Hindawi
Complexity
Volume 2020, Article ID 5206087, 19 pages
https://doi.org/10.1155/2020/5206087

Research Article

Recommendation Algorithm in Double-Layer Network Based on Vector Dynamic Evolution Clustering and Attention Mechanism

Jianrui Chen,[1,2] **Zhihui Wang**,[2] **Tingting Zhu**,[2] **and Fernando E. Rosas**[3]

[1]*Key Laboratory of Modern Teaching Technology, Ministry of Education, Xi'an, China*
[2]*School of Computer Science, Shaanxi Normal University, Xi'an, China*
[3]*Data Science Institute and Department of Brain Science, Imperial College London, London, UK*

Correspondence should be addressed to Jianrui Chen; jianrui_chen@snnu.edu.cn

Received 13 March 2020; Accepted 23 May 2020; Published 7 July 2020

Guest Editor: Liang Wang

The purpose of recommendation systems is to help users find effective information quickly and conveniently and also to present the items that users are interested in. While the literature of recommendation algorithms is vast, most collaborative filtering recommendation approaches attain low recommendation accuracies and are also unable to track temporal changes of preferences. Additionally, previous differential clustering evolution processes relied on a single-layer network and used a single scalar quantity to characterise the status values of users and items. To address these limitations, this paper proposes an effective collaborative filtering recommendation algorithm based on a double-layer network. This algorithm is capable of fully exploring dynamical changes of user preference over time and integrates the user and item layers via an attention mechanism to build a double-layer network model. Experiments on Movielens, CiaoDVD, and Filmtrust datasets verify the effectiveness of our proposed algorithm. Experimental results show that our proposed algorithm can attain a better performance than other state-of-the-art algorithms.

1. Introduction

Information overload is a pervasive problem in our era of big data, being a consequence of the rapid development of the Internet and other information technologies. Recommendation algorithms are one of the most widespread approaches to address this problem [1], whose purpose is to help users to find information quickly and conveniently. Additionally, recommendation systems usually suggest new items using information from previous searches, including product or media recommendation [2, 3]. Recommendation systems play a key role in the digital economy, as they allow web services to improve user's experience, increase product sales, and help products to realize their commercial value.

While the literature of recommendation systems is vast, most algorithms can be classified in three categories: content-based [4], collaborative filtering [5], and hybrid recommendation systems [6]. Among these methods, collaborative filtering recommendation algorithms are the most popular in both research and industry, as they can exploit social information better. Moreover, collaborative filtering algorithms can be further divided into three categories: memory-based [7], model-based [8], and hybrid filtering [9]. Among model-based recommendation algorithms, matrix factorization models stand out for their superior speed and strong scalability. In this literature, many matrix factorization models were proposed following the seminal work of Billsus and Pazzani [10]. Recent contributions include nonnegative matrix factorization methods for community detection [11] and dynamic networks [12]. Matrix factorization methods are particularly attractive as they can consider the influence of various factors, while bringing good performance and scalability. Unfortunately, conventional collaborative filtering approaches are not well-suited to deal with various problems related with the explosive development of information technologies, which often involve sparse data, cold start, or multidimensional data. These issues are likely to become widespread in the near future due to the continuous increase of online users and items, and hence new approaches that can address these

challenges are required. Since the community detection method can be applied to the recommendation system to find the interest communities of the target users, it can effectively achieve personalized recommendation.

On the above basis, the aim of this paper is to establish a double-layer network, apply attention mechanism to connect the double-layer network, and use the vector dynamic evolution clustering method to detect the community of users and items. The main challenges are as follows:

(i) Constructing the double-layer network: in this paper, the recommender system is modeled as a double-layer network, and they are the user layer and item layer. Similarity between nodes is the weighted edge between nodes in their corresponding layer.

(ii) Applying attention mechanism to connect the double-layer network: this paper puts forward a novel approach to carry out real-time recommendations, which is based on an attention mechanism and forgetting function that are used to fit scores and build relationships between users and items. The attention mechanism allows the algorithm to focus on particularly relevant factors while ignoring others, hence surpassing previous approaches based on limited scores.

(iii) Evolving the state vectors to detect the community of users and items: as the community detection method that can be used to find users' interest communities, it can be used to effectively achieve personalized recommendation. Hence, this paper proposes a community detection procedure for users and items, which uses an evolutionary clustering method based on vectorial dynamics. This clustering procedure enables a more accurate representation of the state value than other approaches based on scalar quantities. Our approach then leverages the community structure of users and items, finding neighbor sets of target users via a Cosine similarity method.

The efficiency of our approach is confirmed via various simulation experimental results, which show that the attained similarity between attribute and rating information is better than that of other state-of-the-art approaches.

The rest of the paper is organized as follows. The state of the art and related work is given in Section 2. Then, the proposed algorithm is introduced in Section 3. Section 4 presents experimental results, and finally Section 5 summarises our main conclusions. The convergence analysis of dynamical evolution clustering method in the double-layer network is shown in Appendix.

2. Related Work

Contemporary recommendation algorithms usually have to deal with challenges including sparse data, cold start, or multidimensional data. To deal with these challenges, Ling et al. proposed a recommendation algorithm for solving cold user problems by applying character capture and clustering methods [13]. Also, West et al. [14] illustrate how clustering technology can be combined with collaborative filtering to improve the recommendation performance. Importantly, as the clustering method divides users and items into several categories and then carries out collaborative filtering recommendation within the class, the recommendation time is greatly reduced. Building up on the well-known k-means clustering algorithm [15], Zahra et al. proposed the different kinds of recommendation algorithms for random selection of initial center of improved k-means clustering algorithm [16]. Because the community detection method can cluster the users with similar interest into the same community, push the users with different interest into different communities, and one can then find the nearest neighbor set in the similar interest community carrying out collaborative filtering. By performing recommendations within the community of the target user not only improves the recommendation accuracy but also reduces the complexity of the algorithm. So, community detection methods have been considered, including the community detection algorithm based on the similarity of paths proposed by Wu et al. [17], and the community detection algorithm is based on the simplification of complex network proposed by Bai et al. [18].

Other focus of study has been scenarios where networks are not static but evolve in time. In these cases, dynamic clustering algorithms are needed in order to obtain an adequate clustering effect. Wu et al. proposed a method for clustering based on dynamic synchronization [19, 20], and then we developed community detection approaches based on evolutionary clustering [21, 22]. Both methods brought similar users together while making dissimilar items distant, which improved the performance of the clustering algorithms. Bu et al. proposed a dynamic clustering strategy based on the attribute clustering graph [23]. One of the main findings of these works is that dynamical clustering methods tend to enable more efficient recommendation algorithms than traditional clustering methods.

Additionally, researchers have proposed different recommendation algorithms for various scenarios, including recommendation algorithm based on graph network models, attention networks, and multilayer networks. These are reviewed as follows:

(i) Graph network models: graph network has a flexible topology and can express complex relationships. This method treats users and items as nodes and represents relationships as edges between them. Edges are usually weighted and might be directed or undirected. To deal with sparse data and cold start, Moradi et al. applied trust information to the collaborative filtering method by graph clustering algorithm [24]. For cases with strong time constraints, such as financial news, Ren et al. proposed a graph embedding recommendation algorithm based on a heterogeneous graph model [25]. The performance

of recommendation can be greatly improved by exploring the graph network to recommend suitable items to users. Therefore, the relationship between the user and the item is represented by a graph network in this paper, so as to dig out more information in recommendation.

(ii) Attention mechanism: attention mechanisms are crucial in modulating the user experience and have been leveraged in various engineering applications including image processing and natural language processing. Extending previous recommendation algorithms were driven on user preferences but did not consider user attention. It is a good idea to incorporate the attention mechanism into the recommendation algorithm, which can make the recommendation algorithm more practical. Liang et al. proposed a mobile application for feature interaction through an attention mechanism [26]. At the same time, Feng et al. proposed a recommendation algorithm based on an attention network [27]. The abovementioned work has improved the performance to a certain extent by incorporating the attention mechanism into the recommendation system. Therefore, this paper uses the attention mechanism to connect the user layer and the item layer, so as to obtain a large double-layer graph network.

(iii) Multilayer networks: while most recommendation algorithms do not consider interactions that can take place between users and items, recent works have proposed to encode them in multilayer networks. Shang et al. proposed a video recommendation algorithm based on a hyperlink multilayer graph model [28]. Yasami proposed a new knowledge-based link recommendation approach using a nonparametric multilayer model of dynamic complex networks [29]. And the methods proposed by them all improve the algorithm based on single-layer network to some extent.

3. Proposed Algorithm

3.1. Scenario. Sometimes users do not know what clothes to buy, what movies to watch, what songs to listen, and so on. At this time, users can look at the items recommended by the recommendation system for users. Recommendation system is an information filtering scheme and it is based on the user's historical behavior data. The main task of the recommendation system is to predict the rating of the items by users and recommend the items to the users. However, given user history evaluation information (evaluation, rating, and timing), user attribute information (gender, age, occupation, zip code, etc.), and item attribute information (comedy, science, war, etc.), it is a problem to recommend appropriate and interested items to users.

Suppose there are m users and n items in a recommendation system. Most recommendation methods work in two ways. On the one hand, a recommendation might be made based on the attributes of the items that users like. For example, if a user likes comedies, then the system will recommend her to choose comedies within the available films. On the other hand, the system might look for the users who have similar preferences to a target user and recommend to the target user what similar users have chosen before.

3.2. Algorithm Detail Description. Here, we give the detail demonstration of our proposed method. Among them, Section 3.2.1 presents the construction of the double-layer network, and Section 3.2.2 presents the vector dynamic evolution clustering method and main convergence analysis. Furthermore, the predicted scores are obtained in each cluster, as shown in Section 3.2.3. Finally, the complete pseudocode and flow chart of our proposed algorithm are shown in Section 3.2.4.

The notations and their explanations in this paper are summarized in Table 1.

3.2.1. Network Model Construction. In this paper, all users and items are treated as nodes in networks, and the state of each node is represented as a vector. The user layer is set up by all users in the system, and the item layer is constructed by all items. The similarity between users is regarded as the edge weight of the user layer. The similarity between items is regarded as the edge weight of the item layer. The user and item layers are connected through attention mechanisms and ratings. In this way, the double-layer network model is formed and a simple example is shown in Figure 1.

(1) Constructing the User Layer Network. Everyone observes the same thing in different angles. In the same way, interests of everyone will be various. According to Movielens dataset, from the perspective of gender, male prefers action movies and female prefers romantic movies. Based on this, to a large extent, people who have similar attribute information have similar interests and preferences. We integrate the score information into the calculation of similarity. Firstly, the ages are divided into three stages: younger than 18, 18 to 55, and older than 55. The occupations are divided into three classes: culture class, leisure class, and management class. The sexes are categorized into male and female. If the user has this attribute, it is 1; and if the user does not have this attribute, it is 0. Thus, the attribute vector of the ith user is 8-dimensional 0-1 vector, denoted as U_i.

Then, attribute similarity between users $S_U^{\text{attribute}}$ is defined as follows:

$$S_U^{\text{attribute}}(i, j) = \frac{1}{1 + \left\| U_i - U_j \right\|}, \tag{1}$$

where U_i denotes the attribute vector of the ith user.

Define rating similarity between users S_U^{rating} as follows:

$$S_U^{\text{rating}}(i, j) = \frac{1}{1 + \left\| Ur_i - Ur_j \right\|}, \tag{2}$$

Table 1: Notations and explanations.

Notations	Explanations
m	The number of users
n	The number of items
U_i	Attribute vector of user i
I_i	Attribute vector of item i
R_{ui}	Original score of user u to item i
t_{ui}	Rating time of user u to item i
$S_U^{\text{attribute}}(i,j)$	Attribute similarity between users i and j
$S_U^{\text{rating}}(i,j)$	Rating similarity between users i and j
S_U	Similarity matrix between users
$S_I^{\text{attribute}}(i,j)$	Attribute similarity between items i and j
$S_I^{\text{rating}}(i,j)$	Rating similarity between items i and j
S_I	Similarity matrix between items
A_U	Attention matrix of user to item
A_I	Attention matrix of item to user
$\mathbf{x}_i(t)$	State vector of user node i at time t
$\mathbf{y}_i(t)$	State vector of item node i at time t

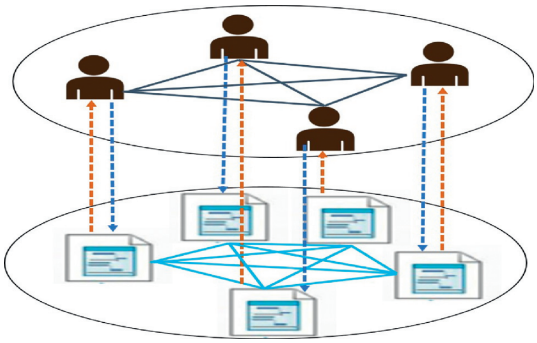

Figure 1: Double-layer network model.

where Ur_i and Ur_j denote the score vectors of users i and j with sharing scores on items.

