Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 9815423, 1 page
https://doi.org/10.1155/2023/9815423

*Retraction*

# Retracted: Research on the Spectral Domain Graph Convolution Collaborative Filtering Algorithm Based on Reinforcement Learning and Chebyshev

**Wireless Communications and Mobile Computing**

Received 27 June 2023; Accepted 27 June 2023; Published 28 June 2023



This article has been retracted by Hindawi following an investigation undertaken by the publisher [1]. This investigation has uncovered evidence of one or more of the following indicators of systematic manipulation of the publication process:

(1) Discrepancies in scope
(2) Discrepancies in the description of the research reported
(3) Discrepancies between the availability of data and the research described
(4) Inappropriate citations
(5) Incoherent, meaningless and/or irrelevant content included in the article
(6) Peer-review manipulation

The presence of these indicators undermines our confidence in the integrity of the article's content and we cannot, therefore, vouch for its reliability. Please note that this notice is intended solely to alert readers that the content of this article is unreliable. We have not investigated whether authors were aware of or involved in the systematic manipulation of the publication process.

Wiley and Hindawi regrets that the usual quality checks did not identify these issues before publication and have since put additional measures in place to safeguard research integrity.

We wish to credit our own Research Integrity and Research Publishing teams and anonymous and named external researchers and research integrity experts for contributing to this investigation.

The corresponding author, as the representative of all authors, has been given the opportunity to register their agreement or disagreement to this retraction. We have kept a record of any response received.

## References

[1] S. Wang, C. Zeng, B. Song, X. Huang, and S. Zhou, "Research on the Spectral Domain Graph Convolution Collaborative Filtering Algorithm Based on Reinforcement Learning and Chebyshev," *Wireless Communications and Mobile Computing*, vol. 2022, Article ID 7787633, 11 pages, 2022.

Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7787633, 11 pages
https://doi.org/10.1155/2022/7787633





*Research Article*

# Research on the Spectral Domain Graph Convolution Collaborative Filtering Algorithm Based on Reinforcement Learning and Chebyshev

**Song Wang, Cheng Zeng, Bailing Song, Xuxiang Huang, and Sicheng Zhou**

*School of Computer Science and Information Engineering, Hubei University, Wuhan 430062, China*

Correspondence should be addressed to Sicheng Zhou; 201931119020161@stu.hubu.edu.cn

Received 15 June 2022; Revised 1 July 2022; Accepted 13 July 2022; Published 8 August 2022

Academic Editor: Mohammad Farukh Hashmi



To solve the abovementioned problem, we propose a collaborative filtering recommendation algorithm that incorporates singular value decomposition (SVD) and Chebyshev truncation in spectral domain convolution. Firstly, the SVD algorithm is used to optimize the adjacency matrix, mine the potential association information between users and items, and expand the user-item adjacency matrix. Finally, based on the MovieLens-1M public dataset, the proposed algorithm (CBSVD-SCF) is compared with other commonly used algorithms. The results show that the article optimizes the recommendation effect of the algorithm based on the traditional collaborative filtering algorithm by combining the temporal order and sequence of user interaction information, as well as the popularity of items and the activity of users; the experimental results on MovieLens show that the optimized collaborative filtering recommendation algorithm can effectively improve the recommendation effect.

## 1. Introduction

Collaborative filtering (CF) [1] is one of the widely used and prominent techniques in RS, and the basic assumption of the CF method is that if user u1 shares a common item with another user u2, u1 may also be as interested in the project as u2. Although CF has been successfully applied to many recommendation applications, the cold start problem is still considered to be one of its main challenges. The problem arises when the user interacts with very few items [2]. Therefore, when users share few items with other users, making effective recommendations to users becomes a challenging task for RS [3].

