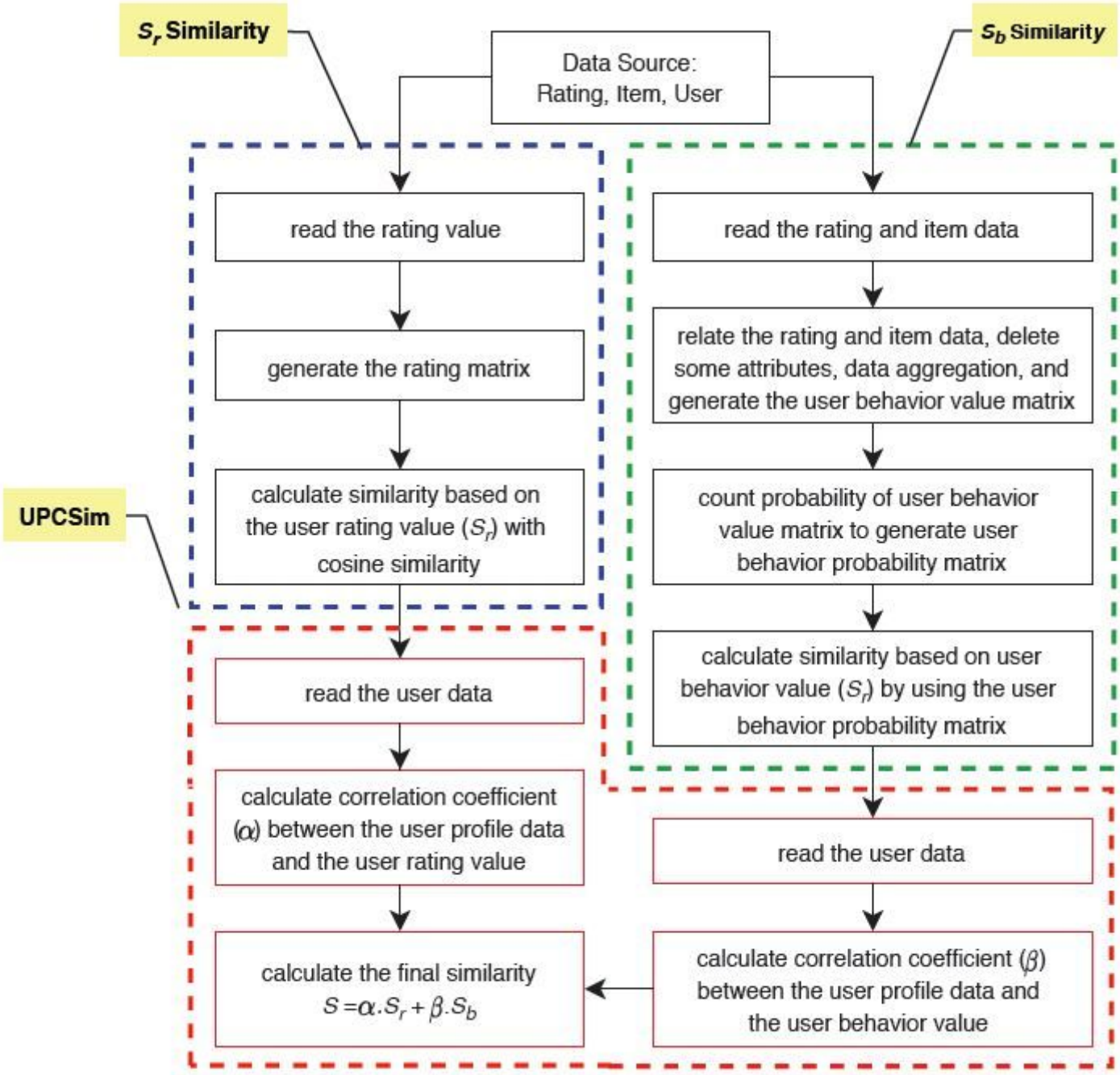

Figure 3

Flowchart of similarity calculation

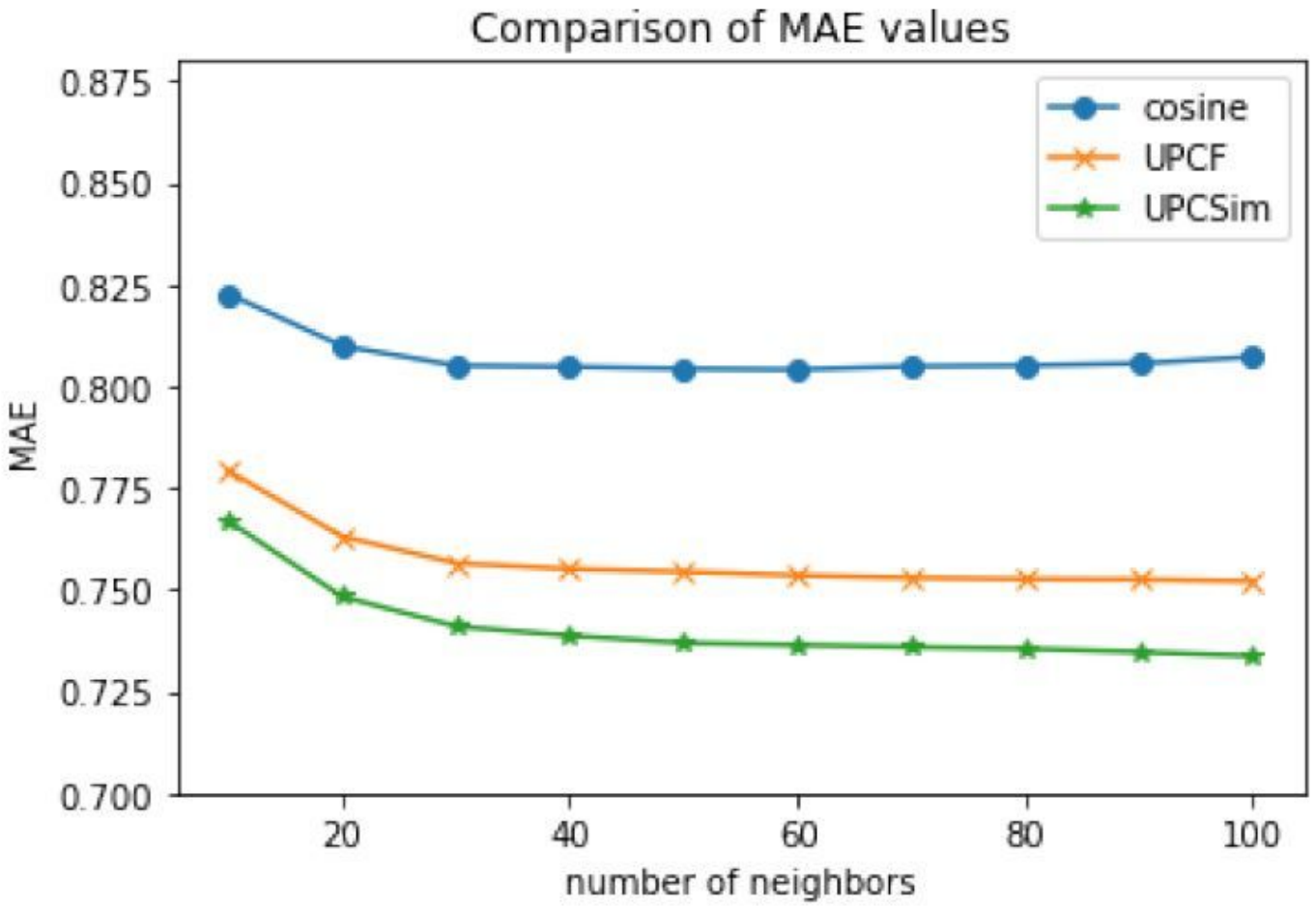

Figure 4

Comparison of the average MAE value of the three similarity algorithms using Movielens 100k dataset