In order to get more consistent with the actual user similarity, the final user similarity obtained by associating user attribute similarity and common rating similarity is defined as follows:

$$S_U = \alpha \cdot S_U^{\text{attribute}} + (1-\alpha) \cdot S_U^{\text{rating}}. \tag{3}$$

Here, $\alpha \in [0,1]$ is a mixture parameter, which is the convex combination of two similarity with different angles. So, the adjacent matrix of user layer is obtained, which can represent the coupling relation between users.

(2) Constructing the Item Layer Network. Similar to the construction of the user layer, this paper integrates the score information and item attribute into the calculation of similarity between items. In MovieLens dataset, there are 19 attributes of movies. If movie has this attribute, it is 1; if movie does not have this attribute, it is 0. Thus, the attribute vector of the ith item can be represented as a 19 dimensional vector, denoted as I_i.

Define attribute similarity between items $S_I^{\text{attribute}}$ is defined as follows:

$$S_I^{\text{attribute}}(i,j) = \frac{1}{1+\left\|I_i - I_j\right\|}, \tag{4}$$

where I_i denotes the attribute vector of the ith item.

Define rating similarity between items S_U^{rating} as follows:

$$S_I^{\text{rating}}(i,j) = \frac{1}{1+\left\|Ir_i - Ir_j\right\|}, \tag{5}$$

where Ir_i and Ir_j denote the score vectors of items i and j which are evaluated jointly by the users.

In order to get more accurate relation between items, the final item similarity obtained by associating item attribute similarity and common rating similarity is defined as follows:

$$S_I = \beta \cdot S_I^{\text{attribute}} + (1-\beta) \cdot S_I^{\text{rating}}. \tag{6}$$

Here, $\beta \in [0,1]$ is a mixture parameter, which is the convex combination of two similarities with different angles. So, the adjacent matrix of the item layer is obtained, which can represent the coupling relation between items.

(3) Connections between the User Layer and the Item Layer. In addition, to establish the relationship in the user layer and the item layer, the connection between the two network layers plays a critical role. German psychologist *Hermann Ebbinghaus* found that the human brain forgetting rule is as follows: the process of forgetting is very fast, and forget quickly at first, and then slowly [30]. Inspired by *Ebbinghaus*, we believe that interests of people will also change with time. The closer the score to the current time is, the better it can express current interests of users. According to fitting the *Ebbinghaus* forgetting curve, we obtain a forgetting curve more in line with interests and hobbies of the people. The forgetting function proposed in this paper is as follows:

$$f(t_{ui}) = a \cdot \exp\left(b \cdot \frac{t_{ui} - t_u^{\min}}{t_u^{\max} - t_u^{\min}}\right) + c \cdot \exp\left(d \cdot \frac{t_{ui} - t_u^{\min}}{t_u^{\max} - t_u^{\min}}\right). \tag{7}$$

Moreover, the fitted coefficients are $a = 0.6368$, $b = -1.947$, $c = 0.3623$, and $d = 0.0025$. Additionally, $\exp(\cdot)$ is the exponential function. $f(t_{ui})$ represents the forgetting degree of uth user to the ith item. t_{ui} represents the time of the uth user rating for the ith item. $t_u^{\min}$ represents the earliest rating time of user u. $t_u^{\max}$ represents the latest rating time of user u.

Then, we use the attention mechanism to connect the user layer to the item layer on the processed score. In daily life, everyone pays attention to different things differently. For example, young girls pay more attention to romantic movies than to war movies, and no one can pay attention to everything. Based on this, we define the attention of the user to the item as follows:

$$A_U(u,i) = \frac{\exp\left(R_{ui} \times f(t_{ui})\right)}{\sum_{n \in N_u} \exp\left(R_{ni} \times f(t_{ni})\right)}. \tag{8}$$

Similarly, every item is not designed for everyone. Some items target different types of users. So, define the attention of the item to the user as follows:

$$A_I(i,u) = \frac{\exp\left(R_{iu}^T \times f\left(t_{iu}\right)\right)}{\sum_{n\in N_i}\exp\left(R_{in}^T \times f\left(t_{in}\right)\right)}. \tag{9}$$

In equations (8) and (9), R_{ui} represents the initial score of the ith item rated by the uth user. R_{iu}^T is the score that item i was rated by user u. N_u represents the neighbor user set of user u. We view the items evaluated by user u as the item neighbor set of user u. N_i represents the neighbor set of item i. We treat all users who have evaluated item i as neighbors of item i. $A_U(u,i)$ represents the attention of the uth user to the ith item. $A_I(i,u)$ represents the attention of the ith item to the uth user. Obviously, A_U and A_I are not mutually symmetric matrices.

To illustrate the previous definitions, we give a simple example with three users U_1–U_3 and four items I_1–I_4. R represents the original scoring matrix:

$$R = \begin{array}{c} \\ U_1 \\ U_2 \\ U_3 \end{array} \begin{array}{c} \begin{array}{cccc} I_1 & I_2 & I_3 & I_4 \end{array} \\ \begin{pmatrix} 5 & 3 & 0 & 3 \\ 3 & 3 & 5 & 1 \\ 5 & 0 & 3 & 0 \end{pmatrix} \end{array}. \tag{10}$$

A_U represents the user attention matrix, obtained by equation (8):

$$A_U = \begin{array}{c} \\ U_1 \\ U_2 \\ U_3 \end{array} \begin{array}{c} \begin{array}{cccc} I_1 & I_2 & I_3 & I_4 \end{array} \\ \begin{pmatrix} 0.0140 & 0.0012 & 0 & 0.0003 \\ 0.0008 & 0.0008 & 0.0057 & 0.0001 \\ 0.0048 & 0 & 0.0008 & 0 \end{pmatrix} \end{array}. \tag{11}$$

In the original score matrix R, we can find that $R_{12} = 3$ and $R_{14} = 3$. They have the same scores, but they are $A_U(1,2) = 0.0012$ and $A_U(1,4) = 0.0003$ in the attention matrix of users. That means the attention mechanism is meaningful in this paper.

A_I represents the item attention matrix, obtained by equation (9):

$$A_I = \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{ccc} U_1 & U_2 & U_3 \end{array} \\ \begin{pmatrix} 0.0109 & 0.0012 & 0.0016 \\ 0.0088 & 0.0104 & 0 \\ 0 & 0.0251 & 0.0009 \\ 0.0031 & 0.0020 & 0 \end{pmatrix} \end{array}. \tag{12}$$

Besides, in the original score matrix R, we can find that $R_{11} = 5$ and $R_{31} = 5$, but by processing, $A_I(1,1) = 0.0109$ and $A_I(1,3) = 0.0016$ in the attention matrix of items. The attention values of the users to the items are the directed edges from the user layer to the item layer, and the attention values of the items to the users are the directed edges from the item layer to the user layer. Experiments show that attention mechanism can greatly improve the recommendation performance.

3.2.2. Dynamic Evolution Clustering in Double-Layer Network. In recent years, due to the explosion of data, clustering methods emerge endlessly. The clustering method can not only greatly reduce the recommendation time but also improve the recommendation performance. Cluster analysis finds different communities, gathers similar things into one cluster, and pushes dissimilar things in different clusters. Among them, dynamic clustering is more in line with the real situation, so it is applied in various scenarios [19, 21, 22, 31–35]. The phase of the previous dynamic evolution clustering method is only a scalar, which cannot express the interest of users better. In order to grasp the changing rules of interest in different periods, we propose a vector dynamic evolution clustering method.

In this paper, we propose a vector dynamic evolution clustering method in the user layer as follows:

$$\begin{aligned} \mathbf{x}_i(t+1) = \mathbf{x}_i(t) &+ K_1 \sum_{j\in DU_i} S_{U_{ij}} \cdot \sin\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right) \\ &+ K_2 \sum_{j\notin DU_i} S_{U_{ij}} \cdot \sin\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right) \\ &+ K_3 \sum_{j\in AU_i} A_{U_{ij}} \cdot \sin\left(M\cdot\mathbf{y}_j(t) - \mathbf{x}_i(t)\right) \\ &+ K_4 \sum_{j\notin AU_i} A_{U_{ij}} \cdot \sin\left(M\cdot\mathbf{y}_j(t) - \mathbf{x}_i(t)\right), \\ &i, = 1,2,\ldots,m. \end{aligned} \tag{13}$$

Here, $\mathbf{x}_i(t)$ and $\mathbf{x}_i(t+1)$ represent the state vectors of user node i at time t and time $t+1$. $DU_i \triangleq \left\{j \mid S_{U_{ij}} \geq \mathrm{Ave}\left(S_{U_i}\right)\right\}$, where $\mathrm{Ave}\left(S_{U_i}\right)$ represents the average edge weight of user node i, which is the average similarity between user node i and other users. $AU_i \triangleq \left\{j \mid A_{U_{ij}} \geq h_1\right\}$, where h_1 represents the average value of nonzero elements in attention matrix A_U. K_1, K_2, K_3, and K_4 are the clustering coefficients. K_1 and K_3 are the positive coupling coefficients, and K_2 and K_4 are the negative coupling coefficients. The matrix M is defined as follows:

$$M_{ij} = \frac{\left|V_j \cap W_i\right|}{\left|V_j\right|}. \tag{14}$$

Here, M_{ij} represents the influence degree of the ith user attribute on the jth item attribute. V_j represents the set of users who have evaluated items in the jth category. W_i represents the set of users with the ith attribute. The purpose of adding matrix M to the evolution equation is to emphasize the different influences of different user attributes.

In the same way, we propose the vector dynamic clustering method in the item layer as follows:

$$\begin{aligned} \mathbf{y}_i(t+1) = \mathbf{y}_i(t) &+ K_5 \sum_{j\in DI_i} S_{I_{ij}} \cdot \sin\left(\mathbf{y}_j(t) - \mathbf{y}_i(t)\right) \\ &+ K_6 \sum_{j\notin DI_i} S_{I_{ij}} \cdot \sin\left(\mathbf{y}_j(t) - \mathbf{y}_i(t)\right) \\ &+ K_7 \sum_{j\in AI_i} A_{I_{ij}} \cdot \sin\left(M^T\cdot\mathbf{x}_j(t) - \mathbf{y}_i(t)\right) \\ &+ K_8 \sum_{j\notin AI_i} A_{I_{ij}} \cdot \sin\left(M^T\cdot\mathbf{x}_j(t) - \mathbf{y}_i(t)\right), \\ &i = 1,2,\ldots,n. \end{aligned} \tag{15}$$

Among them, $\mathbf{y}_i(t)$ and $\mathbf{y}_i(t+1)$ represent the state vectors of item node i at time t and time $t+1$. $DI_i \triangleq \{j \mid S_{I_{ij}} \geq \mathrm{Ave}(S_{I_i})\}$, where $\mathrm{Ave}(S_{I_i})$ represents the average edge weight of user node i, that is, the average similarity between item node i and other items. $AI_i \triangleq \{\{j\} A_{I_{ij}} \geq h_2\}$, where h_2 represents the average value of nonzero elements in attention matrix $\mathbf{A}_I$. K_5, K_6, K_7, and K_8 are the clustering coefficients. K_5 and K_7 are the positive coupling coefficients, and K_6 and K_8 are the negative coupling coefficients. M^T is the transpose of the matrix M. The purpose of adding matrix M^T to the evolution equation is to emphasize the different influences of different item attributes.

Besides, this community detection evolution process can be stable after some iterations. The convergence results are obtained from the following theorems according to *Lyapunov* theory.

Theorem 1. *Vector dynamic evolution process equations (13) and (15) can be converted into the following forms:*

$$\begin{aligned} \mathbf{x}(t+1) &\leq \mathbf{x}(t) + S\mathbf{x}(t) + T\mathbf{y}(t), \\ \mathbf{y}(t+1) &\leq \mathbf{y}(t) + G\mathbf{y}(t) + H\mathbf{x}(t). \end{aligned} \tag{16}$$

Proof. See it in the Appendix, in the end of this paper. □

Theorem 2. *If appropriate parameters K_1, K_2, K_3, K_4, K_5, K_6, K_7, and K_8 make the $\theta < 0$ and $\omega < 0$, then the fixed points in equations (13) and (15) are uniformly stable.*

Proof. See it in the Appendix, in the end of this paper.