If the relationship between users and items is represented as a bifurcated graph, this paper argues that the connectivity information present in the graph structure can play an important role in solving the cold-start problem [4]. For example, the bifurcated graph B in Figure 1, cold-start user u1 interacts only with item i1. In this paper, we use the term "graph" to refer to the graph/network structure of the data and "network" to refer to the structure of the machine learning model [5]. Since u1 shares i2 with user u2 and user u3, all three items (i2, i3, and i4) connected to u2 or u3 can be recommended to u1 by the cf-based model. However, a naturally important question arises: which of these three items is the most reliable one to recommend to u1, and the key to answering this question lies in the information about the user's connection to the item [6]. In fact, if you look at the connections of the graph, you will see that there is only one path between u1 and i2 (or i3) and two paths between u1 and i4. Therefore, it is clear that i4 is a more reliable recommendation for u1 compared to i2 and i3.

However, existing cf-based methods, including model-based and memory-based methods, often have difficulty modeling connection information [7]. Previous model-based methods, such as matrix decomposition (MF) [8], are usually designed to approach direct connections (or proximity points). However, the indirect connection information hidden in the graph structure is rarely captured by traditional model-based approaches. For example, modeling the number of paths between u1 and i4 in Figure 1 is difficult. Although many memory-based methods [9] have been introduced to model connection information, these methods

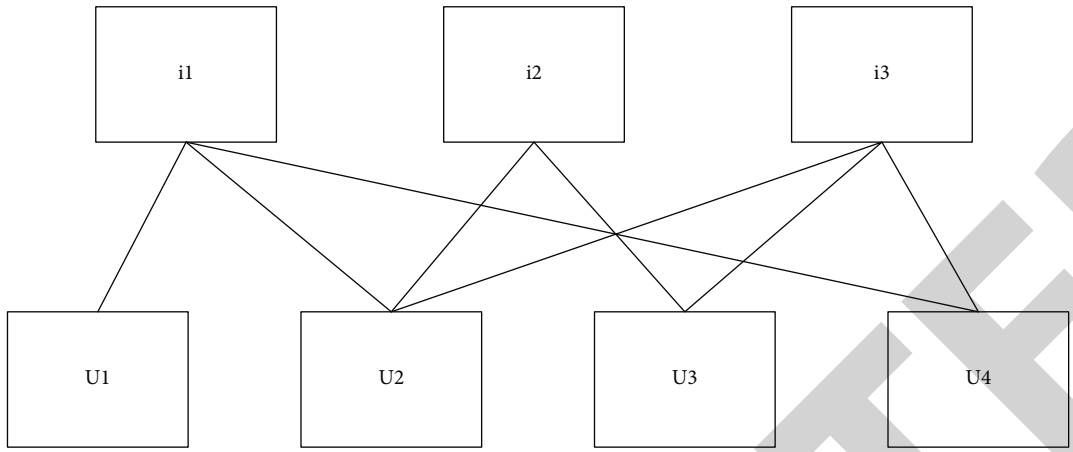


Figure 1: User-item bipartite graph.

often rely on predefined similarity functions. However, in the real world, it is not easy to define a similarity function that is suitable for different application situations [10].

Inspired by the latest progress in node/graph classification methods [11], this paper is based on the method of the spectral graph theory and uses the multivariate implicit information in the field of graphs to overcome the above-mentioned shortcomings and challenges. Specifically, in order to overcome the difficulty of learning recommendation directly from the spectral domain, this paper proposes a new spectral convolution operation, which is approximated by the Chebyshev first-order truncation and dynamically amplifies or attenuates each frequency domain [12]. Then, this paper introduces a recommendation model: multivariate fusion spectral convolution collaborative filtering (CBSVD-SCF), as shown in Figure 1.

## 2. Related Work

*2.1. Research Status of the Recommendation System.* With the rise of personalized recommendation business, recommendation systems have attracted the attention of various industries and academia [13]. Since the early 1990s, the theory and research methods of recommendation systems have been developed and improved and knowledge from many fields such as cognitive science, management science, approximation theory, and prediction theory has been widely used in their research [14]. Precipitated and accumulated, recommendation systems have formed a popular topic integrating multiple dimensions of information retrieval, information filtering, and management prediction.