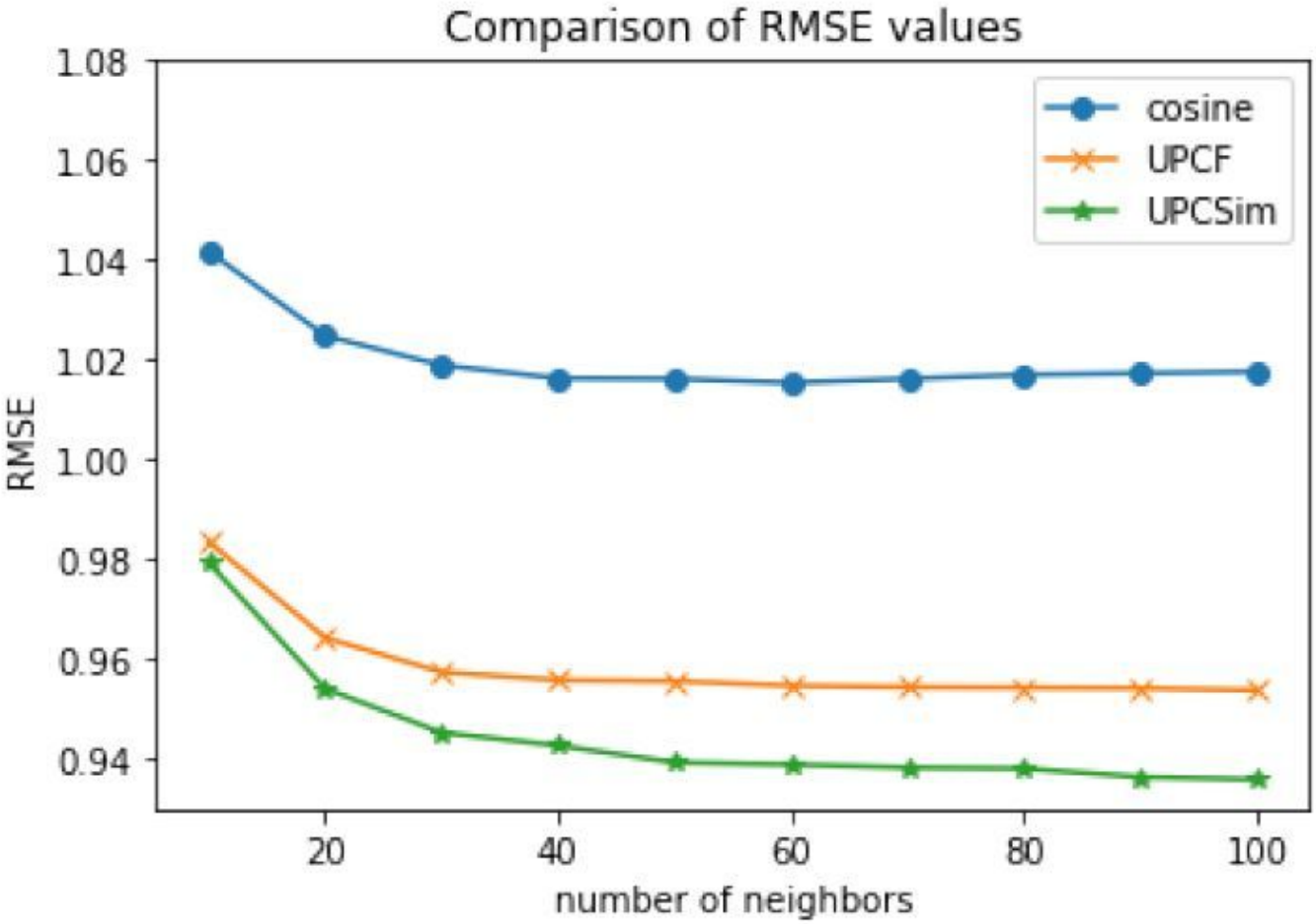

Figure 5

Comparison of the average RMSE value of the three similarity algorithms using Movielens 100k dataset