The convergence analysis shows that our community detection algorithm will be stable after some iterations. Finally, the user nodes with similar state vectors and item nodes with similar state vectors are assigned to the same community, that is, all users and all items with higher similarity are assigned to their community with similar interest. □

3.2.3. Score Prediction and Recommendation. In order to sort the user similarity of the same community and obtain the nearest neighbor set of the target user for collaborative filtering recommendation, Cosine similarity is used in this paper to calculate the community similarity:

$$\mathrm{Sim}_{uv}^{\mathrm{COS}} = \frac{\sum_{i \in S_{uv}} R_{ui} R_{vi}}{\sqrt{\sum_{i \in S_{uv}} R_{ui}^2} \sqrt{\sum_{i \in S_{uv}} R_{vi}^2}}. \tag{17}$$

To compare different similarity indexes, Pearson correlation coefficient and adjusted Cosine similarity [36, 37] are adopted:

$$\begin{aligned} \mathrm{Sim}_{uv}^{\mathrm{PCC}} &= \frac{\sum_{i \in S_{uv}} (R_{ui} - \overline{R}_u)(R_{vi} - \overline{R}_v)}{\sqrt{\sum_{i \in S_{uv}} (R_{ui} - \overline{R}_u)^2} \sqrt{\sum_{i \in S_{uv}} (R_{vi} - \overline{R}_v)^2}}, \\ \mathrm{Sim}_{uv}^{\mathrm{ACOS}} &= \frac{\sum_{i \in S_{uv}} (R_{ui} - \overline{R}_u)(R_{vi} - \overline{R}_v)}{\sqrt{\sum_{i \in S_u} (R_{ui} - \overline{R}_u)^2} \sqrt{\sum_{i \in S_v} (R_{vi} - \overline{R}_v)^2}}. \end{aligned} \tag{18}$$

Here, R_{ui} represents the score of the ith item is rated by the uth user, S_{uv} represents the set of items both scored by users u and v, $\overline{R}_u$ represents the average score of user u, and S_u represents the set of items scored by user u. Through the above methods, the similarity ranking of target users within the community can be obtained, and the score can be predicted.

Previous prediction methods did not take into account the item community and averaged all item scores, which could not better express the score of target users in the community of target items. In order to better reflect the role of the community, this paper proposes the following methods to predict the score:

$$P_{ui} = \overline{R}_{uC_i} + \frac{\sum_{v \in N_u} \mathrm{Sim}_{uv} \times (R_{vi} - \overline{R}_u)}{\sum_{v \in N_u} \mathrm{Sim}_{uv}}. \tag{19}$$

Here, C_i is the community of item i, $\overline{R}_{uC_i}$ is the average score of user u to the ith item community, N_u represents the nearest neighbor set of user u, P_{ui} represents the prediction score of target user u for item i, Sim_{uv} represents the similarity value between users u and v, and $\overline{R}_u$ represents the average score of user u.

3.2.4. Algorithm Flowchart. This paper presents a recommendation algorithm in **D**ouble-layer **N**etwork based on **V**ector dynamic evolution **C**lustering and **A**ttention mechanism (denoted as *DN-VCA*). The pseudocode is shown in Algorithm 1.

Firstly, we use node attribute information to construct an undirected network in the layer. Secondly, connect double layers of networks with attention mechanism so that a double-layer network connection is established as directed relationship. Thirdly, community detection is carried out according to vector dynamic evolution clustering. Fourthly, according to the new prediction method proposed in this paper, score prediction is carried out. Finally, we give a list of recommendations.

The flow chart of the algorithm proposed in this paper is shown in Figure 2.

4. Experiments and Results

4.1. Datasets. To verify the effectiveness of the model and our proposed algorithm, Movielens-100k, CiaoDVD, and Filmtrust datasets are tested in this paper.

(i) **Movielens-100k**[1]: MovieLens is a set of movie ratings. The dataset contains 100,000 ratings provided by 943 users for 1682 movies, and scores are 1–5. Each score has its corresponding time. The dataset also has user attribute information and movie category information. Each user has at least 20 score records. The data sparsity is 93.7%.

(ii) **CiaoDVD**[2]: the dataset contains 278,483 ratings provided by 7375 users for 99746 DVDs, and scores are 1–5. As the sparsity of this dataset was 99.97%, we retain the users with more than 20 evaluation DVD and the DVD with more than 20 evaluation values so that the sparsity after

Input: Training set u_1, Test set u_2. Parameters K_1, K_2, …, K_8, ε_1, ε_2.
Output: **Prediction** score P_{ui}, *MAE*, *RMSE*, Recommendation list.
//Constructing **the** double-layer network
Compute the User **similarity** matrix S_U, Item similarity matrix S_I, User-item attention matrix A_U, Item-user attention matrix A_I according to equations (3), (6), (8) and (9).
//Community detection in double-layer network
for each node **do**
Apply vector dynamic evolution clustering equations (13) and (15) to find the appropriate community for each node.
If $\|\mathbf{x}_i(t+1) - \mathbf{x}_i(t)\| < \varepsilon_1$ and $\|\mathbf{y}_j(t+1) - \mathbf{y}_j(t)\| < \varepsilon_2$ **then** all nodes stop iterating
end if
end for
//Calculate the dynamic similarity
for each user community and item community **do**
Apply equation (17) to calculate the similarity matrix in the same community.
end for
//Prediction score
for each target user and target item **do**
Find the neighbor set of the target user by similarity sort.
Compute the prediction score according to equation (19)
end for
//Select the Top-N items as recommendation list for target user

Algorithm 1: *DN-VCA*.

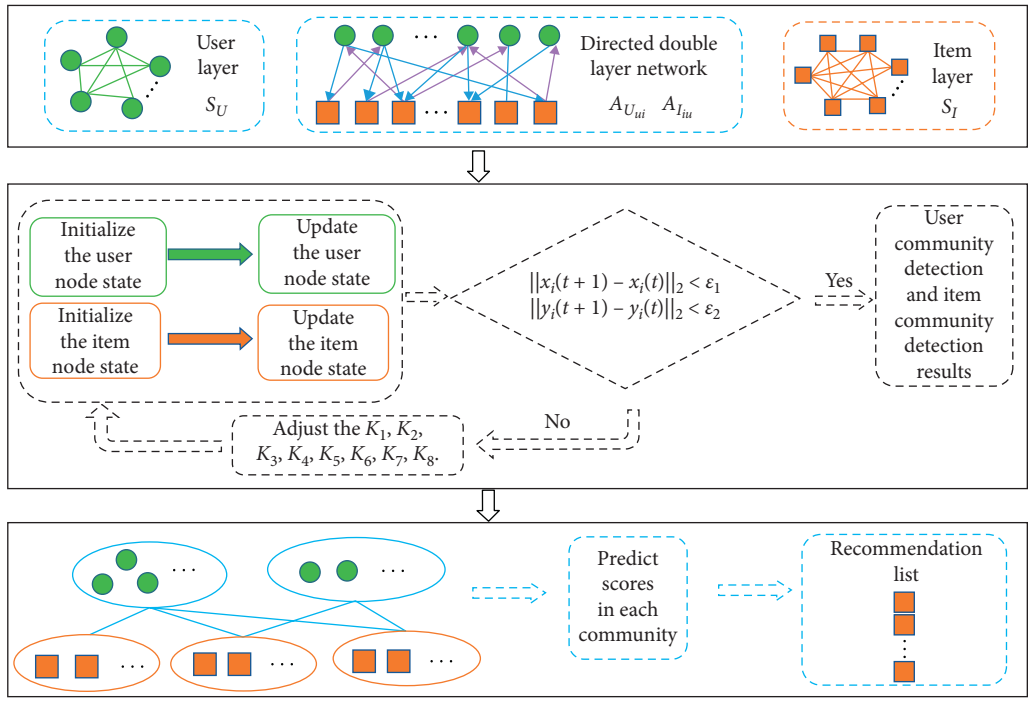

Figure 2: Flowchart of DN – VCA.

processing is 97.01% and the processed dataset is denoted as **CiaoDVD-1.**

(iii) **Filmtrust**[3]: the dataset contains 35,497 ratings provided by 1508 users for 2071 movies, and scores are 0.5–4 in intervals of 0.5. The data sparsity is 98.86%. Moreover, most items have not been evaluated by users, so we only keep the items that have been evaluated more than 3 times. At the same time, we delete the users whose evaluation times are less than 3 times. Finally, the data sparsity was slightly reduced to 96.61% and the processed dataset is denoted as **Filmtrust-1.**

The original dataset is randomly divided into 80% training set and 20% test set to verify the proposed algorithm on three datasets. In addition, all the algorithms in this paper have conducted five cross experiments, and finally, the

average of five cross experiments are taken as the results. Table 2 gives the statistics of three datasets.

4.2. Evaluation Indexes. In order to verify the accuracy of the algorithm proposed in this paper, we use the following five evaluation indicators:

$$\text{MAE} = \frac{\sum_{i=1}^{M} |R_{ui} - P_{ui}|}{M},$$
$$\text{RMSE} = \sqrt{\frac{M \sum_{i=1}^{M} (R_{ui} - P_{ui})^2}{M}}. \quad (20)$$

Here, MAE is the mean absolute error, RMSE is the root mean squared error, R_{ui} represents the true score of item i by user u in the test set, P_{ui} represents the prediction score generated by the algorithm, and M represents the number of scores be predicted in the test set. Fixing integer processing is conducted on Movielens and CiaoDVD-1 datasets, but the scores on Filmtrust-1 are floating point numbers with 0.5 as the interval, so no integer processing is performed on this dataset.

Since we present recommendations to target users after the prediction, the following three indexes are used to verify the efficiency of recommendations:

$$\text{Recall} = \frac{\sum_u |R_u \cap T_u|}{\sum_u |T_u|},$$
$$\text{Precision} = \frac{\sum_u |R_u \cap T_u|}{\sum_u |R_u|}, \quad (21)$$
$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

Here, R_u represents the set of items recommended by the algorithm for the target user u and T_u represents the set of items that target user u really likes in the test set. Precision indicates the proportion of the items the user really likes in the recommendation list to the total number of recommendations. Recall represents the ratio of the favorite items of users in the recommendation list to the favorite items of the user. F_1 value is a comprehensive indicator of Precision and Recall.

4.3. Parameter Analysis. There are several parameters in our proposed algorithm and we will discuss their influences in the recommendation performance.

The influence of convex combination parameters α in equation 3 and β in equation 6: here, on the Movielens dataset, the selected sets of α and β both are [0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]. We tested the effect of α and β values on MAE and RMSE.

As shown in Figure 3(a), when $\alpha = 0$ and $\beta = 0$, that is, user similarity and item similarity only contain score information, MAE = 0.7085. When $\alpha = 1$ and $\beta = 1$, that is, user similarity and item similarity only contain attribute information, MAE = 0.7637. When $\alpha = 1$ and $\beta = 0$, that is, user similarity only contains attribute information and item similarity only contains score information, MAE = 0.7087. When $\alpha = 0$ and $\beta = 1$, that is, user similarity only contains score information and item similarity only contains attribute information, MAE = 0.7634. From the results, we can see that MAE = 0.7045 is the lowest, as $\alpha = 0.3$ and $\beta = 0.4$, in other words, the experimental results show the best when the similarity includes both score information and attribute information. Similarly, as shown in Figure 3(b), RMSE = 0.9947 is lowest when $\alpha = 0.3$ and $\beta = 0.4$. In summary, $\alpha = 0.3$ and $\beta = 0.4$ on Movielens dataset in the following experiment. Because there is no attribute information in CiaoDVD-1 and Filmtrust-1 datasets, so $\alpha = 0$ and $\beta = 0$.

The influence of thresholds ε_1 and ε_2 in termination criteria: in order to obtain stable state vectors of nodes more accurately and faster, we limit the number of iterations of dynamic evolution clustering equations. If the state difference between the front and back vectors is less than the threshold, iteration is terminated. We selected several pairs of user layer threshold ε_1 and item layer threshold ε_2 for experiments.

As shown in Figure 4, on the Movielens dataset, both *MAE* and *RMSE* are the lowest when $\varepsilon_1 = 0.2$ and $\varepsilon_2 = 0.1$. In other words, when $\varepsilon_1 = 0.2$ and $\varepsilon_2 = 0.1$, the algorithm achieves a better prediction accuracy. Therefore, on the Movielens dataset, we choose $\varepsilon_1 = 0.2$ and $\varepsilon_2 = 0.1$ for experiments.

As shown in Figure 5, on the CiaoDVD-1 dataset, both *MAE* and *RMSE* are the lowest when $\varepsilon_1 = 2.1$ and $\varepsilon_2 = 2.1$, except for the prediction when the number of neighbors is 90. So, on the CiaoDVD-1 dataset, we choose $\varepsilon_1 = 2.1$ and $\varepsilon_2 = 2.1$ for following experiments.