A large amount of research literature has been accumulated academically, and fruitful results have been achieved, making it an important research element for e-commerce and personalized search. The earliest recommendation systems date back to 1994. The GroupLens lab at the University of Minnesota designed the first automated recommendation system, GroupLens [15]. This problem established a formal model that had a huge impact on the later development of recommender systems. By 1997, the GroupLens research group had created a noncommercial MovieLens movie recommendation system, which collects information about users' movie viewing history and ratings through a website and combines it with collaborative filtering techniques to recommend movies that may be of interest to users but have not been seen [16]. To facilitate the research of recommendation algorithms by many scholars, the research group released the MovieLens dataset, the dataset used in the experiments [17]. This dataset contains user ratings and tagging data and later added information such as timestamps. It is one of the most cited datasets in the field of recommendation systems. Despite the continuous maturation and improvement of recommendation algorithms and the expansion of the application areas of the recommendation theory, there are still many problems that hinder the further development of recommendation systems [18]:

(1) Data sparsity and cold start problems. The recommendation system completes the recommendation based on the user's hobbies and behavior information. However, the number of users of e-commerce websites is basically in the millions, so the rating matrix established by the recommendation system will be very huge, and these millions of users are not enough. It is possible to evaluate tens of thousands of products, or only a small number of them; without the historical visit data and behavioral information of a large number of users, the recommendation system cannot make accurate recommendations. The sparser the data matrix, the lower the efficiency of the recommendation system, and when new users enter the system, there is no historical information, which will cause the cold start problem [19]

(2) The problem of dynamic timeliness. The recommendation system will analyze and recommend according to the user's historical information, but the interests of most users will change over time. When users have new interests, users hope that the recommendation system can accurately capture their own interests. Do not repeat recommendations for old hobbies, which can reduce the effectiveness of

recommendations. Also, users expect recommendation algorithms to discover their potential sides and surprise them [20]

(3) Privacy and security issues. Recommendation systems have such a contradictory problem. On the one hand, due to the sparseness of data, we hope to obtain more data from users; on the contrary, to obtain more and more personal data of users, revealing personal privacy and security, the potential hidden problems are more likely to appear. Therefore, how to solve data sparsity while protecting user privacy and security is a question worth weighing

*2.2. Recommendation Algorithm Based on Reinforcement Learning.* Reinforcement learning is one of three machine learning methods alongside supervised learning and unsupervised learning. Supervised learning: it uses features and labels, even if the data is unlabeled, the relationship between features and labels can be learned to determine the label-classification. Unsupervised learning: it uses only features, not labels. The training dataset with only features and no labels is clustered into several classes by the intrinsic connection and similarity between the data. Some characteristics are learned from the data based on some metric according to the characteristics of the data itself. Reinforcement learning is similar to semisupervised learning in that both use unlabeled data, but reinforcement learning makes use of incentive and penalty functions by algorithmically learning whether it is getting closer to the goal. Supervised learning trains the model through labeled input and output sample sets; unsupervised learning trains the model by analyzing hidden features between unlabeled data; reinforcement learning also trains through unlabeled data. Unlike unsupervised learning, reinforcement learning has a delay reward value. The trained model is dynamic, reinforcement learning finds a balance between exploring unknown areas and utilizing current knowledge, and maximizes the cumulative reward by optimizing the agent's strategy for taking actions in the environment. Meanwhile, the training data for supervised and unsupervised learning are generally independent, while there may be a sequential relationship between the training data for reinforcement learning. Depending on the usage of the environment model, model-based reinforcement learning can be divided into three categories:

(1) As a new data source: the environment model and the agent interact to generate data, which is used as an additional training data source to supplement the training of the algorithm

(2) Increasing the contextual information of decision-making: during the $Q$-value estimation, the environment model and the agent interact and the information in the interaction process is provided to the agent as contextual information to help in its decision-making

(3) Increasing the quality of $Q$-value estimation: when performing $Q$-value estimation, a sequence of a certain length will be expanded through the environment model, and then, combined with the $Q$-value estimation of model-free reinforcement learning, a more accurate estimation will be given

