IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 30, 2021, accepted February 15, 2021, date of publication February 18, 2021, date of current version February 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3060016

Rating Prediction in Recommender Systems Based on User Behavior Probability and Complex Network Modeling

ZHAN SU[1], ZUYI LIN[1], JUN AI[1], AND HUI LI[2]
[1]School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China
[2]Information Science and Technology College, Dalian Maritime University, Dalian 116026, China

Corresponding authors: Zhan Su (suzhan@foxmail.com) and Jun Ai (aijun@outlook.com)

This work was supported by the Young Scientists Fund of the National Natural Science Foundation of China under Grant 61803264.

ABSTRACT In recommender systems, measuring user similarity is essential for predicting a user's ratings on items. Most traditional works calculate the similarity based on historical ratings shared between two users, without considering the probability of users' different behaviors. To address this issue, our work designs a similarity measure based on the users' historical behavior probabilities. Based on the similarity method, a complex network of user relationships is modeled. The user degree and community information of the modeled network, as well as the number of shared ratings between users, are used with the proposed similarity measure to design a rating prediction algorithm for recommender systems. Experiments based on MovieLens and Netflix data sets show that this method is effective and can successfully improve the accuracy of rating predictions and reduce the number of neighbors required to achieve the optimal prediction accuracy. Our research shows that in a complex system, the relationship between users can be effectively measured by the users' historical behavior probability, providing a new perspective for future research on similarity measurement.

INDEX TERMS Complex network modeling, recommender systems, degree centrality, link prediction, user behavior probability.

I. INTRODUCTION

Rating prediction is a very important part of modern recommendation algorithms. In the era of information overload, recommender systems (RS) and algorithms have become effective tools that can help companies and users find common interests. If the recommendation work was done properly, users' preferences would be automatically understood and predicted and the ''goods'' from merchants would be delivered to those who were truly interested, saving them their time and money. As a result, recommendation algorithms have been extensively researched and applied in various fields, such as web [1], music [2], and movie [3] recommendations.

In summary, traditional recommendation algorithms can be divided into three categories as follows [4], [5].

- Content-based (CB) recommendation models [6], [7]
- Collaborative filtering (CF) recommendation models [8]
- Hybrid recommendation models [9], [10]

CF models further include subcategories such as memory-based and model-based methods. On the other hand, content-based recommendation models use the content characteristics of the user's preferred items for analysis and recommendation, and hybrid models fully utilize both CF and CB methods to make predictions and recommendations [11], [12].

CF-related approaches rely on finding neighbors with similar historical ratings to the target user or item for rating predictions. Accordingly, there are various approaches proposed in the field to address this challenge.

For example, memory-based methods include CF based on the Pearson correlation coefficient [13], CF based on the cosine correlation coefficient [14], resource allocation CF [15], and multi-level CF [8]. Likewise, model-based approaches include user opinion spreading [16], modularized improved opinion spreading [17], Slope One [18], and so on.

The associate editor coordinating the review of this manuscript and approving it for publication was Moussa Ayyash.

Additionally, scientists utilized many other approaches for the improvement of recommender systems. In the work of [19], the authors proposed a reputation-based approach to improve trust-aware recommender systems by enhancing users' rating profiles with insufficient ratings and trust information. [20] proposed the Dempster–Shafer theory to incorporate implicit relationships to achieve the target. In the work of [21], authors used the Markov Decision Process to design a reinforcement learning model.

Likewise, a hybrid recommender system was proposed, which utilized a k-means clustering algorithm with bio-inspired artificial bee colony optimization technique [22]. The method can also be time-aware and with community considered [23], or combined neural network and matrix factorization techniques [24]. The work of [25] even introduced virtual ratings to improve prediction accuracy.

Researchers also explored other techniques in the field, such as reliability [26], points of interest [27], fuzzy logic [28], [29], complex network modeling [17], community detection [30], confidence measures [31], node centrality [32], vector similarity [33], and so on.

Also, since scientists revisited complex systems using network theory, various real-world systems have been modeled and analyzed by complex networks [34]–[38], including biological systems [39], social networks [40]–[45], semantic analysis [46], the Internet [47], link prediction [48], [49], software [50], [51], and recommender systems [17], [52].

Various techniques in complex network theory are beneficial for researching and improving the performance of recommender systems. For example, social relations [53] or trust [54] are two factors that scientists often use to enhance rating predictions and recommendation lists. Similarly, user-user network modeling based on similarity results is another convenient way to reduce the overall prediction error [32].

However, the similarity measurement in RS of user behavior probabilities has not been explored sufficiently in the existing literature. For example, user u and user v rated several books, some of which were rated by both of them. Traditionally, the similarity between u and v depends on the number of shared rated books and the statistical difference between their ratings [15].

If user u and user v have similar ratings for a set of items, it means that their preferences or tastes are very similar. Conversely, if the ratings of u and v are significantly different, it indicates that the preferences of u and v are inconsistent with each other. Therefore, user ratings are generally the primary indicator of their similarity, which is a general assumption of mainstream collaborative filtering algorithms [5], [55].

However, even if some items are rated negative, the user's behavior of rating the items can be considered a preference for such items, especially when they repeatedly rated items with similar attributes. By measuring the probabilities of all user behaviors, we can design similarity methods in novel ways.

For instance, user u and user v both rated on many horror books, and they usually have different opinions on the books. Both u and v have a 70% probability of reading a book with a horror tag, which means that although u and v have different ratings for shared rated books, they have the same preference for horror books. Therefore, the similarity based on their behavior probabilities is higher than the similarity calculated based on their shared ratings.

In summary, the existing literature has not yet fully explored how to measure the similarity based on user behavior probabilities. To solve this problem, our main contribution can be emphasized as follows.

1) We designed a method to calculate the similarity based on the user's selection behavior of different types of movies;
2) Our work constructed a complex network model to express the relationship between users based on the similarity between user behavior probabilities;
3) We proposed a method to predict ratings in RS based on degree and community information in the modeled network and the number of shared rated items.

Our study shows that the proposed similarity method and rating prediction algorithm effectively improve the accuracy of the rating prediction in the recommender systems and significantly reduce the required number of neighbors for the optimal prediction.

The rest of our study is organized as follows. Related works are reviewed in Section II. The proposed method is introduced in Section III, and section IV presents the experimental results by comparing our design to some state-of-the-art algorithms. Finally, we conclude our work in Section V.

II. RELATED WORKS

Evaluating users' preferences is a critical factor of CF recommendation algorithms and link prediction methods. Scientists in related fields have presented various approaches to measure similarity, including Pearson correlation coefficient [13], [56], Cosine-based similarity [14], Euclidean distance [57], mean squared distance [58], Jaccard [59], Spearman correlation [58], Bhattacharyya coefficient [60], [61], user opinion spreading [16], and so on. However, most methods measure similarity based on the correlated ratings between users. Although users' behavior have been considered differently [62], [63], there is no relevant published research regarding user behavior probability as a primary factor of similarity.

In addition to user behavior probability, our method also incorporates other factors inspired by recent literature. Based on the network of user similarities, we considered three factors, including the degree centrality, the community of user nodes, and the number of shared ratings used for each similarity calculation.

Jun Ai *et al.* first proposed modeling recommender systems by similarity results [30], providing extra information about node centrality and community. The authors designed a

complex network modeling method based on the similarity results between users in recommender systems, which use users as nodes and the similarity as the links.

Using similar results, the researchers can further reduce prediction errors by incorporating community [17], [64] and node centrality [32], [65] into predictions. The author used the constructed user-user network for community detection, and community information is beneficial for lowering rating prediction error. Similarly, the centrality [32] in this type of network indicates the popularity of the nodes. By reducing the weight of the popular node, we can effectively improve the performance in rating prediction.

Besides, the number of shared ratings is an essential factor for both cold-start problems [66] and the similarity calculation [67]. Su *et al.* [31] found that the number of shared ratings used to calculate similarity affects rating predictions. Therefore, the confidence coefficient [31] is also included in our research to improve the performance of the proposed method.