As it can be seen from Figure 6, MAE = 0.6164 when $\varepsilon_1 = 0.01$ and $\varepsilon_2 = 0.01$, and the number of neighbors is 60. When $\varepsilon_1 = 0.001$ and $\varepsilon_2 = 0.001$, MAE = 0.6155. When the neighbor number is taken into other values, both MAE and RMSE of $\varepsilon_1 = 0.01$ and $\varepsilon_2 = 0.01$ are the lowest, that is, the prediction effect is the best. As a result, the two threshold values selected in the experiment on Filmtrust-1 dataset in this paper are $\varepsilon_1 = 0.01$ and $\varepsilon_2 = 0.01$.

In this paper, only four pairs of parameters are selected for comparison on each dataset, and finally the threshold values with the best effect are selected. In fact, thresholds other than the above can lead to better predictions.

4.4. Comparing Similarity. In order to get better results, we choose one of Cosine similarity, Pearson correlation coefficient, and adjusted Cosine similarity, so we conducted experiments on the Movielens dataset.

As shown in Figure 7, because MAE and RMSE of Cosine similarity are lower than Pearson correlation coefficient and adjusted Cosine similarity regardless of the number of neighbors, we select Cosine as the measure of dynamic similarity within the community.

4.5. Comparing Results of Different Recommendation Algorithms. In this paper, our proposed algorithm (*DNVCA*) is compared with three existing algorithms from Collaborative Filtering recommendation algorithm based on *K-means* clustering (named as *K-CF*), Ref. [38] (named as *DTNM*), and Ref. [32] (named as *EHC-CF*). The selected neighbor set is [10, 20, 30, 40, 50, 60, 70, 80, 90, 100].

Table 2: Statistics of datasets.

	Movielens-100k	CiaoDVD	CiaoDVD-1	Filmtrust	Filmtrust-1
User	943	7375	471	1508	1227
Item	1682	99746	689	2071	793
Rating	100000	278483	9707	35497	33009
Scale	[1–5]	[1–5]	[1–5]	[0.5–4]	[0.5–4]
Sparse degree (%)	93.71	99.97	97.01	98.86	96.61

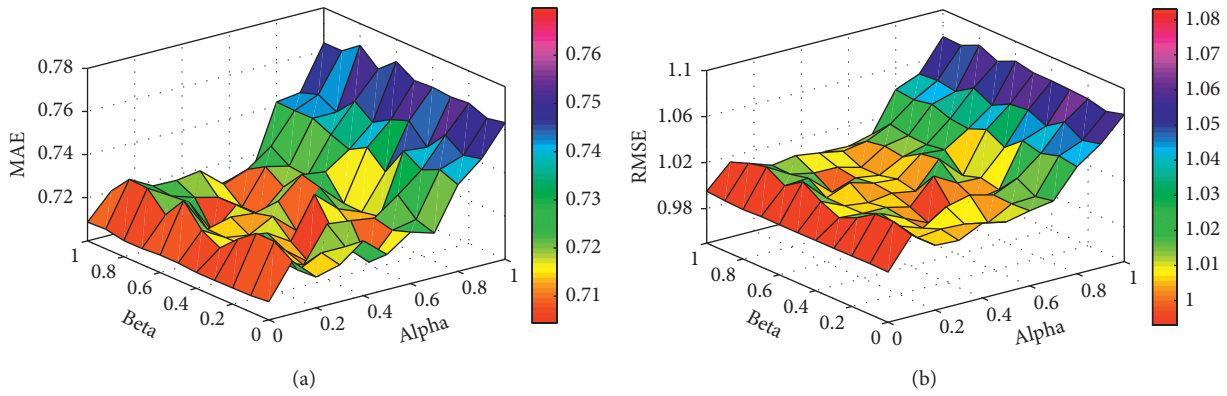

Figure 3: Movielens. (a) MAE results of different values of α and β. (b) RMSE results of different values of α and β.

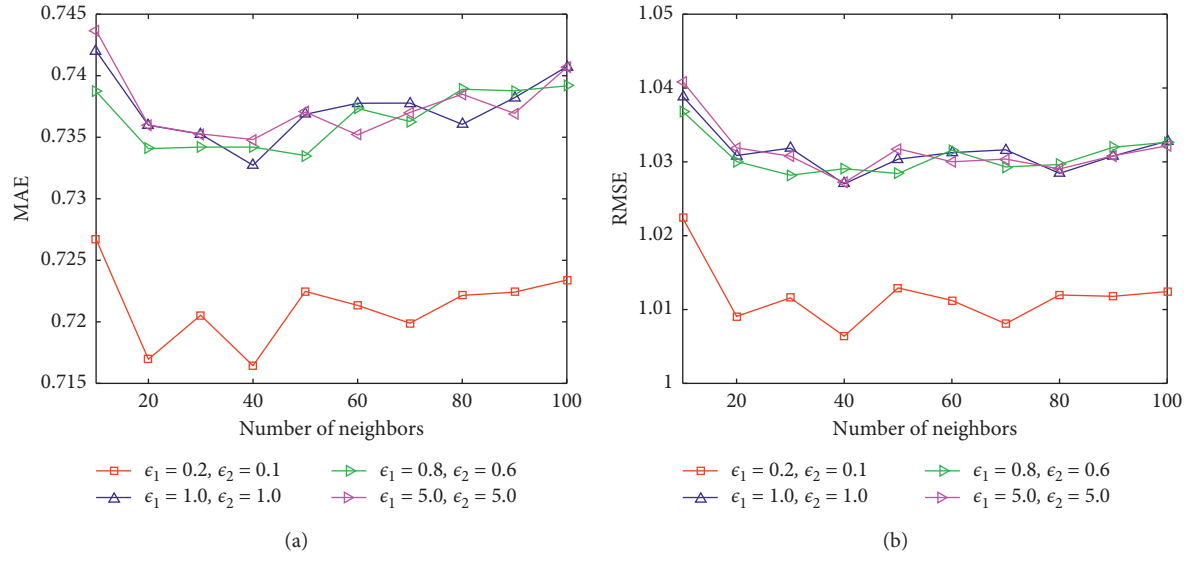

Figure 4: Movielens. (a) MAE results between different values of ε_1 and ε_2. (b) RMSE results between different values of ε_1 and ε_2.

4.5.1. Movielens-100k. In order to compare the accuracy of our proposed algorithm with other algorithms, a number of experiments are carried out on the *Movielens* dataset.

Figure 8 shows the clustering results based on our double-layer network evolutionary clustering method. Six communities are formed for *Movielens-100k*.

As shown in Figure 9, regardless of the number of neighbors, the *MAE* values and *RMSE* values of our *DN-VCA* are lower than the other three compared algorithms.

The reason our *DN-VCA* has a good performance is that double-layer network can better represent the relations between users and items. Moreover, our prediction method is no longer based on the user community, but adds the item community into the prediction method, highlighting the role of the item community.

Next, we present the Top-N recommendation list for the user, and select the set of recommended number as [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]. The comparison results of *Precision*, *Recall*, and F_1 are shown in Table 3.

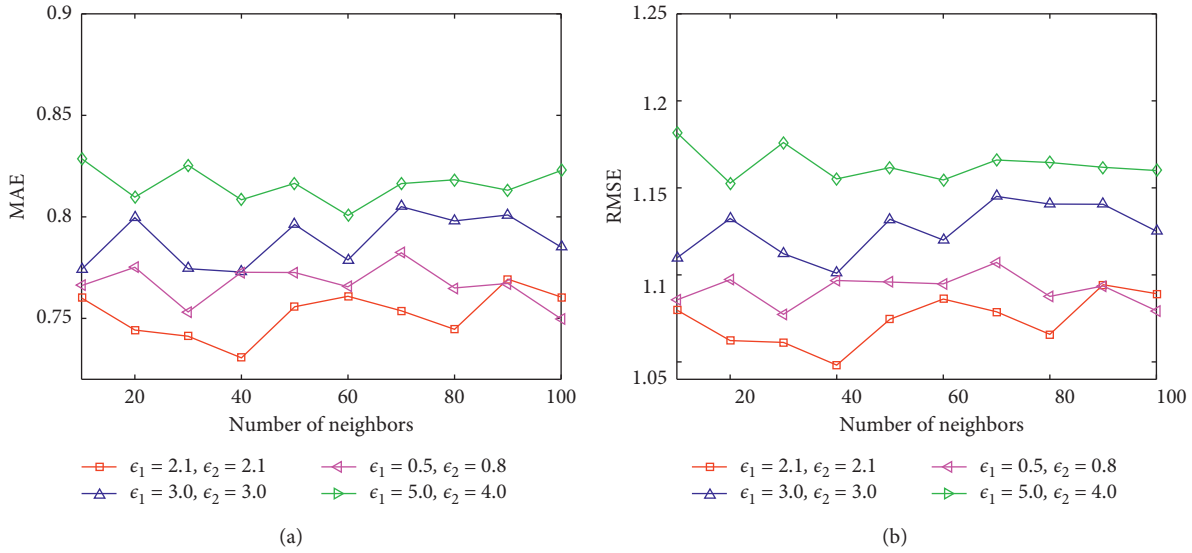

Figure 5: CiaoDVD-1. (a) MAE results between different values of ϵ_1 and ϵ_2. (b) RMSE results between different values of ϵ_1 and ϵ_2.

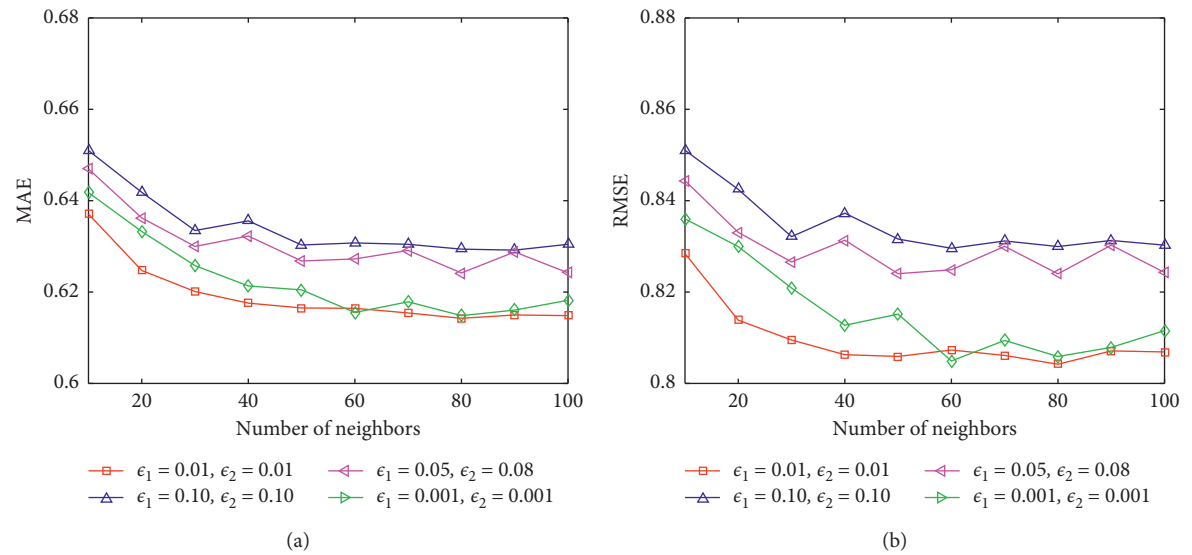

Figure 6: Filmtrust-1. (a) MAE results between different values of ϵ_1 and ϵ_2. (b) RMSE results between different values of ϵ_1 and ϵ_2.

As shown in Table 3, as the number of recommendations changes, *Recall* and F_1 values will increase, while the *Precision* will decrease. Compared with algorithm *DTNM*, *EHC-CF*, and *K-CF*, our *DN-VCA* has a slightly higher advantage in *Precision*, *Recall*, and F_1 values.

4.5.2. CiaoDVD-1. To further test the performance of the algorithm, we test many experiments on the CiaoDVD-1 dataset.As shown in Figure 10, because the sparsity of this dataset is very large, the result of the algorithm has a strong oscillation. However, no matter how many neighbors we have, the MAE and RMSE values of our *DN-VCA* are lower than the other three compared algorithms, except that when the number of neighbors is 20, and the *MAE* and *RMSE* values are slightly higher than *EHC-CF*.

As shown in Table 4, compared with algorithm *DTNM* and *K-CF*, our *DN-VCA* has a significant advantage, and our *DN-VCA* is very close with *EHC-CF*. In general, the *DN-VCA* algorithm we proposed is higher than the other three algorithms in *MAE*, *RMSE*, *Precision*, *Recall*, and F_1 values.

4.5.3. FilmTrust-1. Different from the previous three datasets, the scale of this dataset is 0.5, so in the final prediction results, the experiments in this paper do not conduct rounding processing on the predicted scores.