The following are four typical model-based reinforcement learning methods:

(1) In the world model approach, agents learn with the help of variable autoencoders

(2) The imaginative augmentation method extracts implicit features and interprets predictions from a model of the learning environment by using the prediction results as a perceptual background in a deep policy network. The method combines model-based and model-free approaches and is therefore a hybrid learning method

(3) Model-based prior results are used in model-free reinforcement learning methods, which are aimed at bridging the gap between model-free reinforcement learning and model-based reinforcement learning

(4) The model-based value expansion method, which controls the stability of the model by limiting the depth of the model network, adds a dynamic model to the model-free reinforcement learning method, improves the value function calculation process, and reduces the sample complexity of learning, to speed up the convergence

*2.3. Chebyshev Polynomials.* Chebyshev polynomials have always been a research hotspot, and many good properties have been found, such as orthogonality, parity, boundedness, and completeness; many identities have been generated, and some product-sum formulas have been obtained. Preliminary research results have also been made on the recurrence relations, fixed points, and equations (groups) composed of Chebyshev polynomials. Scheffe-type equations have also appeared in more in-depth studies.

*Definition 1.* Let $p, q$ be a nonzero constant, and $U_{p,q,n}(x)(0 < n \in N)$ satisfies the following second-order linear push sequence:

$$\begin{aligned} U_{p,q,0}(x) &= 1, \\ U_{p,q,n}(x) &= 2px, \\ U_{p,q,n+1}(x) &= 2pxU_{p,q,n}(x) - qU_{p,q,n-1}(x). \end{aligned} \tag{1}$$

Obviously, when $p = 1$, $q = 1$, the Chebyshev polynomial is $U_n(x) = U_{1,1,n}(x)$.

*Definition 2.* Let the convolution of $U_{p,q,n}(x)$ be

$$H(x,k,n,p,q) = \sum_{a_1+a_2+\cdots+a_k=n} U_{p,q,a1}(x)U_{p,q,a2}(x)\cdots U_{p,q,ak}(x). \tag{2}$$

When $p=1$, $q=1$, the convolution of $U_n(x)$ is

$$H(x,k,n) = \sum_{a_1+a_2+\cdots+a_k=n} U_{a1}(x)U_{a2}(x)\cdots U_{ak}(x). \tag{3}$$

When $k$ is a positive integer and $n$ is a non-negative integer, $\sum_{a_1+a_2+\cdots+a_k=n}$ represents the sum of all different non-negative integer groups $a_1+a_2+\cdots+a_k$ of $a_1+a_2+\cdots+a_k=n$.

*Property 1.* Let $U_{p,q,n}(x)$ be the polynomial in Definition 2; then, its general term and generating function are as follows:

$$U_{p,q,n}(x) = \frac{\left(a^{n+1}-\beta^{n+1}\right)}{(a(x)-\beta(x))}, \qquad \frac{1}{1-2pxt+qt^2} = \sum_{n=0}^{\infty} U_{p,q,n}(x)^{t^n}. \tag{4}$$

Then, $a(x)=px+\sqrt{p^2x^2-q}$, $\beta(x)=px-\sqrt{p^2x^2-q}$, $a(x)-\beta(x)=2\sqrt{p^2x^2-q}a(x)\beta(x)=q$.

## 3. Chebyshev Collaborative Filtering Recommendation Algorithm Based on Reinforcement Learning Optimization

Guided by the specific learning needs of learners, through the description of learners' personality characteristics, the identification of learning needs, and the provision of personalized recommendation services for educational resources, it essentially reflects the modern educational concept of taking learners as the center, which is also the development basis of personalized recommendation services for educational resources under the current situation of mass generalized learning. In the context of the "Internet +" information age, according to the personalized needs of users, many researchers in the fields of information science, education management, and so on have carried out a lot of theoretical and practical exploration, through the collection, collation, induction, and classification of information resources, and recommend to potential users the education resource data that they may need and the specific process of resource push.