III. RATING PREDICTION BASED ON USER BEHAVIOR PROBABILITY AND COMPLEX NETWORK MODELING

In this section, a similarity method based on the probabilities of user behavior is proposed first. Subsequently, three other factors, including the confidence coefficient, node degree, and community information, are introduced to improve prediction accuracy further.

A. SIMILARITY BASED ON USER BEHAVIOR PROBABILITY

In recommender systems, a user u has a behavior $\tilde{b}$ on item i, in which a tag in a set L can label $\tilde{b}$. Similarly, user u has a behavior $\tilde{b}$ interacting with user v, and a tag in the set L can mark $\tilde{b}$.

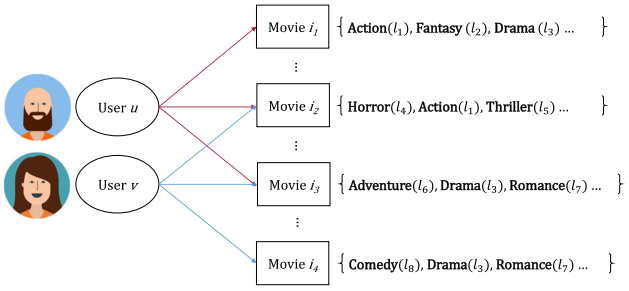
Therefore, we can define a behavior set for user u, and construct a behavior sequence $\tilde{L} = \{l_1, l_2, l_3, \ldots, l_e\}$ corresponding to each action the user u has taken. The higher the probability that the user u took a specific behavior, the more the user prefers that type of behavior.

In our work, we focus on users' choices on items in recommender systems. Thus, the set $\tilde{L}$ presents user rating action on different genre labels of movies, such as Romance, Sci-Fi, Horror, Drama, etc. Moreover, movies typically include more than one genre label. For instance, *Tenet* (2020) directed by Christopher Nolan has two genre labels: Action and Sci-Fi. It is also worth noting that users' ratings don't affect the probability of their selection of movies.

On this basis, labels of all possible user behaviors in the specific system can be denoted as $\hat{L} = \{l_1, l_2, \ldots, l_z\}$. It worth noting that the set $\hat{L}$ has a fixed size z for a given system. There are only eighteen labels standing for movie genres in a movie recommender system.

Therefore, we can analyze user preferences based on the probability of their behavior. The preferred ones reveal their preferences despite their rating scores.

Even if a user's ratings on horror movies are low, it is reasonable to assume that the user strongly prefers horror movies when 90% of his ratings given on horror-related items. Therefore, the problem to be solved is how to measure the probability of user behavior and evaluate the similarity between different users' behavior probabilities.



	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9	l_{10}	...	total
User u	25	10	2	1	13	1	1	0	0	0	...	x=80
User v	1	0	2	1	1	3	10	5	1	1	...	y
p_{ug}	$25/x$	$10/x$	$2/x$	$1/x$	$13/x$	$1/x$	$1/x$	0	0	0	...	1
p_{vg}	$1/y$	0	$2/x$	$1/y$	$1/y$	$3/y$	$10/y$	$5/y$	$1/y$	$1/y$	...	1

$P_u = [p_{ul1} \quad p_{ul2} \quad p_{ul3} \quad p_{ul4} \quad \ldots]^T = [0.3125 \quad 0.125 \quad 0.025 \quad 0.0125 \quad \ldots]^T$

FIGURE 1. The figure shows how to calculate the user behavior probability for user u, who rated several movies with different genre labels (from l_1 to l_{10}). First, we can count the number of times user u have rated a specific genre of movie. For instance, user u rated 80 ($x = 80$) movies and 25 of them are with "Action" genre label (l_1). The historical probability that user u rated a movie with genre g can be obtained as $p_{ug} = p_{ul1} = 25/80$. Thus, we have user behavior probabilities P_u of user u standing for probabilities on rating different labels.

Fig.1 shows an example on how to calculate user behavior probability P_u for the probabilities of user u on different genres. Moreover, P_u consists of a set of probabilities, each representing the probability of user u's selection of a specific sub-type of behavior in his historical actions.

Subsequently, we can use the Pearson correlation coefficient to calculate the similarity between any P_u and P_v, which represents the behavior correlation between the two users u and v.

Thus, the probability that user u has a behavior with label l_g is defined by Equ. (1),

$$P_{ug} = \frac{\sum_{\tilde{b} \in \tilde{L}} \left(\sum_{l_q \in \hat{L}} \delta \left(l_q, l_g \right) \right)}{|\tilde{L}|}, \tag{1}$$

where $\tilde{L}$ is a set of behaviors $\tilde{b}$ took by user u on items with a genre label g, and $\tilde{L}$ is a subset of $\hat{L}$. The Kronecker δ function $\delta \left(l_q, l_g \right)$ is defined as follows:

$$\delta(l_q, l_g) = \begin{cases} 1, & l_q = l_g, \\ 0, & l_q \neq l_g \end{cases} \tag{2}$$

According to the definition, the probability of a user u rating an item labeled with l_g represents the ratio of the user u choosing to rate the item labeled with l_g to the total number of ratings given by the user. Statistically, the higher P_{ug} is, the more preferable the items with label l_g are for the user u. The Equ. (1) and Equ. (2) do not consider the user's specific rating scores.

For $\hat{L} = \{l_1, l_2, \ldots, l_z\}$, each user u in the system has a vector that $P_u = [P_{u1}, P_{u2}, \ldots P_{ug} \ldots, P_{uz}]$, which measures the probabilities of user u choosing different genres based on historical data. When user has not rated any item with label l_g, the corresponding $P_{ug} = 0$ in the vector. Algorithm 1 specifies the calculation of the user behavior probability.

Algorithm 1 Calculation of User Behavior Probability

Input: User ratings set R_u, behavior label set L, item set I (with labels).
Output: User behavior probability P_u.
$|\tilde{L}|$ = the number of rating behaviors by u
for q from 1 to z, $l_q \in L$ **do**
$\sum l_q$ = the number of behaviors labeled as l_q by u
$P_u[q] \leftarrow \frac{\sum l_q}{|\tilde{L}|}$
$q \leftarrow q + 1$
end for
return User behavior probability P_u of user u

Thus, Equ. 4 is the proposed similarity based on user behavior probability (**UBP**), in which the similarity between the behavior probabilities of user u and v is evaluated by Pearson correlation coefficient in Equ. 3.

$$PCC(U, V) = \frac{\sum_i^n [(U_i - \bar{U}) \cdot (V_i - \bar{V})]}{\sqrt{\sum_i^n (U_i - \bar{U})^2} \cdot \sqrt{\sum_i^n (V_i - \bar{V})^2}}, \quad (3)$$

$$s_{uv} = PCC(P_u, P_v), \quad (4)$$

where $\bar{U}$ and $\bar{V}$ are the average of U and V, respectively. Specifically, $U = P_u$ and $V = P_v$ are used for the similarity computation based on user behavior probability.

B. CONFIDENCE COEFFICIENT, DEGREE AND COMMUNITY

Firstly, confidence coefficient [31] is defined by Equ.5.

$$CC(|T_{uv}|) = \begin{cases} e^{-(5-|T_{uv}|)}, & 0 \le |T_{uv}| \le 5 \\ \ln(|T_{uv}| - 5) + 1, & |T_{uv}| > 5, \end{cases} \quad (5)$$

where T_{uv} is the set of shared rated items by both user u and v, and $|T_{uv}|$ the the number of items in the set.

We combine Equ. 4 and Equ. 5 to further consider the similarity as follows.

$$\hat{s}_{uv} = CC(T_{uv}) \cdot PCC(P_u, P_v) \cdot PCC(R_u, R_v), \quad (6)$$

where P_u and P_v is calculated by Algorithm 1. R_u, R_v are scores on shared rated items given by both user u and v, respectively.

Secondly, when users in the system are considered nodes, the similarities between users are considered links. We model a network based on the similarity of user preferences in the recommender system. The core issue of modeling the complex network based on user similarities is how to remove redundant links. Similarity exists between all pairs of users, making users fully connected to all other nodes in the network.