It can be seen from Figure 11 that the *DN-VCA* proposed in this paper is slightly lower than *EHC-CF* when the number of neighbors is 10, and the *MAE* of EHC-*CF* is 0.6365, while the *MAE* of *DN-VCA* is 0.6371. When the number of neighbors is taken as other values, the accuracy of

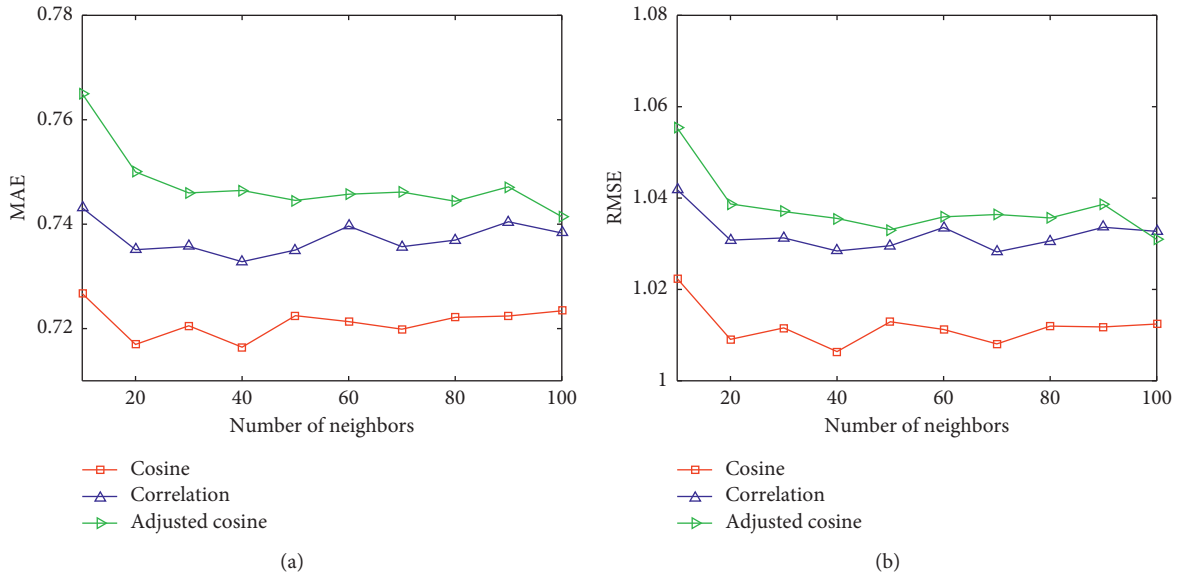

Figure 7: Movielens. (a) MAE results between three different similarities. (b) RMSE results between three different similarities.

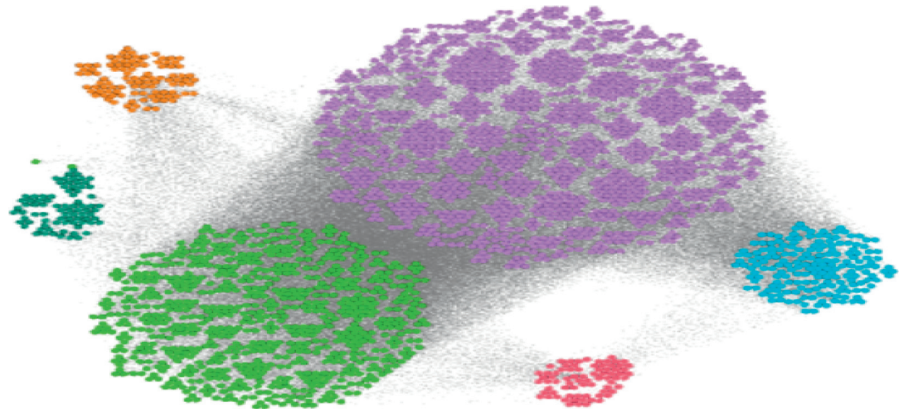

Figure 8: Clustering results based on *Movielens-100k*.

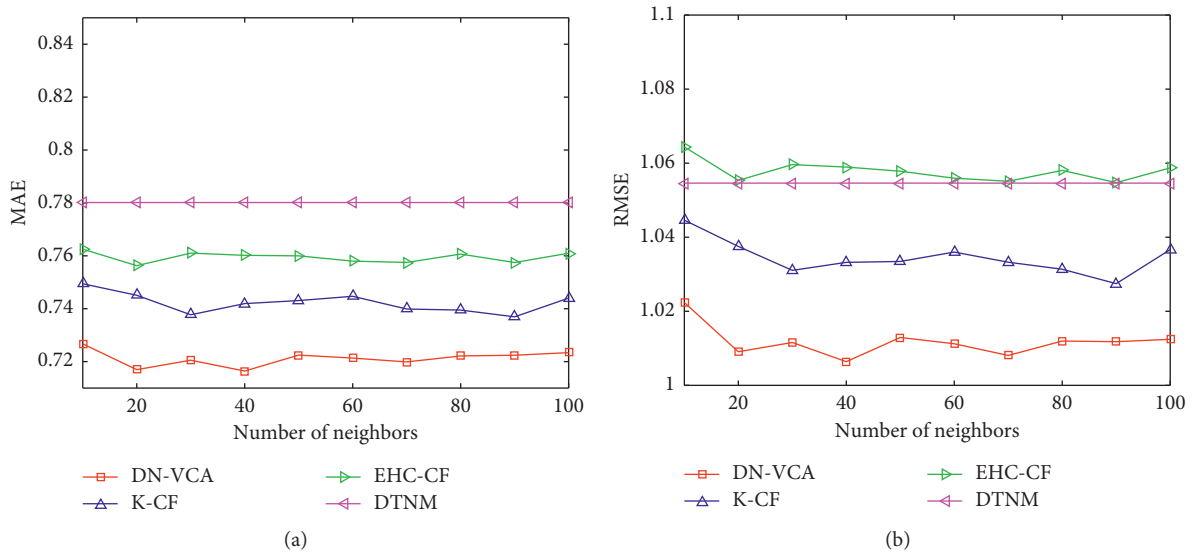

Figure 9: Movielens. (a) Comparison results of MAE of four algorithms. (b) Comparison results of RMSE of four algorithms.

Table 3: Precision, recall, and F_1 of EHC-CF, DTNM, K-CF, and DN-VCA based on Movielens dataset.

Algorithm	Metrics (%)	Recommended number									
		2	4	6	8	10	12	14	16	18	20
EHC-CF	Precision	90.3805	89.7659	89.3578	88.7287	88.3573	88.2147	87.8782	87.7129	87.6330	87.4568
	Recall	7.8480	15.1348	21.7199	27.5310	32.6933	37.3786	41.5252	45.3709	48.9157	52.1453
	F_1	14.3758	25.7226	34.6698	41.6885	47.3606	52.1235	56.0057	59.4090	62.3873	64.9418
DTNM	Precision	91.2053	89.7964	89.1047	88.8487	88.4917	88.2884	88.2264	87.9556	87.7681	87.6785
	Recall	7.9072	15.1289	21.6744	27.5701	32.7656	37.4209	41.6886	45.4873	48.9955	52.2710
	F_1	14.4857	25.7146	34.5931	41.7475	47.4577	52.1785	56.2263	59.5654	62.4880	65.1010
K-CF	Precision	91.2274	90.1739	89.5158	89.1396	88.8328	88.4051	88.2116	88.0297	87.9305	87.7079
	Recall	7.9072	15.2000	21.7640	27.6597	32.8672	37.4738	41.6812	45.5312	49.0895	52.2939
	F_1	14.4860	25.8335	34.7383	41.8829	47.6137	52.2509	56.2164	59.6208	62.6062	65.1270
DN-VCA	Precision	**91.7247**	**90.4707**	**89.7541**	**89.4433**	**88.9443**	**88.5499**	**88.3380**	**88.0890**	**87.9783**	**87.8778**
	Recall	**7.9570**	**15.2437**	**21.8078**	**27.7400**	**32.9021**	**37.5253**	**41.7313**	**45.5466**	**49.1077**	**52.3803**
	F_1	**14.5765**	**25.9100**	**34.8122**	**42.0086**	**47.6662**	**52.3267**	**56.2874**	**59.6466**	**62.6328**	**65.2411**

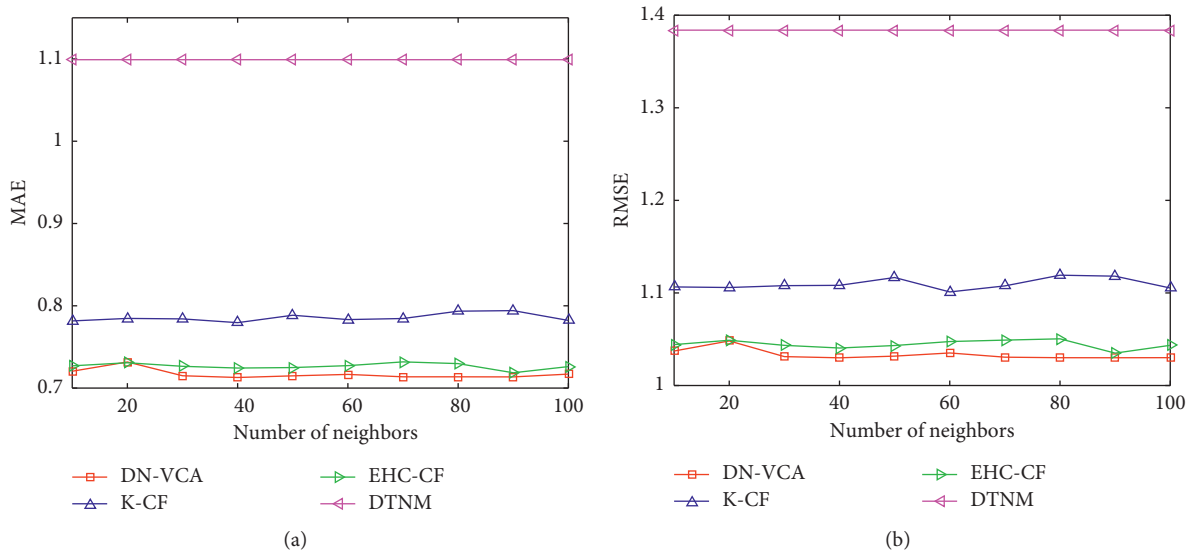

Figure 10: CiaoDVD-1. (a) Comparison results of MAE of four algorithms. (b) Comparison results of RMSE of four algorithms.

Table 4: Precision, recall, and F_1 of EHC-CF, DTNM, K-CF, and DN-VCA based on CiaoDVD-1 dataset.

Algorithm	Metrics (%)	Recommended number									
		2	4	6	8	10	12	14	16	18	20
EHC-CF	Precision	24.5900	13.6557	9.6985	7.6019	6.3958	5.5571	4.9320	4.4214	4.0643	3.7721
	Recall	11.3877	11.8937	12.3989	12.8134	13.3843	13.8904	14.3312	14.6425	15.1090	15.5502
	F_1	15.5662	12.7134	10.8831	9.5420	8.6551	7.9379	7.3381	6.7916	6.4052	6.0711
DTNM	Precision	13.9274	8.4802	5.9297	5.1109	4.4484	3.8422	3.6072	3.4007	3.1722	3.0071
	Recall	6.4437	7.3774	7.5714	8.6098	9.2974	9.5947	10.4794	11.2553	11.7857	12.3821
	F_1	8.8107	7.8901	6.6504	6.4139	6.0174	5.4869	5.3667	5.2231	4.9987	4.8388
K-CF	Precision	20.6154	12.1088	8.2584	6.7321	5.5848	4.9203	4.3153	4.0033	3.7157	3.3792
	Recall	9.5448	10.5441	10.5573	11.3497	11.6845	12.2939	12.5426	13.2556	13.8116	13.9290
	F_1	13.0480	11.2720	9.2669	8.4509	7.5571	7.0276	6.4210	6.1492	5.8557	5.4387
DN-VCA	Precision	**24.8136**	**13.7930**	**9.8938**	**7.6697**	**6.5324**	**5.6663**	**4.9717**	**4.4229**	**4.1280**	**3.8314**
	Recall	**11.4897**	**12.0078**	**12.6432**	**12.9301**	**13.6684**	**14.1613**	**14.4467**	**14.6415**	**15.3412**	**15.7958**
	F_1	**15.7062**	**12.8381**	**11.1002**	**9.6277**	**8.8396**	**8.0936**	**7.3972**	**6.7932**	**6.5052**	**6.1667**

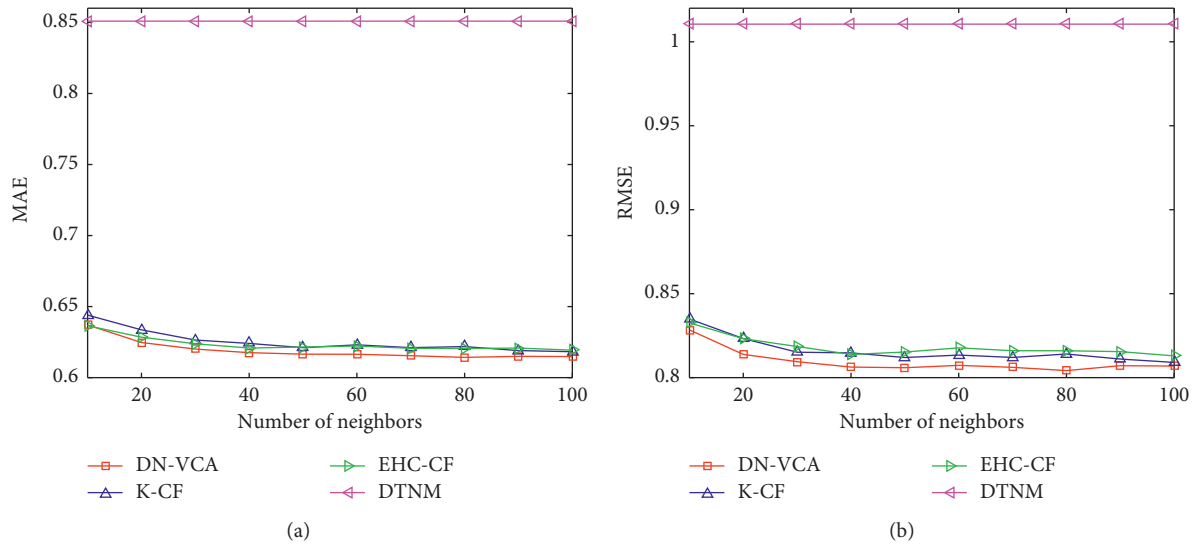

Figure 11: FilmTrust-1. (a) Comparison results of MAE of four algorithms. (b) Comparison results of RMSE of four algorithms.