An overview of the research results of personalized recommendation services of educational resources in the fields of artificial intelligence and educational management shows that it mainly focuses on the introduction of different algorithms and application platforms into the push service for the analysis of learners' resource needs, including the application of the latest big data processing technologies such as personalized learning resource navigation, learners' cognitive modeling, semantic framework analysis, and intelligent agents. However, these studies generally lack effective solutions to the dynamic changes in the process of educational resource recommendation.

This can be seen from the following two aspects. First, at present, educational resources are very popular and the update iteration speed is fast. The past resources will soon be eliminated, and emerging resources will quickly occupy the highland. And users' demand for resources will gradually vary with the passage of time. Second, most recommendation algorithms generally take user clicks and other operations as feedback information but the interval between users returning to resource services can also convey the user's preference for the resources recommended by the recommendation algorithm.

The lack of consideration and research on learners' personal information characteristics and behavior record information generated in the network interaction system leads to insufficient research on learners' personal attribute characteristics, insufficient analysis of learners' interests and preferences, and inaccurate personalized recommendation services of educational resources. Therefore, based on the practical experience of extensive application of reinforcement learning in finance, the Internet, and other fields, this paper introduces reinforcement learning into the field of personalized recommendation service of educational resources, adds some randomness when searching for new resources, avoids users' fatigue of similar resources, and provides users with the resources that they need more accurately. It provides reference and development ideas for the research on personalized recommendation service of educational resources under the current background.

*3.1. Recommender System Algorithm Based on Reinforcement Learning.* Therefore, we propose a deep learning framework that can better solve the problem of educational resource recommendation. The framework can complete the construction of the user portrait model with the help of the reinforcement learning algorithm based on the basic fixed data and dynamic change data of learners, so as to achieve more accurate personalized recommendation service of educational resources. In the process of building the user portrait model, a vector model is established to scientifically and effectively describe the data information describing the resource needs of learners and calculate the weight.

At the same time, we consider the current and future rewards and avoid using split user operation data to specifically describe the user's state. These two points provide a basic guarantee for the application of large-scale recommendation systems. In addition, we regard the user's reward as feedback information and consider the multiple intervals of the user returning to the recommendation service, so as to realize the user modeling at any time, so as to better evaluate the user's activities and maintain the user's model. Moreover, the gradient descent is also fully utilized in our recommendation system. The direction of the maximum value of the directional derivative on the surface represents the direction of the gradient, so when

we do the gradient descent. Updating the weight along the opposite direction of the gradient can effectively find the global optimal solution, avoid irrelevant recommendation information, and improve the performance and accuracy of the recommendation system, as shown in Figure 2.

The results of our work can be summarized as follows:

(1) We propose a resource recommendation framework combined with the DQN algorithm, which can realize real-time processing and recommendation. At the same time, the system has good scalability and portability and can be fully used in other fields, not just applied to online resource recommendation

(2) Our system takes into account multiple time intervals when users return to the recommendation service, which is more efficient and reliable than the traditional method of considering only the content of a certain part of the website that users click

(3) We introduce gradient descent and update the weight along the opposite direction of the gradient, which can effectively find the global optimal solution, avoid irrelevant recommendation information, and improve the performance and accuracy of the recommendation system

### 3.2. Model

*3.2.1. DQN Algorithm.* DQN includes three elements: state, action, and reward. The reward value statically describes the immediate reward value for the transfer between states, and the action determines the transfer rules between states. During Q-learning iteration, the immediate reward value, $Q$-value function, and discount rate are used to form the evaluation function. The $Q$-value table stores the estimated values of each state action $(s, a)$. For the Q-learning algorithm under a given strategy $h(x)$, the evaluation function of taking action $A_t$ in state $S_t$ is as follows:

$$Q(S_t, A_t) \longleftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max \alpha Q(S_{t+1}, \alpha) - Q(S_t, A_t)], \tag{5}$$