As such, links with low weight (low similarity) need to be removed to ensure that the network structure is analyzable. It is also necessary to ensure that the network topology stays as connected as possible so that the number of isolated nodes is small [17]. Algorithm 2 provides the process for complex network modeling.

Algorithm 2 Modeling Complex Network Based on User Similarities

Input: User node set U, similarity results S, parameters k and p.
Output: Network N contains all users U as nodes and remained links E defined by similarity.
for all user u in U **do**
S_u = all similarity results related to user u, as $S_u \subset S$
for j from 1 to p **do**
Add the maximum similarity s_{uv} in S_u to E_r, consider it as a link between u and v with similarity as the link weight;
Remove s_{uv} from S_u;
Remove s_{uv} from S;
$j \leftarrow j + 1$;
end for
end for
for j from 1 to $(kn - pn)$ **do**
Add the maximum similarity s_{uv} in S to E_r, consider it as a link between u and v with similarity as the link weight;
Remove s_{uv} from S;
$j \leftarrow j + 1$;
end for
$N = U$;
$E = E_r$;
return Network with nodes N and links E.

The parameter k is significant for the modeling because k controls the number of links in the network with node number fixed at n. Fig.3 showed an example network based on the MovieLens data set, revealing a user-user similarity relationship in the recommender system. The example network is constructed with $k = 1.5$, and $k = 3.0$ is used in other experiments, determined by experiments to achieve the optimal prediction accuracy.

According to existing research [32], we need to reduce the weight of high-degree nodes (larger nodes in Fig.2) and neighbors' weight in a different community [17] (different colored nodes in Fig.2) with the prediction target user.

C. RATING PREDICTION BASED ON USER BEHAVIOR PROBABILITY

In Fig.2, users have different statuses, and the degree centrality indicates their popularity. If the user has a higher degree, their similarity to others is relatively more remarkable than that of a lower degree. The reason is that high-degree users rate items more than low-degree users and their ratings are closer to the majority.

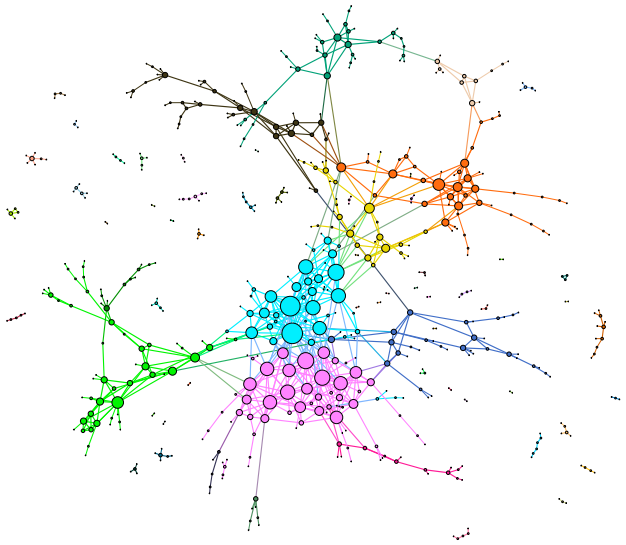


FIGURE 2. Fig.2 shows a modeled complex network example based on UBP similarity. The nodes and links in the topology stand for users and their similarity, respectively. Colors indicate the communities in the network, and node size stands for their degree centrality. The result is constructed in one of the 10-fold cross-validations, using 90% of ratings and all users as the data source.

In the work of [32], authors found that the degree value of the similarity network has a negative effect on the rating prediction. The higher the degree value, the smaller the weight in the prediction equation should be to obtain a more accurate rating prediction.

Therefore, we propose the following equation to predict ratings based on user behavior probability, confidence coefficient, node degree, and community information.

$$\hat{r}_{ui} = \overline{r}_u + \frac{\sum_{v=1}^{n} \frac{(r_{vi}-\overline{r}_v)\cdot\hat{s}_{vu}\cdot\omega(uv)}{Degree^*(v)+1}}{\sum_{v=1}^{n}|\hat{s}_{vu}|} \tag{7}$$

where $\hat{s}_{vu}$ is the UBP similarity between user v and u, $\overline{r}_v$ is the average rating given by user v. The equation selects n nearest users of user u for the prediction, and $Degree^*(v)$ is the normalized degree of user v in the modeled user-user similarity network. $\omega(uv)$ is the community parameter and is defined as follow.

$$\omega(uv) = \begin{cases} \omega_1 \in [1,2], & c_u = c_v \\ \omega_2 \in (0,1], & c_u \neq c_v, \end{cases} \tag{8}$$

which is used to adjust the impact of neighbors, making neighbors in the same community more critical, and neighbors in other communities less critical in the rating prediction. Respectively, c_u and c_v stands for user u's and user v's community, labeled by a community detection algorithm [68]. In our work, we selected $\omega_1 = 1.0$ and $\omega_2 = 0.7$ by experiments, which is omitted in the paper for the sake of clarity.

IV. EXPERIMENTAL ANALYSIS AND RESULTS

Experiments confirm the effectiveness of our approach in this section. We compared the proposed user behavior probability CF (**UBP**) with several state-of-the-art algorithms available in the recent literature.

A. BENCHMARK ALGORITHMS

Besides **UBP**, we also selected the following benchmark algorithms for experimental comparison in our work.

1) Slope one (**Slope One**, 2005) is proposed in 2005 by Lemire and Maclachlan [18] that pre-computes the average difference between the ratings of two items for users who have rated both items. The difference results are used as the distance between the prediction target and known ratings.
2) The average ratings (**AverageRatings**) of users can also predict unknown ratings [30], and we use it as a benchmark baseline. It is fast and simple, revealing the worst-case scenario for prediction performance.
3) User opinion spreading (**UOS-opt**, 2015) suggests that opinions spread in the recommender system [16], [69], which causes one user's ratings to affect others in the system. The approach measures the similarities between users based on such spreading of user opinions. Moreover, introducing lambda ($\lambda = 0.5$ in our experiments) to the similarity equation can further improve the performance of the method.
4) Cosine correlation coefficient (**Cosine**, 2001) is a popular measure in user-based collaborative filtering [14].
5) Bhattacharyya coefficient (**Bhattacharyya**, 2020) is proposed in recent work by Bidyut Kr. Patra *et al.*, which uses all ratings made by a pair of users instead of only ratings of co-rated items [60].
6) Modularized Improved Opinion Spreading (**MIOS**, 2019) is designed to research the impact of community in link prediction [17]. Parameters $\omega_1 = 1.0$, $\omega_2 = 0.7$ and $k = 3$ are used in our experiments.
7) A Multi-Level Collaborative Filtering (**MLCF**, 2016) is proposed to improve the accuracy of the classical CF [8].

The authors implement the selected algorithm, and all possible parameters used in the experiment are set to be the same for comparison. Otherwise, the values in the original paper are used.

B. EVALUATION CRITERIA

We evaluated the system using the accuracy of rating prediction and the ranking performance of the recommendation list. To evaluate the prediction accuracy, we use mean absolute error (MAE) and root mean square error (RMSE) simultaneously, and they are widely used in related fields [5]. They are defined as follows.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|r_{ui} - \hat{r}_{ui}| \tag{9}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_{ui} - \hat{r}_{ui})^2} \tag{10}$$

where test set has n ratings of users on items, $\hat{r}_{ui}$ is the predicted rating and r_{ui} is the real rating in the test data set.

We gradually increased the number of neighbors from 1 to 120 and compared the MAE and RMSE results of different methods. The metrics compare the actual ratings given by the user with the predicted ratings assigned by the algorithms. The smaller the MAE and RMSE are, the more accurate the algorithm is. Because RMSE squares the error before summing, the result reveals larger errors more obvious than MAE.

On the other hand, we select normalized discounted cumulative gain (NDCG) and half-life utility (HLU) for ranking performance of the recommendation lists. They are defined by Equ.5 and Equ.6.