Table 5: Precision, recall, and F_1 of EHC – CF, DTNM, K – CF, and DN – VCA based on FilmTrust-1 dataset.

Algorithm	Metrics (%)	Recommended number									
		2	4	6	8	10	12	14	16	18	20
EHC–CF	Precision	82.2555	83.8450	82.8444	82.2364	81.5371	81.2952	81.1014	81.0292	80.9482	80.9327
	Recall	34.3482	57.5925	73.4343	84.5687	91.3565	95.2834	97.2184	98.1894	98.6618	99.0253
	F_1	48.9674	68.2819	77.8552	83.3853	86.1669	87.7341	88.4304	88.7868	88.9306	89.0687
DTNM	Precision	85.2386	84.0522	83.0014	82.1382	81.6513	81.2537	81.0887	80.9642	80.9636	80.9419
	Recall	34.3415	57.7353	73.5729	84.4672	91.4840	95.2343	97.2034	98.1105	98.6805	99.0364
	F_1	48.9578	68.4510	78.0024	83.2855	86.2874	87.6891	88.4167	88.7155	88.9475	89.0788
K-CF	Precision	85.1992	84.1723	83.0559	82.2075	81.7250	81.3210	81.1107	81.0446	**81.0157**	80.9571
	Recall	34.3256	57.8176	73.6217	84.5388	91.5668	95.3132	97.2297	98.2083	**98.7445**	99.0555
	F_1	48.9351	68.5486	78.0539	83.3559	86.3654	87.7617	88.4407	88.8037	**89.0050**	89.0957
DN-VCA	Precision	**85.8327**	**84.3690**	**83.3180**	**82.4043**	**81.7817**	**81.3914**	**81.1733**	**81.0943**	81.0097	**80.9695**
	Recall	**34.5804**	**57.9529**	**73.8548**	**84.7414**	**91.6301**	**95.3956**	**97.3045**	**98.2682**	98.7368	**99.0703**
	F_1	**49.2986**	**68.7090**	**78.3006**	**83.5556**	**86.4252**	**87.8376**	**88.5088**	**88.8580**	88.9982	**89.1092**

our *DN-VCA* has significant advantages with *DTNM*. Compared with *EHC-CF* and *K-CF*, the *MAE* and *RMSE* of *DN-VCA* are also the lowest. The results show that the proposed *DN-VCA* achieves good performance on FilmTrust-1 dataset when predicting scores.

As shown in Table 5, each of the four algorithms gives similar results in *Precision*, *Recall*, and F_1 values. However, *DN-VCA* that we proposed has the best results, which are optimal in *Precision*, *Recall*, and F_1 values except that when the number of recommendations is 18, the F_1 value of *DN-VCA* is lower than 0.0068 of *K-CF*, the *Precision* of *DN-VCA* is lower than 0.006 of *K-CF*, and the *Recall* of *DN-VCA* is lower than 0.0077 of *K-CF*. *DN-VCA* obtained the best recommendation result when the number of recommendations is 20.

Experimental results on three datasets show that *MAE* and *RMSE* of our proposed *DN-VCA* are lower than the other four comparison algorithms in predicting scores, so our proposed algorithm is effective in predicting scores. In addition, in terms of recommendation, *DN-VCA* also achieves better results than other algorithms in *Precision*, *Recall*, and F_1 values. The *K-CF* and *EHC-CF* of the three comparison methods are recommendation algorithms based on clustering algorithms. Our results show that our vector dynamic evolution clustering algorithm outperforms these other clustering algorithms, hence suggesting that our proposed method can also be effective for generating recommendations. So, it proves that the algorithm we proposed is effective in the recommend system. That means our double-layer network construction is meaningful and the dynamic clustering in the double-layer network can gather the similar interest users together to the same community. Neighbor users with high similarity give valuable suggestions in the collaborative filter recommendation process.

5. Conclusion

In this paper, a novel vector dynamic evolutionary clustering recommendation algorithm *DN-VCA* based on double-layer network and attention mechanism is proposed. Our algorithm firstly constructs a double-layer network model through node similarity and an attention mechanism. Then, the improved vector dynamic evolution clustering equation

is used in the double-layer network to cluster the nodes into the most suitable community. Finally, the similarity between nodes is calculated within the community for enabling collaborative filtering recommendation. We not only verify the validity of the *DN-VCA*, but also prove the theoretical results of the algorithm. Additionally, we solve the shortcomings of the existing methods. For example, previous algorithms are only based on the single-layer network, and the state of nodes is only a scalar when clustering is dynamically evolving. With the development of big data, the recommendation system faces more and more users and items. It is impossible for us to carry out similar comparison with all users for collaborative filtering based on users, and the vector dynamic evolution clustering proposed in this paper is a good community detection method to solve this problem. Please note that our algorithm has a number of degrees of freedom that are highly nontrivial to optimize. Actually, our results do not use optimal parameters values; if those parameters could be further optimized then the performance of our algorithm would further improve. Finding efficient optimization methods for these parameters constitute an interesting field of future research.

Appendix

A. Proof of Theorem 1

In order to evolve the double-layer network model, the following matrices are firstly defined:

$$\begin{aligned}
P^{+} &= \left\{ A_{U_{ij}} \,\middle|\, A_{U_{ij}} \geq h_1, \quad i = 1, \ldots, m, j = 1, \ldots, n \right\},\\
P^{-} &= \left\{ A_{U_{ij}} \,\middle|\, A_{U_{ij}} < h_1, \quad i = 1, \ldots, m, j = 1, \ldots, n \right\},\\
Q^{+} &= \left\{ A_{I_{ij}} \,\middle|\, A_{I_{ij}} \geq h_2, \quad i = 1, \ldots, n, j = 1, \ldots, m \right\},\\
Q^{-} &= \left\{ A_{I_{ij}} \,\middle|\, A_{I_{ij}} < h_2, \quad i = 1, \ldots, n, j = 1, \ldots, m \right\},\\
U^{+} &= \left\{ S_{U_{ij}} \,\middle|\, S_{U_{ij}} \geq \mathrm{Ave}\left(S_{U_i}\right), \quad i = 1, \ldots, m, j = 1, \ldots, m \right\},\\
U^{-} &= \left\{ S_{U_{ij}} \,\middle|\, S_{U_{ij}} < \mathrm{Ave}\left(S_{U_i}\right), \quad i = 1, \ldots, m, j = 1, \ldots, m \right\},\\
I^{+} &= \left\{ S_{I_{ij}} \,\middle|\, S_{I_{ij}} \geq \mathrm{Ave}\left(S_{I_i}\right), \quad i = 1, \ldots, n, j = 1, \ldots, n \right\},\\
I^{-} &= \left\{ S_{I_{ij}} \,\middle|\, S_{I_{ij}} < \mathrm{Ave}\left(S_{I_i}\right), \quad i = 1, \ldots, n, j = 1, \ldots, n \right\}.
\end{aligned} \tag{A.1}$$

Matrix P^{+} keeps the elements of $\mathrm{Att}_{U_{ij}} \geq h_1$, and the rest are all 0. Matrix Q^{+} keeps the elements of $Att_{I_{ij}} \geq h_2$, and the rest elements in Q^{+} are all 0. And similarly, matrix P^{-} keeps the elements of $\mathrm{Att}_{U_{ij}} < h_1$, and the rests are all 0. Matrix Q^{-} keeps the elements of $\mathrm{Att}_{I_{ij}} < h_2$, and the rest elements in Q^{-} are all 0. U^{+} and U^{-} contain elements that satisfy $S_{U_{ij}} \geq \mathrm{Ave}\,(S_{U_i})$ and $S_{U_{ij}} < \mathrm{Ave}\,(S_{U_i})$, and the other elements are all 0. Matrix I^{+} contains all elements that are greater than or equal to the mean similarity of each item. Matrix I^{-} contains all elements less than the mean similarity of each item, and all other elements are equal to 0.

Theorem A.1. *Vector dynamic evolution process equations (13) and (15) can be converted into the following forms:*

$$\begin{aligned}
\mathbf{x}(t+1) &\leq \mathbf{x}(t) + S\mathbf{x}(t) + T\mathbf{y}(t),\\
\mathbf{y}(t+1) &\leq \mathbf{y}(t) + G\mathbf{y}(t) + H\mathbf{x}(t).
\end{aligned} \tag{A.2}$$

Proof

(i) The user layer vector dynamic evolution process equation (13) is

$$\begin{aligned}
\mathbf{x}_i(t+1) = \mathbf{x}_i(t) &+ K_1 \sum_{j \in DU_i} S_{U_{ij}} \cdot \sin\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right)\\
&+ K_2 \sum_{j \notin DU_i} S_{U_{ij}} \cdot \sin\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right)\\
&+ K_3 \sum_{j \in AU_i} A_{U_{ij}} \cdot \sin\left(M \cdot \mathbf{y}_j(t) - \mathbf{x}_i(t)\right)\\
&+ K_4 \sum_{j \notin AU_i} A_{U_{ij}} \cdot \sin\left(M \cdot \mathbf{y}_j(t) - \mathbf{x}_i(t)\right).
\end{aligned} \tag{A.3}$$

The initial values are all selected from the range of $[0, \pi/2]$. $S_{U_{ij}} \in [0,1]$ and $A_{U_{ij}} \in [0,1]$. We can obtain $-(\pi/2) \leq \mathbf{x}_j(t) - \mathbf{x}_i(t) \leq (A_{U_{ij}} \in [0,1])$ and $-(\pi/2) \leq \mathbf{y}_j(t) - \mathbf{y}_i(t) \leq (\pi/2)$. And then can obtain $\sin(\mathbf{x}_j(t) - \mathbf{x}_i(t)) \leq \mu_{ij}(\mathbf{x}_j(t) - \mathbf{x}_i(t))$ and $\sin(\mathbf{y}_j(t) - \mathbf{y}_i(t)) \leq \lambda_{ij}(\mathbf{y}_j(t) - \mathbf{y}_i(t))$, where $\mu_{ij} > 0$ and $\lambda_{ij} > 0$ and they meet the following conditions:

$$\mu_{ij} = \begin{cases} \dfrac{1}{2}, & -\dfrac{\pi}{2} \leq \mathbf{x}_j(t) - \mathbf{x}_i(t) < 0,\\[2ex] 1, & 0 \leq \mathbf{x}_j(t) - \mathbf{x}_i(t) \leq \dfrac{\pi}{2}, \end{cases} \tag{A.4}$$

$$\mu_{ij} = \begin{cases} \dfrac{1}{2}, & -\dfrac{\pi}{2} \leq \mathbf{y}_j(t) - \mathbf{x}_i(t) < 0,\\[2ex] 1, & 0 \leq \mathbf{y}_j(t) - \mathbf{x}_i(t) \leq \dfrac{\pi}{2}. \end{cases}$$