where $\alpha \in (0, 1)$ is the learning step; $\gamma \in (0, 1]$ is the discount rate, which determines the weight of the agent to consider the future reward; $t$ is the time step; it is the immediate reward when taking the current reward $(S_t, A_t)$; the max function indicates that the algorithm will evaluate according to the maximum value of the predicted value in the next one $(S_{t+1}, A_{t+1})$. In the formula $R_{t+1} + \gamma \max \alpha Q(S_{t+1}, \alpha) - Q(S_t, A_t)$, it is defined as the time difference error (TD error) and the algorithm updates the estimated value incrementally through TD error until it converges. Action selection is often used as the $\varepsilon$-greedy strategy. The whole process of transferring an agent from the start state to the target state is called an episode, in which the time of each state transition is called a timestep.

The process of the deep Q-learning algorithm is as follows:

(1) First, initialize the "memory D" which is called D for short. Its capacity is $N$. Initialize the $Q$ network and randomly generate weights $\omega$; initialize the target $Q$ network with a weight of $\omega - = \omega$; loop through episode $= 1, 2, \cdots, M$; initialize initial state $S_1$

(2) Loop traversal step $= 1, 2, \cdots, t$; use $\varepsilon$-greedy policy to generate action at (with $\varepsilon$ probability, select a random action or select at $= \max aQ(S_t, a; \omega)$

(3) Execute the action at time $t$ receive the reward at time $t$ and the state at time $t + 1$

(4) Store the transition sample $(S_t, a_t, r_t, S_{t+1})$ in D

(5) Randomly select a minipatch transition $(S_j, a_j, r_j, S_{j+1})$ from D

(6) If $j + 1$ is terminal, make $y_j = r_j$; otherwise, make $y_j = r_j + \gamma \max a' Q(S_{j+1}, a'; \omega -)$

(7) $y_j - Q(S_j, a_j; \omega)2$ about $\omega$ uses the gradient descent method to update

(8) Update the target $Q$ network by $\omega - = \omega$ every $C$ step

(9) The optimal strategy $h * (x)$ of the output original problem is to greedily select the action with the largest $Q$ value in each state

*3.2.2. DDQN Algorithm.* The DDQN network structure is the same as DQN, and it also has the same two $Q$ network structures. Based on DQN, the problem of overestimation is eliminated by decoupling the selection of the target $Q$ value action and the calculation of the target $Q$ value. In DDQN, instead of directly finding the maximum $Q$ value of each action in the target $Q$ network, first, find the action corresponding to the maximum $Q$ value in the current $Q$ network, that is,

$$a \max \left(S_j', w\right) = \arg \max a' \left(\varphi\left(S_j'\right), a, w\right). \tag{6}$$

Then, use this selected action $a \max (S'_j, w)$ to calculate the target $Q$ value in the target network, namely,

$$y_j = R_j + \gamma Q' \left(\varphi\left(S_j'\right), a \max \left(S_j', w\right), w'\right). \tag{7}$$

Put it all together:

$$y_j = R_j + \gamma Q' \left(\varphi\left(S_j'\right), \arg \max a' \left(\varphi\left(S_j'\right), a, w\right), w'\right). \tag{8}$$

The DDQN algorithm flow is as follows:

(1) The values corresponding to all states and actions are initialized randomly, and all parameters $w$ of the