Discounted cumulative gain evaluates the usefulness of an item based on its rank in a recommendation list. The more relevant items are with higher ranks, the more valuable the recommendation list is for the user [70].

$$DCG(u) = \sum_{i=1}^{b} R_i + \sum_{i=b+1}^{N} \frac{R_i}{\log_b r_i} \tag{11}$$

$$NDCG = \frac{\sum_{u=1}^{m} \frac{DCG(u)}{\max(DCG(u))}}{m} \tag{12}$$

where DCG is calculated for user u according to u's real and predicted lists. r_i is the rank of the item in the list. b is a parameter to enhance the influence of important items and control the degree of item reduction. We used $b = 2$ in our experiments according to [70]. R_i stands for the relevance of the recommended item, $R_i = 1$ if the real rating of a recommended item is larger than the user's average rating, and $R_i = 0$ if otherwise.

Moreover, normalized DCG can be calculated based on the maximal of DCG. Thus, the average of normalized DCG for all users in the testing set indicates the ranking performance [70].

Likewise, half-life utility (HLU) measures the utility of a recommendation list based on a hypothesis that as the item rank in the recommendation list decreases, the probability of user's tendency to examine the item reduces exponentially [71]. The half-life utility is defined as follows.

$$HLU = \frac{\sum_{i=1}^{N} \frac{\max(r_{ui} - d, 0)}{2^{(i-1)/(h-1)}}}{m} \tag{13}$$

where h is the half-life threshold and d is a parameter indicating users' neutral vote. In our experiments, d is set as the average rating of the user and $h = 2$. The average of all half life results for users is the score. The greater the value of HLU is, the better the recommendation list is.

C. DATA SETS

MovieLens[1] and Netflix[2] are two well-known data sets that are often used in the field to verify proposed methods [72], [73], and both data sets are used in our work.

[1]https://grouplens.org/datasets/movielens/, ml-latest-small, updated on 9/2018

[2]https://www.kaggle.com/netflix-inc/netflix-prize-data

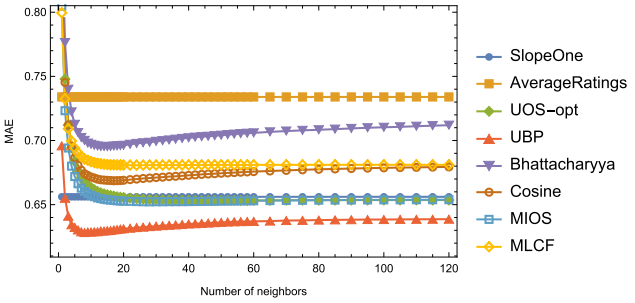


FIGURE 3. MAE (the smaller, the better) results of different methods against the number of top-k neighbors used in the prediction on MovieLens data set. The proposed UBP reached the optimal accuracy at $k = 11$ by contrast to UOS-opt reaching the optimal at $k = 28$, the error is reduced by 2.72% and the required number of users for the optimal is reduced by 61%.

The MovieLens data set used in our work contains 671 users, 9125 movies, and 100,000 ratings. It is named "ml-latest-small" by the provider and can be found by the footnote URL. On the other hand, the Netflix data set contains more than 59 million ratings, in which we randomly pick 2000 users for our experiments. Those users rated 3101 movies, and the total number of selected ratings is 237,059.

Because the Netfilx data doesn't contain information on movie genres, the movie titles are used to find their corresponding genres in the MovieLens data set. Therefore, we only use part of the movie data in Netflix, which genres can be found in MovieLens.

The rating values are integer numbers in the range of 1 (bad) to 5 (excellent). Applying the proposed method and other benchmark approaches on the data sets presents us with the evaluation of the algorithms' overall performance.

Additionally, 10-fold cross-validation is applied, by which the data set is randomly divided into ten subsets. Each subset is used as testing data, and the other nine subsets are used for the corresponding training. Algorithms predict all ratings in each testing set based on only the training set. The final results are the average of all experiments.

D. RATING PREDICTION ON MovieLens

The results of MAE and RMSE on MovieLens data set are reported in Fig.3 and Fig.4, respectively.

MAE and RMSE reveal the general error status of rating prediction. The smaller the values are, the better the prediction performance is. Our proposed UBP method produces smaller errors compared to the other benchmark algorithms that use state-of-the-art similarity metrics. UBP not only has smaller errors in the prediction but also requires fewer neighbors for the optimal prediction.

Moreover, UOS-opt is the closest competitor on the MoveiLens data set, and Slope One is the third close to UOS-opt. Slope One needs all possible neighbors for the prediction. Therefore, the MAE and RMSE of Slope One doesn't change against the number of neighbors in both figures. UBP has the superiority of MAE 0.628 and RMSE 0.851 at only

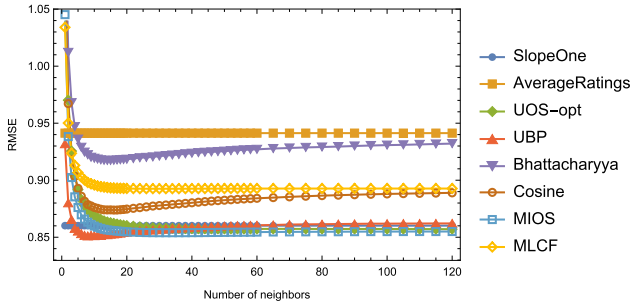


FIGURE 4. RMSE (the smaller, the better) results of different methods against the number of top-k neighbors used in the prediction on MovieLens data set. The proposed UBP reached the optimal accuracy at $k = 11$ by contrast to UOS-opt reaching the optimal at $k = 28$, the error is reduced by 2.76% and the required number of users for the optimal is reduced by 61%.

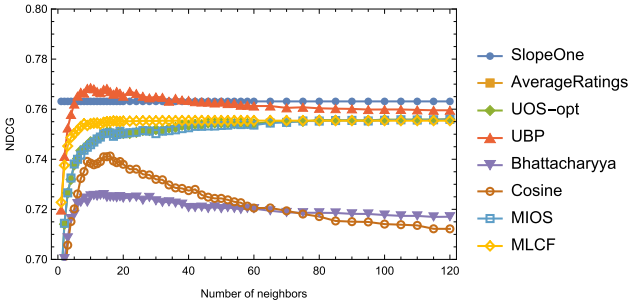


FIGURE 5. NDCG (the larger, the better) results of different methods on the y-axis against the number of top-k neighbors used in the prediction on the x-axis (MovieLens). The proposed UBP reached the optimal ranking at $k = 10$ in contrast to COS reaching the optimal at $k = 16$. The ranking performance is improved by 3.0% to 8.8%. Slope One has the second-best ranking performance in NDCG, but the method uses all possible neighbors ($k \geq 120$) instead of top-k the traditional CF-based approaches.

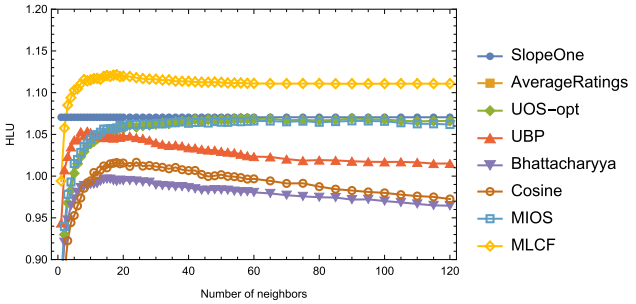


FIGURE 6. HLU (the larger, the better) results of different methods on the y-axis against the number of top-k neighbors used in the prediction on the x-axis (MovieLens). The proposed UBP reached the optimal ranking at $k = 9$ in contrast to Bhattacharyya reaching the optimal at $k = 15$, the change of ranking fluctuates from −6.0% to 5.76% and the required number of users for the optimal is reduced by 41% to 93%.

eight neighbors in optimal prediction accuracy. Compared to UOS-opt, the UBP reduces MAE by 3.826% and RMSE by 0.692%, respectively.