Then, equation (13) can be converted to

$$\begin{aligned}
\mathbf{x}_i(t+1) = \mathbf{x}_i(t) &+ K_1 \sum_{j=1}^{m} U_{ij}^{+} \cdot \mu_{ij}\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right)\\
&+ K_2 \sum_{j=1}^{m} U_{ij}^{-} \cdot m_{ij}\left(\mathbf{x}_j(t) - \mathbf{x}_i(t)\right)\\
&+ K_3 \sum_{j=1}^{n} P_{ij}^{+} \cdot \lambda_{ij}\left(M\mathbf{y}_j(t) - \mathbf{x}_i(t)\right)\\
&+ K_4 \sum_{j=1}^{n} P_{ij}^{-} \cdot \lambda_{ij}\left(M\mathbf{y}_j(t) - \mathbf{x}_i(t)\right).
\end{aligned} \tag{A.5}$$

Then,

$$\begin{aligned}\mathbf{x}_i(t+1) &= \mathbf{x}_i(t) + K_1\sum_{j=1}^{m}\left(U_{ij}^{+}\mu_{ij}\mathbf{x}_j(t)\right) - K_1\left(\sum_{j=1}^{m}U_{ij}^{+}\mu_{ij}\right)\mathbf{x}_i(t) + K_2\sum_{j=1}^{m}\left(U_{ij}^{-}\mu_{ij}\mathbf{x}_j(t)\right) - K_2\left(\sum_{j=1}^{m}U_{ij}^{-}\mu_{ij}\right)\mathbf{x}_i(t)\\ &+ K_3\sum_{j=1}^{n}\left(P_{ij}^{+}\lambda_{ij}M\mathbf{y}_j(t)\right) - K_3\left(\sum_{j=1}^{n}P_{ij}^{+}\lambda_{ij}\right)\mathbf{x}_i(t) + K_4\sum_{j=1}^{n}\left(P_{ij}^{+}\lambda_{ij}M\mathbf{y}_j(t)\right) - K_4\left(\sum_{j=1}^{n}P_{ij}^{-}\lambda_{ij}\mathbf{x}_i(t)\right)\\ &\le \mathbf{x}_i(t) + K_1\sum_{j=1}^{m}U_{ij}^{+}\mathbf{x}_j(t) - \frac{K_1}{2}\left(\sum_{j=1}^{m}U_{ij}^{+}\right)\mathbf{x}_i(t) + K_2\sum_{j=1}^{m}U_{ij}^{-}\mathbf{x}_j(t) - \frac{K_2}{2}\left(\sum_{j=1}^{m}U_{ij}^{-}\mathbf{x}_i(t)\right)\\ &+ K_3\sum_{j=1}^{n}P_{ij}^{+}M\mathbf{y}_j(t) - \frac{K_3}{2}\left(\sum_{j=1}^{n}P_{ij}^{+}\right)\mathbf{x}_i(t) + K_4\sum_{j=1}^{n}P_{ij}^{-}M\mathbf{y}_j(t) - \frac{K_4}{2}\left(\sum_{j=1}^{n}P_{ij}^{-}\right)\mathbf{x}_i(t).\end{aligned} \tag{A.6}$$

Suppose the user state vector $\mathbf{x}_i(t)$ is an f-dimensional column vector, and the item state vector $\mathbf{y}_i(t)$ is a g-dimensional column vector. Denote $\mathbf{x}(t) = (\mathbf{x}_1(t), \mathbf{x}_2(t), \ldots, \mathbf{x}_m(t))^T$ and $\mathbf{y}(t) = (\mathbf{y}_1(t), \mathbf{y}_2(t), \ldots, \mathbf{y}_n(t))^T$. So, $\mathbf{x}(t)$ is a column vector in mf dimension and $\mathbf{y}(t)$ is a column vector in ng dimension.

Define a special matrix A^+ as follows:

$$\begin{bmatrix} U_{11}^{+} & 0 & \cdots & 0 & \cdots & U_{1m}^{+} & 0 & \cdots & 0\\ 0 & U_{11}^{+} & \cdots & 0 & \cdots & 0 & U_{1m}^{+} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & U_{11}^{+} & \cdots & 0 & 0 & \cdots & U_{1m}^{+}\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ U_{m1}^{+} & 0 & \cdots & 0 & \cdots & U_{mm}^{+} & 0 & \cdots & 0\\ 0 & U_{m1}^{+} & \cdots & 0 & \cdots & 0 & U_{mm}^{+} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & U_{m1}^{+} & \cdots & 0 & 0 & \cdots & U_{mm}^{+} \end{bmatrix}. \tag{A.7}$$

Matrix A^+ is a large matrix of $mf \times mf$ dimension, composed of m^2 small diagonal matrices with f dimension. Similarly, the elements of A^- are made up of the elements of U^-.

Define the diagonal elements of matrix B^+ as follows:

$$\begin{aligned} B_{ii}^{+} &= \sum_{j=1}^{m}U_{1j}^{+}, \quad i = 1, \ldots, f,\\ B_{ii}^{+} &= \sum_{j=1}^{m}U_{2j}^{+}, \quad i = f+1, \ldots, 2f,\\ &\cdots\\ B_{ii}^{+} &= \sum_{j=1}^{m}U_{mj}^{+}, \quad i = (m-1)f+1, \ldots, mf. \end{aligned} \tag{A.8}$$

B^+ matrix is the diagonal matrix of $mf \times mf$, the first f diagonal elements are the same, the $f+1$ to $2f$ elements are the same, and so on, and the last f elements are the same. Similarly, we can define B^-, and the elements of B^- are made up of the elements of the matrix P^-.

Then, define the matrix C^+ as follows:

$$\begin{bmatrix} P_{11}^{+} & 0 & \cdots & 0 & \cdots & P_{1n}^{+} & 0 & \cdots & 0\\ 0 & P_{11}^{+} & \cdots & 0 & \cdots & 0 & P_{1n}^{+} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & P_{11}^{+} & \cdots & 0 & 0 & \cdots & P_{1n}^{+}\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ P_{m1}^{+} & 0 & \cdots & 0 & \cdots & P_{mn}^{+} & 0 & \cdots & 0\\ 0 & P_{m1}^{+} & \cdots & 0 & \cdots & 0 & P_{mn}^{+} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & P_{m1}^{+} & \cdots & 0 & 0 & \cdots & P_{mn}^{+} \end{bmatrix}. \tag{A.9}$$

Matrix C^+ is a large matrix of $mf \times nf$ dimension. When the elements of C^+ become are substituted by the elements of P^-, the above matrix becomes C^-.

Define the matrix E as follows:

$$E = \begin{bmatrix} M & & & \\ & M & & \\ & & \ddots & \\ & & & M \end{bmatrix}. \tag{A.10}$$

The matrix E is made up of multiple M matrices and it is diagonal block matrix. Matrix M is the $f \times g$ dimension matrix defined by equation (14). The dimension of matrix E is $nf \times ng$, which means that matrix E is made up of n matrices M.

And finally, we define the matrix D^+ as follows:

$$\begin{aligned} D_{ii}^+ &= \sum_{j=1}^{n} P_{1j}^+, \quad i = 1, \ldots, f, \\ D_{ii}^+ &= \sum_{j=1}^{n} P_{2j}^+, \quad i = f+1, \ldots, 2f, \\ &\cdots \\ D_{ii}^+ &= \sum_{j=1}^{n} P_{mj}^+, \quad i = (m-1)f+1, \ldots, mf. \end{aligned} \tag{A.11}$$

The composition of the matrix D^- is the same as the composition of the matrix D^+, and the element source is the matrix P^-.

Based on the above definitions, $\mathbf{x}(t+1)$ can be further transformed:

$$\begin{aligned} \mathbf{x}(t+1) &\le \mathbf{x}(t) + K_1 A^+ \mathbf{x}(t) - \frac{K_1}{2} B^+ \mathbf{x}(t) + K_2 A^- \mathbf{x}(t) - \frac{K_2}{2} B^- \mathbf{x}(t) \\ &\quad + K_3 C^+ E\mathbf{y}(t) - \frac{K_3}{2} D^+ \mathbf{x}(t) + K_4 C^- E\mathbf{y}(t) - \frac{K_4}{2} D^- \mathbf{x}(t) \\ &\le \mathbf{x}(t) + \left(K_1 A^+ - \frac{K_1}{2} B^+ + K_2 A^- - \frac{K_2}{2} B^- - \frac{K_3}{2} D^+ - \frac{K_4}{2} D^- \right) \mathbf{x}(t) \\ &\quad + \left(K_3 C^+ E + K_4 C^- E \right) \mathbf{y}(t). \end{aligned} \tag{A.12}$$

Denote

$$\begin{aligned} S &= K_1 A^+ - \frac{K_1}{2} B^+ + K_2 A^- - \frac{K_2}{2} B^- - \frac{K_3}{2} D^+ - \frac{K_4}{2} D^-, \\ T &= K_3 C^+ E + K_4 C^- E. \end{aligned} \tag{A.13}$$

Then, we have

$$\mathbf{x}(t+1) \le \mathbf{x}(t) + S\mathbf{x}(t) + T\mathbf{y}(t). \tag{A.14}$$

(ii) Similarly, the item layer vector dynamic evolution quation 15 can also be in a similar form.

J^+ is defined as follows:

$$\begin{bmatrix} I_{11}^+ & 0 & \cdots & 0 & \cdots & I_{1n}^+ & 0 & \cdots & 0 \\ 0 & I_{11}^+ & \cdots & 0 & \cdots & 0 & I_{1n}^+ & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_{11}^+ & \cdots & 0 & 0 & \cdots & I_{1n}^+ \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ I_{n1}^+ & 0 & \cdots & 0 & \cdots & I_{nn}^+ & 0 & \cdots & 0 \\ 0 & I_{n1}^+ & \cdots & 0 & \cdots & 0 & I_{nn}^+ & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & I_{n1}^+ & \cdots & 0 & 0 & \cdots & I_{nn}^+ \end{bmatrix}, \tag{A.15}$$

$$\begin{aligned} K_{ii}^+ &= \sum_{j=1}^{n} I_{1j}^+, \quad i = 1, \ldots, g, \\ K_{ii}^+ &= \sum_{j=1}^{n} I_{2j}^+, \quad i = g+1, \ldots, 2g, \\ &\cdots \\ K_{ii}^+ &= \sum_{j=1}^{n} I_{nj}^+, \quad i = (n-1)g+1, \ldots, ng. \end{aligned}$$

L^+ is defined as follows:

$$\begin{bmatrix} Q_{11}^+ & 0 & \cdots & 0 & \cdots & Q_{1m}^+ & 0 & \cdots & 0 \\ 0 & Q_{11}^+ & \cdots & 0 & \cdots & 0 & Q_{1m}^+ & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Q_{11}^+ & \cdots & 0 & 0 & \cdots & Q_{1m}^+ \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ Q_{n1}^+ & 0 & \cdots & 0 & \cdots & Q_{nm}^+ & 0 & \cdots & 0 \\ 0 & Q_{n1}^+ & \cdots & 0 & \cdots & 0 & Q_{nm}^+ & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & Q_{n1}^+ & \cdots & 0 & 0 & \cdots & Q_{nm}^+ \end{bmatrix},$$

$$F = \begin{bmatrix} M^T & & & \\ & M^T & & \\ & & \ddots & \\ & & & M^T \end{bmatrix}, \tag{A.16}$$

$$\begin{aligned} N_{ii}^+ &= \sum_{j=1}^{m} Q_{1j}^+, \quad i = 1, \ldots, g, \\ N_{ii}^+ &= \sum_{j=1}^{m} Q_{2j}^+, \quad i = g+1, \ldots, 2g, \\ &\cdots \\ N_{ii}^+ &= \sum_{j=1}^{m} Q_{nj}^+, \quad i = (n-1)g+1, \ldots, ng. \end{aligned}$$

Similarly, the elements of J^- are made up of the elements of I^-. The elements of K^- are made up of the elements of the matrix Q^-. When the elements of the L^+ become elements of Q^-, this matrix becomes the L^-. The matrix F is made up of

multiple M^T matrices. The composition of the matrix N^- is the same as the composition of the matrix N^+, and the element source is the matrix Q^-.

Then, we have

$$\mathbf{y}(t+1) \le \mathbf{y}(t) + G\mathbf{y}(t) + H\mathbf{x}(t). \tag{A.17}$$

Here,

$$G = K_5 J^+ - \frac{K_5}{2}K^+ + K_6 J^- - \frac{K_6}{2}K^- - \frac{K_7}{2}N^+ - \frac{K_8}{2}N^-,$$
$$H = K_7 L^+ F + K_8 L^- F. \tag{A.18}$$

□

B. Proof of Theorem 2

Theorem B.1. *If appropriate parameters K_1, K_2, K_3, K_4, K_5, K_6, K_7, and K_8 make the $\theta < 0$ and $\omega < 0$, then the fixed points in equations (13) and (15) are uniformly stable.*

Proof. Since any isolated equilibrium state can be moved to the origin of the state space by coordinate transformation, we only discuss the stability of the equilibrium state at the origin of the coordinates.