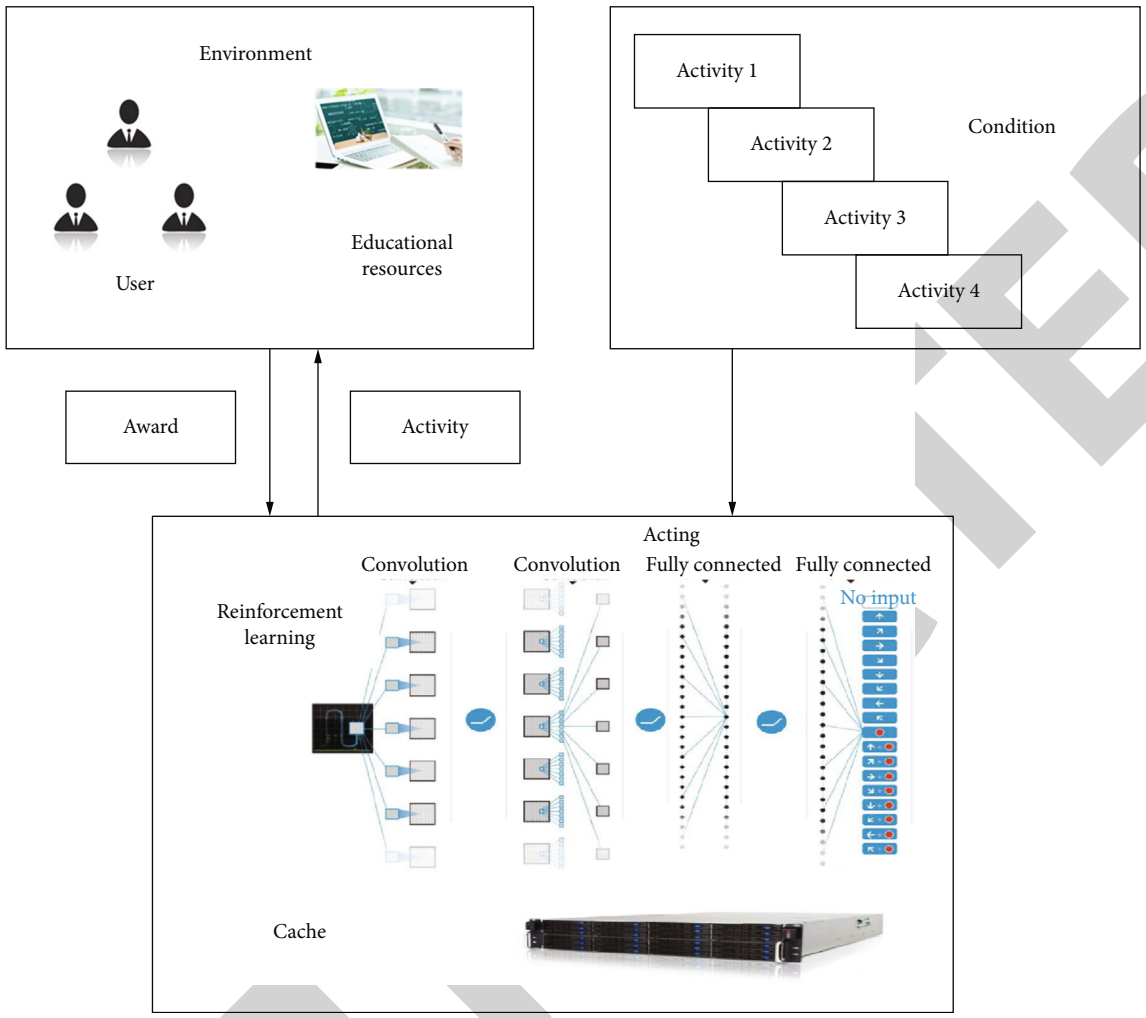


Figure 2: Deep reinforcement recommendation system.

current $Q$ network and the parameters $w' = w$ of the target $Q$ network $Q'$ are initialized randomly. Clear experience playback set D

(2) Iterate the following

(i) Initialize $S$ as the first state of the current state sequence and get its eigenvector $\varphi(S)$

(ii) Use as input $\varphi(S)$ in the network to get the value output corresponding to all actions of the $Q$ network. Use $\varepsilon$-greedy policy to select the corresponding action A in the current $Q$ value output

(iii) Execute the current action $A$ in state $S$ to get the feature vector $\varphi(S')$ and reward $R$\$ corresponding to the new state $S'$ and whether to terminate the state

(iv) Store the quintuple $\{\varphi(S), A, R, \varphi(S'), \text{is\_end}\}$ into experience playback set D; $S = S'$