The percentage of improvement in MAE is larger than the one in RMSE. Based on our understanding, the proposed UBP similarity selects the top 20 or so neighbors more accurately and reduce the general errors in rating prediction. However, UBP does not significantly reduce the large errors in the prediction, shown in the results of RMSE. UBP makes small errors smaller, but the larger errors in prediction stay. We couldn't find evidence that suggests those larger errors get worse compared to other benchmark methods.

In a word, UBP has the best prediction accuracy on MovieLens data set in our experiments.

E. RANKING PERFORMANCE ON MovieLens

Fig.5 and Fig.6 show the result comparison of NDCG and HLU on MovieLens data set, respectively. The results showed that the proposed UBP only holds fifth place in HLU. But UBP has the best performance in NDCG, with Slope One running close. Due to the relevant recommended items requiring the predicted rating greater than the average user rating, the AverageRatings method provides no items that meet the requirement. We omitted AverageRatings in both figures.

Both NDCG and HUL depend on the ranking order of the recommendation list, but HLU tends to evaluate the ranking based on ratings of the whole recommendation list. Thus, the underperformance of UBP on HLU indicates our method does not perform well in putting the highest-rating items at the top of the list. According to NDCG, UBP measures the overall relevance of ranking more accurately than others. UBP increases the NDCG by 1.67% and 5.89% compared to UOS-opt and Bhattacharyya, respectively.

In summary, UBP performs best at the overall ranking relevance of the recommendation list. Still, its performance is moderate on putting the highest-rating items at the top of the list.

F. THE REQUIRED NUMBER OF NEIGHBORS FOR THE OPTIMAL RESULTS

To further observe the number of neighbors required by each algorithm to achieve its optimal performance, the result of the number of neighbors needed on the MovieLens data set is shown in Fig.7 and Fig.8, respectively.

Fig.7 shows the required number of neighbors for MAE and RMSE, and the model-based Slope One needs all possible neighbors for the prediction. By contrast, the AverageRatings method doesn't need neighbors at all (omitted in Fig.7 and Fig.8). The proposed UBP reduces the required user number for optimal accuracy by 41% to 93%, which is also superior on MovieLens data set compared to all others except AverageRatings.

Because the number of neighbors required to obtain the optimal performance affects the computational complexity, and more neighbors need more disk space to store the k-nearest neighbors of all users in the online recommendation (usually as a cache in recommender systems). Thus, the fewer neighbors the algorithm requires, the better the recommendation algorithm.

The proposed UBP only needs eight neighbors to get the optimal accuracy, while UBP only needs 10 and 9 neighbors to achieve optimal ranking performance.

Therefore, the results in Fig.7 show UBP need much fewer neighbors with a more accurate prediction. Likewise, the results of HLU and NDCG show UBP is a better choice

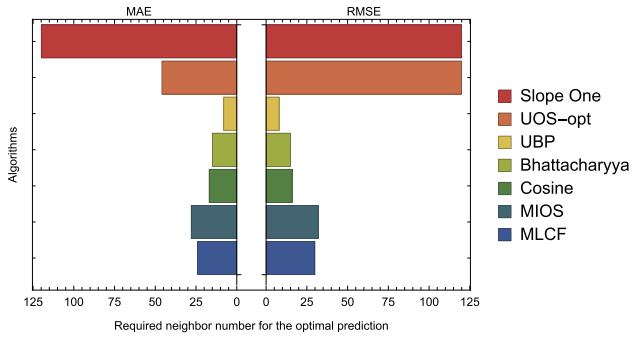


FIGURE 7. The figure shows the required number (the smaller, the better) of top-k neighbors for the optimal prediction performance of MAE and RMSE on MovieLens experiments. The proposed UBP needs only $k = 8$ neighbors for the optimal prediction on both MAE and RMSE.

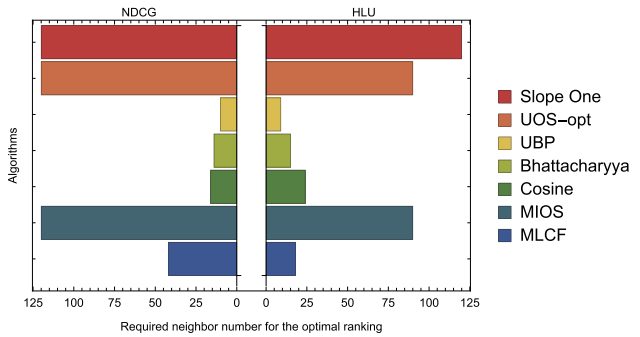


FIGURE 8. The figure shows the required number (the smaller, the better) of top-k neighbors for the optimal prediction performance of NDCG and HLF on MovieLens experiments. The proposed UBP needs only $k = 10$ and $k = 9$ neighbors for the optimal ranking on NDCG and HLF, respectively.

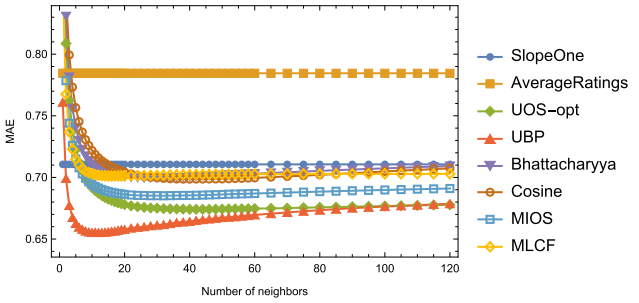


FIGURE 9. MAE (the smaller, the better) results of different methods against the number of top-k neighbors used in predicting the Netflix data set. The proposed UBP reached the optimal accuracy at $k = 12$ in contrast to the second-best UOS-opt achieving the optimal at $k = 44$, the prediction performance of UBP is improved by 3.8% than UOS-opt, and the required number of users for the optimal is reduced by 72.7%.

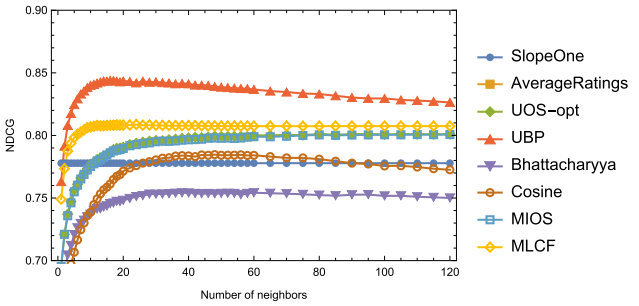


FIGURE 10. NDCG (the larger, the better) results of different methods against the number of top-k neighbors used in predicting the Netflix data set. The proposed UBP reached the optimal ranking at $k = 16$ in contrast to MLCF getting optimal at $k = 24$, the ranking performance is improved by 4.35%, and the required number of users for the optimal is reduced by 33.3%.

considering the required neighbor numbers for the ranking of recommendation in Fig.8.

However, Fig.8 indicates only the number of required neighbors when each algorithm reaches its optimum. Methods like MIOS and UOS-opt keep increasing their scores and stay at high performance after 20 neighbors. By contrast, algorithms like MLCF, UBP, and Cosine reach a maximum performance at specific points, and then their performance decline.

In summary, the proposed UBP reduces the required number of neighbors by 41% to 93% in accuracy and ranking performance, indicating that UBP neighbor selection is more accurate.

G. PREDICTION AND RANKING PERFORMANCE ON NETFLIX

Similarly, we want to know the performance of UBP on other data sets. Therefore, Fig.9 and Fig.10 shows MAE and NDCG result comparison on the subset of Netflix data set, respectively. Since RMSE and HLU have similar patterns like those on MovieLens, they are omitted for clarity.

Experiments have confirmed that UBP provides excellent performance in the prediction accuracy and ranking order of the recommendation list. Besides, compared with the results on MovieLens, the prediction accuracy of UOS-opt on Netflix is significantly reduced, indicating that the stability of others is not as good as the proposed UBP.

To summarize, UBP still works well and maintains its advantage over others, showing the performance of UBP is very stable on different data sets.