Lyapunov function is defined as follows:

$$V(t) = \mathbf{x}(t)^T \mathbf{x}(t),$$
$$W(t) = \mathbf{y}(t)^T \mathbf{y}(t). \tag{B.1}$$

So, the following transformations are conducted:

$$\begin{aligned}
\Delta V &= V(t+1) - V(t) \\
&= \mathbf{x}(t+1)^T \mathbf{x}(t+1) - \mathbf{x}(t)^T \mathbf{x}(t) \\
&\le [\mathbf{x}(t) + S\mathbf{x}(t) + T\mathbf{y}(t)]^T [\mathbf{x}(t) + S\mathbf{x}(t) + T\mathbf{y}(t)] - \mathbf{x}(t)^T \mathbf{x}(t) \\
&\le \left[\mathbf{x}(t)^T + \mathbf{x}(t)^T S^T + \mathbf{y}(t)^T T^T\right] [\mathbf{x}(t) + S\mathbf{x}(t) + T\mathbf{y}(t)] - \mathbf{x}(t)^T \mathbf{x}(t) \\
&\le \mathbf{x}(t)^T \left(S + S^T + S^T S\right)\mathbf{x}(t) + \mathbf{x}(t)^T \left(T + S^T T\right)\mathbf{y}(t) + y(t)^T \left(T^T + T^T S\right)\mathbf{x}(t) + \mathbf{y}(t)^T T^T T \mathbf{y}(t).
\end{aligned} \tag{B.2}$$

Similarly,

$$\begin{aligned}
\Delta W &= W(t+1) - W(t) \\
&= y(t+1)^T \mathbf{y}(t+1) - \mathbf{y}(t)^T \mathbf{y}(t) \\
&\le [\mathbf{y}(t) + G\mathbf{y}(t) + H\mathbf{x}(t)]^T [\mathbf{y}(t) + G\mathbf{y}(t) + H\mathbf{x}(t)] - \mathbf{y}(t)^T \mathbf{y}(t) \\
&\le \left[\mathbf{y}(t)^T + \mathbf{y}(t)^T G^T + \mathbf{x}(t)^T H^T\right] [\mathbf{y}(t) + G\mathbf{y}(t) + H\mathbf{x}(t)] - \mathbf{y}(t)^T \mathbf{y}(t) \\
&\le \mathbf{y}(t)^T \left(G + G^T + G^T G\right)\mathbf{y}(t) + \mathbf{y}(t)^T \left(H + G^T H\right)\mathbf{x}(t) + \mathbf{x}(t)^T \left(H^T + H^T G\right) y(t) + \mathbf{x}(t)^T H^T H \mathbf{x}(t).
\end{aligned} \tag{B.3}$$

If appropriate parameters K_1, K_2, K_3, K_4, K_5, K_6, K_7, and K_8 make

$$\begin{aligned}
\theta &= \mathbf{x}(t)^T \left(S + S^T + S^T S\right)\mathbf{x}(t) + \mathbf{x}(t)^T \left(T + S^T T\right)\mathbf{y}(t) + \mathbf{y}(t)^T \left(T^T + T^T S\right)\mathbf{x}(t) + \mathbf{y}(t)^T T^T T \mathbf{y}(t) < 0, \\
\omega &= \mathbf{y}(t)^T \left(G + G^T + G^T G\right)\mathbf{y}(t) + \mathbf{y}(t)^T \left(H + G^T H\right)\mathbf{x}(t) + \mathbf{x}(t)^T \left(H^T + H^T G\right)\mathbf{y}(t) + \mathbf{x}(t)^T H^T H \mathbf{x}(t) < 0,
\end{aligned} \tag{B.4}$$

according to the stability theory of Lyapunov [34], the fixed points in equations (13) and (15) are uniformly stable. In fact, given the proper values of K_1, K_2, K_3, K_4, K_5, K_6, K_7, and K_8, the stability condition can be achieved.

The above theoretical analysis shows that the evolution process of the vector dynamic evaluation equations is consistent and stable. As time goes on, the state vectors of the nodes are stable and the clustering results are discovered.

In fact, the condition of $\theta < 0$ and $\omega < 0$ can be easily satisfied and we can find clustering parameters $K_1, \ldots, K_8$ with a wide value range. □

Data Availability

Firstly, the data used to support the findings of this study are all open online and we have listed in the article on page 8. (https://grouplens.org/datasets/movielens/100k/, http://www.public.asu.edu/jtang20/datasetcode/truststudy.htm, and https://www.librec.net/datasets/filmtrust.zip) Secondly, the Matlab code used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the National Natural Science Foundation of China (nos. 71561020, 61503203, 61702317, and 61771297) and Fundamental Research Funds for the Central Universities (no. GK201802013).

References

[1] M. Deshpande and G. Karypis, "Item-based top-N recommendation algorithms," *ACM Transactions on Information Systems (TOIS)*, vol. 22, no. 1, pp. 143–177, 2004.

[2] P. Zhou, Y. Zhou, D. Wu, and H. Jin, "Differentially private online learning for cloud-based video recommendation with multimedia big data in social networks," *IEEE Transactions on Multimedia*, vol. 18, no. 6, pp. 1217–1229, 2016.

[3] J.-H. Su, H.-H. Yeh, P. S. Yu, and V. S. Tseng, "Music recommendation using content and context information mining," *IEEE Intelligent Systems*, vol. 25, no. 1, pp. 16–26, 2010.

[4] L. Yao, Q. Z. Sheng, A. H. H. Ngu, J. Yu, and A. Segev, "Unified collaborative and content-based web service recommendation," *IEEE Transactions on Services Computing*, vol. 8, no. 3, pp. 453–466, 2015.

[5] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: item-to-item collaborative filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, 2003.

[6] S. H. Choi, Y.-S. Jeong, and M. K. Jeong, "A hybrid recommendation method with reduced data for large-scale application," *IEEE Transactions on Systems*, vol. 40, no. 5, pp. 557–566, 2010.

[7] K. Yu, A. Schwaighofer, V. Tresp, X. Xu, and H.-P. Kriegel, "Probabilistic memory-based collaborative filtering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 1, pp. 56–69, 2004.

[8] A. Hernando, J. Bobadilla, and F. Ortega, "A non negative matrix factorization for collaborative filtering recommender systems based on a Bayesian probabilistic model," *Knowledge-Based Systems*, vol. 97, pp. 188–202, 2016.

[9] X. Li, X. Cheng, S. Su, S. Li, and J. Yang, "A hybrid collaborative filtering model for social influence prediction in event-based social networks," *Neurocomputing*, vol. 230, pp. 197–209, 2017.

[10] D. Billsus and M. J. Pazzani, "Learning collaborative information filters," in *Proceedings of the International Conference on Machine Learning*, pp. 46–54, Madison,WI, USA, 1998.

[11] X. Ma, D. Dong, and Q. Wang, "Community detection in multi-layer networks using joint nonnegative matrix factorization," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 2, pp. 273–286, 2019.

[12] X. Ma, P. Sun, and Y. Wang, "Graph regularized nonnegative matrix factorization for temporal link prediction in dynamic networks," *Physica A: Statistical Mechanics and Its Applications*, vol. 496, pp. 121–136, 2018.

[13] X. Ling, D. Guo, F. Cai, and H. Chen, "User-based clustering with top-N recommendation on cold-start problem," in *Proceedings of the 2013 Third International Conference on Intelligent System Design and Engineering Applications*, pp. 1585–1589, Hong Kong, China, January 2013.

[14] J. D. West, I. Wesley-Smith, and C. T. Bergstrom, "A recommendation system based on hierarchical clustering of an article-level citation network," *IEEE Transactions on Big Data*, vol. 2, no. 2, pp. 113–123, 2016.

[15] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[16] S. Zahra, M. A. Ghazanfar, A. Khalid, M. A. Azam, U. Naeem, and A. Prugel-Bennett, "Novel centroid selection approaches for K means-clustering based recommender systems," *Information Sciences*, vol. 320, pp. 156–189, 2015.

[17] J. Wu, Y. Hou, Y. Jiao, Y. Li, X. Li, and L. Jiao, "Density shrinking algorithm for community detection with path based similarity," *Physica A: Statistical Mechanics and Its Applications*, vol. 433, pp. 218–228, 2015.

[18] L. Bai, J. Liang, H. Du, and Y. Guo, "A novel community detection algorithm based on simplification of complex networks," *Knowledge-Based Systems*, vol. 143, pp. 58–64, 2018.

[19] J. Wu and Y. Jiao, "Clustering dynamics of complex discrete-time networks and its application in community detection," *Chaos*, vol. 24, Article ID 033104, 2014.

[20] J. Wu, F. Wang, and P. Xiang, "Automatic network clustering via density-constrained optimization with grouping operator," *Applied Soft Computing*, vol. 38, pp. 606–616, 2016.

[21] J. Chen, H. Wang, L. Wang, and W. Liu, "A dynamic evolutionary clustering perspective: community detection in signed networks by reconstructing neighbor sets," *Physica A: Statistical Mechanics and Its Applications*, vol. 447, pp. 482–492, 2016.

[22] J. Chen, l. Zhang, W. Liu, and Z. Yan, "Community detection in signed networks based on discrete-time model," *Chinese Physics B*, vol. 26, no. 1, Article ID 018901, 2017.

[23] Z. Bu, H.-J. Li, J. Cao, Z. Wang, and G. Gao, "Dynamic cluster formation game for attributed graph clustering," *IEEE Transactions on Cybernetics*, vol. 49, no. 1, pp. 328–341, 2019.

[24] P. Moradi, S. Ahmadian, and F. Akhlaghian, "An effective trust-based recommendation method using a novel graph clustering algorithm," *Physica A: Statistical Mechanics and Its Applications*, vol. 436, pp. 462–481, 2015.

[25] J. Ren, J. Long, and Z. Xu, "Financial news recommendation based on graph embeddings," *Decision Support Systems*, vol. 125, Article ID 113115, 2019.

[26] T. Liang, L. Zheng, L. Chen, Y. Wan, P. S. Yu, and J. Wu, "Multi-view factorization machines for mobile app recommendation based on hierarchical attention," *Knowledge-Based Systems*, vol. 187, Article ID 104821, 2020.

[27] C. Feng, Z. Liu, S. Lin, and T. Q. S. Quek, "Attention-based graph convolutional network for recommendation system," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 7560–7564, Brighton, UK, April 2019.

[28] W. Shang, S. Shang, S. Feng, and M. Shi, "An improved video recommendations based on the hyperlink-graph model," in *Proceedings of the 2016 4th International Conference on Applied Computing and Information Technology/3rd International Conference on Computational Science/Intelligence and Applied Informatics/1st International Conference on Big Data, Cloud Computing, Data Science and Engineering*, pp. 379–383, Las Vegas, NV, USA, December 2016.

[29] Y. Yasami, "A new knowledge-based link recommendation approach using a non-parametric multilayer model of dynamic complex networks," *Knowledge-Based Systems*, vol. 143, pp. 81–92, 2018.

[30] H. Ebbinghaus, H. A. Ruger, and C. E. Bussenius, "Memory: a contribution to experimental psychology," *Journal of Annals of Neurosciences*, vol. 20, no. 4, p. 155, 2013.

[31] J. Shao, X. He, Q. Yang, C. Bohm, and C. Plant, "Synchronization inspired partitioning and hierarchical clustering,"

IEEE Transactions on Knowledge and Data Engineering, vol. 25, pp. 893–905, 2013.

[32] J. Chen, H. Uliji, and C. Zhao, "Collaborative rating prediction based on dynamic evolutionary heterogeneous clustering," *Bio-Inspired Computing—Theories and Applications*, vol. 682, pp. 394–399, 2017.

[33] J. Wu, L. Jiao, J. Chao, F. Liu, M. Gong, and R. Shang, "Overlapping community detection via network dynamics," *Physical Review E*, vol. 85, no. 1, Article ID 016115, 2012.

[34] G. Teschl, "Ordinary differential equations and dynamical systems," *Atlantis Studies in Differential Equations*, vol. 140, no. 3, pp. 189–194, 2004.

[35] U. Liji, Y. Chai, and J. Chen, "Improved personalized recommendation based on user attributes clustering and score matrix filling," *Computer Standards and Interfaces*, vol. 57, pp. 59–67, 2018.

[36] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

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

[38] F. Shang, Y. Liu, J. Cheng, and D. Yan, "Fuzzy double trace norm minimization for recommendation systems," *IEEE Transactions on Fuzzy Systems*, vol. 26, no. 4, pp. 2039–2049, 2018.