H. GENERAL IMPROVEMENT OF THE PROPOSED METHOD

To generally evaluate the improvement of the proposed method on both data sets, we calculated the average improvement of UBP on MovieLens and Netflix, comparing UBP to each of the benchmark methods.

In summary, the prediction accuracy of UBP is improved by an average of 2.25% to 15.39%, which shows that the prediction accuracy of UBP is better and more stable. Similarly, according to the results of NDCG, UBP's average improvement in ranking performance ranges from 2.56% to 9.15% on NDCG, which indicates that UBP's ranking is generally better. Still, the ranking order of the recommendation list is not as good as UOS-opt, MIOS, and MLCF (according to HLU).

I. TIME CONSUMPTION OF ALGORITHMS

Finally, Table 1 presents the time consumption of all used algorithms in experiments. We used a computer with Intel(R)

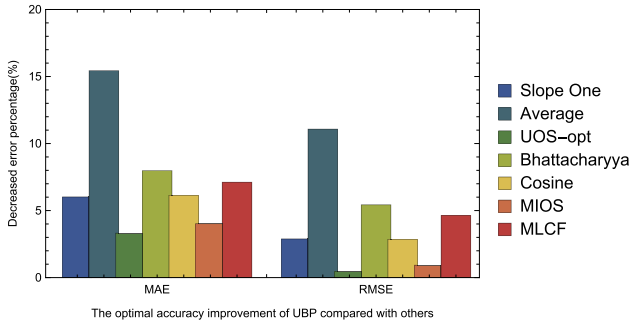


FIGURE 11. The general prediction improvement (the larger, the better) of UBP compared with each of the benchmark algorithms on MovieLens and Netflix. The average improvement is 7.144% and 4.04% on MAE and RMSE, respectively.

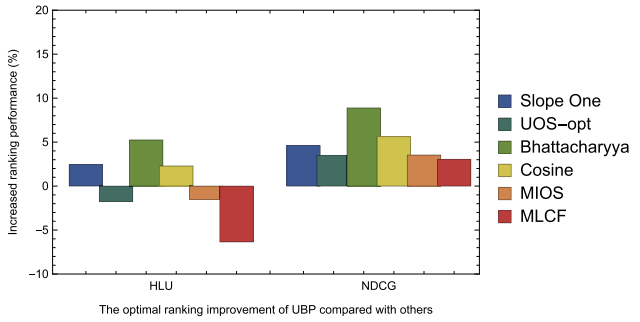


FIGURE 12. The general prediction improvement (the larger, the better) of UBP compared with each of the benchmark algorithms on MovieLens and Netflix. The average improvement is 4.86% and 0.06% on NDCG and HLU, respectively.

TABLE 1. Compare the proposed algorithm's time consumption to the benchmark algorithms on the MovieLens and Netflix data sets.

Algorithms	MovieLens(min.)	Netflix(min.)	Total(min.)
UBP	6.527	12.87	19.40
Cosine	2.746	5.021	7.767
Bhattacharyya	23.10	207.4	230.5
MIOS	5.716	11.23	16.946
UOS-opt	2.982	5.537	8.519
SlopeOne	1.324	4.499	5.824
Average	0.082	0.1155	0.1978

Core i9-9900K CPU @ 4.5GHz, 32GB of main memory @ 4000MHz, a solid-state drive of Samsung 970 evo 1Tb, and FSharp language implementation with.NET 5.0 on a Windows 10 x64 operating system. All the possible parameters affecting the computation time were set the same for all algorithms.

The results show that, compared with UOS-opt, the computational complexity of UBP has doubled and is almost the same as MIOS. This method's main added complexity is the community detection based on the modularity Q in the constructed similarity network. The running time of UBP without node centrality and community is similar to UOS-opt. However, this will result in decreased prediction accuracy and make the method unstable. Our community detection implementation based on modularity Q is run by only one CPU core. With further optimization of the code in future work, the time consumption is expected to be reduced.

V. CONCLUSION AND FUTURE WORKS

This paper proposes a novel recommendation method that can effectively consider user behavior probabilities in the recommendation process. As such, our work proposes a UBP similarity that considers the probability of users choosing different genres as an essential recommendation factor. The method combines user behavior probability, degree centrality, community, and confidence coefficient, provides more accurate rating prediction and a more preferred ranking list, but requires fewer neighbors.

The experimental results confirm the effectiveness of our method. Compared with state-of-the-art methods on MovieLens and Netfilx, the accuracy is improved by 2.25% to 15.39%, while the required number of neighbors is reduced by 41% to 93%. Similarly, compared to the latest methods on MovieLens and Netflix, the ranking performance has increased by 2.56% to 9.15%.

This research can be extended in three directions. First, the time consumption of UBP is an essential factor in the development of a successful recommender system. Through further code optimization, the computational complexity needs to be reduced in the future. Secondly, UBP's ranking performance in HLU is not ideal, and further research is needed. The mechanism of UBP underperformance can be explored to improve ranking performance further. Third, the constructed similarity network for node degree and community calculation can be analyzed in more detail, which could further improve the accuracy of the system.

In summary, the main contributions in our work can be summarized as follows:

1) We proposed a recommendation algorithm that combines user behavior probability and complex network modeling, which provides reliable and accurate rating prediction, more preferred recommendation list with fewer required neighbors to achieve the optimal performance.
2) Our research suggests that the probability of a user choosing a particular type of item indicates their preference, regardless of their ratings on these items.
3) The constructed similarity network that provides node degree and community information proves to be a useful method to improve the proposed method's performance.

ACKNOWLEDGMENT

(Zhan Su, Zuyi Lin, and Jun Ai contributed equally to this work.) Zhan Su and Jun Ai would like to express their love and gratitude to Lingyi Ai, daughter of Zhan Su, for giving them the courage to carry on and fight.

The authors would like to thank the anonymous reviewers for their valuable remarks and comments, which significantly contributed to the quality of the paper. They also consider the anonymous reviewers as anonymous coauthors of their work.

REFERENCES

[1] R. Katarya, "An improved recommender system using two-level matrix factorization for product ontologies," in *Proc. Int. Conf. Intell. Sustain. Syst. (ICISS)*, Dec. 2017, pp. 223–226.

[2] R. Katarya and O. P. Verma, "Recommender system with grey wolf optimizer and FCM," *Neural Comput. Appl.*, vol. 30, no. 5, pp. 1679–1687, Sep. 2018.
[3] R. Katarya, *Reliable Recommender System Using Improved Collaborative Filtering Technique*. Boca Raton, FL, USA: CRC Press, Sep. 2018.
[4] L. Lü, D.-B. Chen, and T. Zhou, "The small world yields the most effective information spreading," *New J. Phys.*, vol. 13, no. 12, Dec. 2011, Art. no. 123005.
[5] M. Jalili, S. Ahmadian, M. Izadi, P. Moradi, and M. Salehi, "Evaluating collaborative filtering recommender algorithms: A survey," *IEEE Access*, vol. 6, pp. 74003–74024, 2018.
[6] D. Wang, Y. Liang, D. Xu, X. Feng, and R. Guan, "A content-based recommender system for computer science publications," *Knowl.-Based Syst.*, vol. 157, pp. 1–9, Oct. 2018.
[7] M. H. Aghdam, "Context-aware recommender systems using hierarchical hidden Markov model," *Phys. A, Stat. Mech. Appl.*, vol. 518, pp. 89–98, Mar. 2019.
[8] N. Polatidis and C. K. Georgiadis, "A multi-level collaborative filtering method that improves recommendations," *Expert Syst. Appl.*, vol. 48, pp. 100–110, Apr. 2016.
[9] R. Xiong, J. Wang, N. Zhang, and Y. Ma, "Deep hybrid collaborative filtering for Web service recommendation," *Expert Syst. Appl.*, vol. 110, pp. 191–205, Nov. 2018.
[10] I. Portugal, P. Alencar, and D. Cowan, "The use of machine learning algorithms in recommender systems: A systematic review," *Expert Syst. Appl.*, vol. 97, pp. 205–227, May 2018.
[11] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surv.*, vol. 52, no. 1, pp. 1–38, Feb. 2019.
[12] C. Wu, J. Wu, C. Luo, Q. Wu, C. Liu, Y. Wu, and F. Yang, "Recommendation algorithm based on user score probability and project type," *EURASIP J. Wireless Commun. Netw.*, vol. 2019, no. 1, p. 80, Dec. 2019.
[13] U. Shardanand and P. Maes, "Social information filtering: Algorithms for automating word of mouth,"' in *Proc. SIGCHI Conf. Hum. Factors Comput. Syst.*, vol. 95. New York, NY, USA: ACM/Addison-Wesley, May 1995, pp. 210–217.
[14] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. Conf. World Wide Web (WWW)*. New York, NY, USA: Association Computing Machinery, 2001, pp. 285–295.
[15] A. Javari, J. Gharibshah, and M. Jalili, "Recommender systems based on collaborative filtering and resource allocation," *Social Netw. Anal. Mining*, vol. 4, no. 1, p. 234, Oct. 2014.
[16] X.-S. He, M.-Y. Zhou, Z. Zhuo, Z.-Q. Fu, and J.-G. Liu, "Predicting online ratings based on the opinion spreading process," *Phys. A, Stat. Mech. Appl.*, vol. 436, pp. 658–664, Oct. 2015.
[17] J. Ai, Z. Su, Y. Li, and C. Wu, "Link prediction based on a spatial distribution model with fuzzy link importance," *Phys. A, Stat. Mech. Appl.*, vol. 527, Aug. 2019, Art. no. 121155.
[18] D. Lemire and A. Maclachlan, "Slope one predictors for online rating-based collaborative filtering," in *Proc. SIAM Int. Conf. Data Mining*. Philadelphia, PA, USA: SIAM, Apr. 2005, pp. 471–475.
[19] S. Ahmadian, M. Afsharchi, and M. Meghdadi, "An effective social recommendation method based on user reputation model and rating profile enhancement," *J. Inf. Sci.*, vol. 45, no. 5, pp. 607–642, Oct. 2019.
[20] S. Ahmadian, N. Joorabloo, M. Jalili, Y. Ren, M. Meghdadi, and M. Afsharchi, "A social recommender system based on reliable implicit relationships," *Knowl.-Based Syst.*, vol. 192, Mar. 2020, Art. no. 105371.
[21] G. Gupta and R. Katarya, "A study of recommender systems using Markov decision process," in *Proc. 2nd Int. Conf. Intell. Comput. Control Syst. (ICICCS)*, Jun. 2018, pp. 1279–1283.
[22] R. Katarya, "Movie recommender system with metaheuristic artificial bee," *Neural Comput. Appl.*, vol. 30, no. 6, pp. 1983–1990, Sep. 2018.
[23] P. Moradi, F. Rezaimehr, S. Ahmadian, and M. Jalili, "A trust-aware recommender algorithm based on users overlapping community structure," in *Proc. 16th Int. Conf. Adv. ICT Emerg. Regions (ICTer)*, Sep. 2016, pp. 162–167.
[24] R. Katarya and Y. Arora, "CapsMF: A novel product recommender system using deep learning based text analysis model," *Multimedia Tools Appl.*, vol. 79, no. 47, pp. 35927–35948, Dec. 2020.
[25] S. Ahmadian, M. Meghdadi, and M. Afsharchi, "Incorporating reliable virtual ratings into social recommendation systems," *Appl. Intell.*, vol. 48, no. 11, pp. 4448–4469, Nov. 2018.
[26] R. Katarya and O. P. Verma, "Efficient music recommender system using context graph and particle swarm," *Multimedia Tools Appl.*, vol. 77, no. 2, pp. 2673–2687, Jan. 2018.
[27] H. A. Rahmani, M. Aliannejadi, S. Ahmadian, M. Baratchi, M. Afsharchi, and F. Crestani, "LGLMF: Local geographical based logistic matrix factorization model for POI recommendation," in *Information Retrieval Technology* (Lecture Notes in Computer Science), F. L. Wang, H. Xie, W. Lam, A. Sun, L.-W. Ku, T. Hao, W. Chen, T.-L. Wong, and X. Tao, Eds. Cham, Switzerland: Springer, 2020, pp. 66–78.
[28] D. K. Yadav and R. Katarya, "Study on recommender system using fuzzy logic," in *Proc. 2nd Int. Conf. Comput. Methodologies Commun. (ICCMC)*, Feb. 2018, pp. 50–54.
[29] Z. Su, Q. Zhang, J. Ai, and X. Sun, "Finite-time fuzzy stabilisation and control for nonlinear descriptor systems with non-zero initial state," *Int. J. Syst. Sci.*, vol. 46, no. 2, pp. 364–376, Jan. 2015.
[30] J. Ai, Y. Liu, Z. Su, H. Zhang, and F. Zhao, "Link prediction in recommender systems based on multi-factor network modeling and community detection," *EPL Europhys. Lett.*, vol. 126, no. 3, p. 38003, Jun. 2019.
[31] Z. Su, X. Zheng, J. Ai, L. Shang, and Y. Shen, "Link prediction in recommender systems with confidence measures," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 29, no. 8, Aug. 2019, Art. no. 083133.
[32] J. Ai, Z. Su, K. Wang, C. Wu, and D. Peng, "Decentralized collaborative filtering algorithms based on complex network modeling and degree centrality," *IEEE Access*, vol. 8, pp. 151242–151249, 2020.
[33] Z. Su, X. Zheng, J. Ai, Y. Shen, and X. Zhang, "Link prediction in recommender systems based on vector similarity," *Phys. A, Stat. Mech. Appl.*, vol. 560, Dec. 2020, Art. no. 125154.
[34] D. J. Watts and S. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.
[35] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct. 1999.
[36] M. E. J. Newman, "The structure and function of complex networks," *SIAM Rev.*, vol. 45, no. 2, pp. 167–256, 2003.
[37] Z. Su, J. Ai, Q. Zhang, and N. Xiong, "An improved robust finite-time dissipative control for uncertain fuzzy descriptor systems with disturbance," *Int. J. Syst. Sci.*, vol. 48, no. 8, pp. 1581–1596, Jun. 2017.
[38] K.-L. Wang, C.-X. Wu, J. Ai, and Z. Su, "Complex network centrality method based on multi-order K-shell vector," *Acta Phys. Sinica*, vol. 68, no. 19, 2019, Art. no. 196402.
[39] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, Apr. 2002.
[40] C. L. Apicella, F. W. Marlowe, J. H. Fowler, and N. A. Christakis, "Social networks and cooperation in hunter-gatherers," *Nature*, vol. 481, no. 7382, pp. 497–501, Jan. 2012.
[41] S. Ahmadian, N. Joorabloo, M. Jalili, M. Meghdadi, M. Afsharchi, and Y. Ren, "A temporal clustering approach for social recommender systems," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM)*. Barcelona, Spain: IEEE Press, Aug. 2018, pp. 1139–1144.
[42] C.-Y. Liu, C. Zhou, J. Wu, Y. Hu, and L. Guo, "Social recommendation with an essential preference space," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 346–353.
[43] S. Ahmadian, M. Meghdadi, and M. Afsharchi, "A social recommendation method based on an adaptive neighbor selection mechanism," *Inf. Process. Manage.*, vol. 54, no. 4, pp. 707–725, Jul. 2018.
[44] E. Bütün, M. Kaya, and R. Alhajj, "Extension of neighbor-based link prediction methods for directed, weighted and temporal social networks," *Inf. Sci.*, vols. 463–464, pp. 152–165, Oct. 2018.
[45] C.-H. Lai, S.-J. Lee, and H.-L. Huang, "A social recommendation method based on the integration of social relationship and product popularity," *Int. J. Hum.-Comput. Stud.*, vol. 121, pp. 42–57, Jan. 2019.
[46] L. Huang, W. Tan, and Y. Sun, "Collaborative recommendation algorithm based on probabilistic matrix factorization in probabilistic latent semantic analysis," *Multimedia Tools Appl.*, vol. 78, pp. 8711–8722, Jun. 2018.
[47] J. Ai, H. Zhao, K. M. Carley, Z. Su, and H. Li, "Neighbor vector centrality of complex networks based on neighbors degree distribution," *Eur. Phys. J. B*, vol. 86, no. 4, p. 163, Apr. 2013.
[48] K.-K. Shang, M. Small, and W.-S. Yan, "Link direction for link prediction," *Phys. A, Stat. Mech. Appl.*, vol. 469, pp. 767–776, Mar. 2017.
[49] V. Martínez, F. Berzal, and J. C. Cubero, "Probabilistic local link prediction in complex networks," in *Proc. Int. Conf. Scalable Uncertainty Manage.*, 2017, pp. 391–396.

[50] W. Pan, H. Ming, C. Chang, Z. Yang, and D.-K. Kim, "ElementRank: Ranking Java software classes and packages using a multilayer complex network-based approach," *IEEE Trans. Softw. Eng.*, early access, Oct. 8, 2019, doi: 10.1109/TSE.2019.2946357.
[51] H. Li, Y. Qu, S. Guo, G. Gao, R. Chen, and G. Chen, "Surprise bug report prediction utilizing optimized integration with imbalanced learning strategy," *Complexity*, vol. 2020, pp. 1–14, Feb. 2020.
[52] G. Xu, Z. Wu, Y. Zhang, and J. Cao, "Social networking meets recommender systems: Survey," *Int. J. Social Netw. Mining*, vol. 2, no. 1, pp. 64–100, 2015.
[53] M. M. Azadjalal, P. Moradi, A. Abdollahpouri, and M. Jalili, "A trust-aware recommendation method based on Pareto dominance and confidence concepts," *Knowl.-Based Syst.*, vol. 116, pp. 130–143, Jan. 2017.
[54] B. Yang, Y. Lei, J. Liu, and W. Li, "Social collaborative filtering by trust," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1633–1647, Aug. 2017.
[55] B. Hawashin, M. Lafi, T. Kanan, and A. Mansour, "An efficient hybrid similarity measure based on user interests for recommender systems," *Expert Syst.*, vol. 37, no. 5, Oct. 2020, Art. no. e12471.
[56] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "GroupLens: An open architecture for collaborative filtering of netnews," in *Proc. ACM Conf. Comput. Supported Cooperat. Work (CSCW)*. New York, NY, USA: Association Computing Machinery, 1994, pp. 175–186.
[57] J. Xu and H. Man, "Dictionary learning based on Laplacian score in sparse coding," in *Proc. 7th Int. Conf. Mach. Learn. Data Mining Pattern Recognit.* Berlin, Germany: Springer-Verlag, Aug. 2011, pp. 253–264.
[58] J. Bobadilla, A. Hernando, F. Ortega, and A. Gutiérrez, "Collaborative filtering based on significances," *Inf. Sci.*, vol. 185, no. 1, pp. 1–17, Feb. 2012.
[59] P. Jaccard, "Etude de la distribution florale dans une portion des Alpes et du Jura," *Bull. de la Soc. Vaudoise des Sci. Naturelles*, vol. 37, pp. 547–579, Jan. 1901.
[60] P. K. Singh, M. Sinha, S. Das, and P. Choudhury, "Enhancing recommendation accuracy of item-based collaborative filtering using Bhattacharyya coefficient and most similar item," *Appl. Intell.*, vol. 50, no. 12, pp. 4708–4731, Dec. 2020.
[61] J. Lu, J. Yue, L. Zhu, and G. Li, "Variational mode decomposition denoising combined with improved Bhattacharyya distance," *Measurement*, vol. 151, Feb. 2020, Art. no. 107283.
[62] C. Feng, J. Liang, P. Song, and Z. Wang, "A fusion collaborative filtering method for sparse data in recommender systems," *Inf. Sci.*, vol. 521, pp. 365–379, Jun. 2020.
[63] J. Huang, H. Oosterhuis, M. D. Rijke, and H. van Hoof, "Keeping dataset biases out of the simulation: A debiased simulator for reinforcement learning based recommender systems," in *Proc. 14th ACM Conf. Recommender Syst.* New York, NY, USA: Association Computing Machinery, Sep. 2020, pp. 190–199.
[64] B. D. Deebak and F. Al-Turjman, "A novel community-based trust aware recommender systems for big data cloud service networks," *Sustain. Cities Soc.*, vol. 61, Oct. 2020, Art. no. 102274.
[65] O. Papakyriakopoulos, J. C. M. Serrano, and S. Hegelich, "Political communication on social media: A tale of hyperactive users and bias in recommender systems," *Online Social Netw. Media*, vol. 15, Jan. 2020, Art. no. 100058.
[66] R. Kiran, P. Kumar, and B. Bhasker, "DNNRec: A novel deep learning based hybrid recommender system," *Expert Syst. Appl.*, vol. 144, Apr. 2020, Art. no. 113054.
[67] M. A. Hassan, M. G. M. Johar, and A. I. Hajamydeen, "A framework for recommender systems using improved collaborative filtering," in *Proc. IEEE Int. Conf. Autom. Control Intell. Syst. (I2CACIS)*, Jun. 2019, pp. 168–173.
[68] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech., Theory Exp.*, vol. 2008, no. 10, Oct. 2008, Art. no. P10008.
[69] J. Ai, L. Li, Z. Su, and C. Wu, "Online-rating prediction based on an improved opinion spreading approach," in *Proc. 29th Chin. Control Decis. Conf. (CCDC)*, May 2017, pp. 1457–1460.
[70] A. Moffat and J. Zobel, "Rank-biased precision for measurement of retrieval effectiveness," *ACM Trans. Inf. Syst.*, vol. 27, no. 1, pp. 1–27, Dec. 2008.
[71] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical analysis of predictive algorithms for collaborative filtering," in *Proc. 14th Conf. Uncertainty Artif. Intell.* San Francisco, CA, USA: Morgan Kaufmann, 1998, pp. 43–52.
[72] L. Lü, M. Medo, C. H. Yeung, Y.-C. Zhang, Z.-K. Zhang, and T. Zhou, "Recommender systems," *Phys. Rep.*, vol. 519, no. 1, pp. 1–49, 2012.
[73] F. H. D. Olmo and E. Gaudioso, "Evaluation of recommender systems: A new approach," *Expert Syst. Appl.*, vol. 35, no. 3, pp. 790–804, Oct. 2008.

ZHAN SU was born in Tieling, Liaoning, China, in 1983. She received the bachelor's degree in information and computing science and the Ph.D. degree in control theory and engineering from Northeastern University, Shenyang, China, in 2005 and 2012, respectively. Since 2013, she has been a Lecturer with the University of Shanghai for Science and Technology. Her research interests include intelligent control, complex networks theory, and recommender systems.

ZUYI LIN was born in Daqing, Heilongjiang, China, in 1996. She received the bachelor's degree in measurement and control technology and instrument specialty from the University of Shanghai for Science and Technology, in 2018, where she is currently pursuing the master's degree in detection technology and automatic equipment. Her research interests include complex networks and recommender systems.

JUN AI was born in Raohe, Heilongjiang, China, in 1980. He received the bachelor's degree in automation from the Shenyang University of Technology, in 2004, and the master's degree in computer application technology and the Ph.D. degree in computer science from Northeastern University, in 2008 and 2013, respectively. Since 2013, he has been a Lecturer with the University of Shanghai for Science and Technology. His research interests include complex networks and recommender systems.

HUI LI received the bachelor's degree in software engineering, the M.Sc. degree in computer science and technology, and the Ph.D. degree in computer architecture from Northeastern University, Shenyang, China, in 2006, 2008, and 2013, respectively. He is currently an Associate Professor with the School of Information Science and Technology, Dalian Maritime University, Dalian, China. His current research interests include complex networks and intelligent software systems.