(v) Sample $m$ samples $\{\varphi(S_j), A_j, R_j, \varphi(S_j'), \text{is\_end}j\}, j = 1, 2, \cdots, m$ from the experience playback set D, and calculate the current target $Q$ value $y_j$:

$$y_j = \left\{R_j R_j + \gamma Q'\left(\varphi\left(S_j'\right), \arg\max a' Q\left(\varphi\left(S_j'\right), a, w\right), w'\right)\right\} \tag{9}$$

(vi) Using the mean square loss function $1/m\sum j = 1/m(y_j - Q(\varphi(S_j), A_j, w))2$, all parameters $w$ of the $Q$ network are updated by gradient back propagation of the neural network

(vii) If $T\%C = 1$, update the target $Q$ network parameter $\omega' = \omega$

(viii) If $S'$ is in the termination state, the current round of iteration is completed; otherwise, go to step (ii)

*3.2.3. Spectral Convolution Process.* The process of convolution on the graph is to convert spatial signals to the spectral domain, and then, multiply and convert the results to the spatial domain. A convolution filter is proposed to enter the spectral field from $g_\theta(\Lambda) = \text{diag}\,[(\theta_0\lambda_0, \theta_1\lambda_1, \cdots, \theta_{N-1}\lambda_{N-1})]$, and the format is as follows:

$$g_\theta * x = \begin{bmatrix} x^u_{\text{new}} \\ x^i_{\text{new}} \end{bmatrix} = U g_\theta(\Lambda) \begin{bmatrix} \widehat{x}^u \\ \widehat{x}^i \end{bmatrix} = U g_\theta(\Lambda) U^T \begin{bmatrix} x^u \\ x^i \end{bmatrix} \begin{bmatrix} \widehat{x}^u \\ \widehat{x}^i \end{bmatrix}, \tag{10}$$

where $x^u_{\text{new}}$ and $x^i_{\text{new}}$ are the new signals learned through the filter $g_\theta(\Lambda)$ on the bipartite graph $G$.

$\Lambda = \{\lambda_0 \ , \lambda_1, \cdots, \lambda_{N-1}\}$ represents the eigenvalue of Laplace matrix $L$ in the figure. In equation (10), a convolution filter $g_\theta(\Lambda)$ is placed on a spectrogram signal $\begin{bmatrix} \widehat{x}^u \\ \widehat{x}^i \end{bmatrix}$ and each $\theta$'s value is responsible for increasing or decreasing each corresponding frequency component.

*3.2.4. Chebyshev Polynomial Approximation.* For large graphs, it may be very expensive to calculate the feature decomposition of $L$ first. To avoid this problem, Chebyshev polynomials are used. Chebyshev polynomial $Tk(k)$ to order $k$ is as follows:

$$g'_\theta(\Lambda) \approx \sum_{K=0}^{K} \theta'_K Tk\left(\tilde{\Lambda}\right)\lambda_{\max}, \tag{11}$$

where $\tilde{\Lambda} = 2\Lambda/\lambda_{\max} - I$, $\lambda_{\max}$ represents the maximum characteristic value of $L$. The Chebyshev polynomial is recursively defined as $T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x)$; let $T_0(x) = 1$, $T_1(x) = x$ return to the convolution definition of signal $x$ and filter $g'_\theta(\Lambda)$, including

$$g'_\theta(\Lambda) \approx \sum_{K=0}^{K} \theta'_K T_K\left(\tilde{L}\right)x. \tag{12}$$

*3.2.5. Chebyshev Simplification.* Only the Chebyshev polynomial of order 1 is considered, and each convolution kernel has only one parameter; then,

$$g_\theta(\Lambda) = \sum_{k=0}^{1} \theta'_k\left(\tilde{\Lambda}\right). \tag{13}$$

The formula after convolution of the spectral domain map is as follows:

$$g_{\theta'} * x = \sum_{k=0}^{1} \theta'_k T_k\left(\tilde{L}\right)x = \left[\theta_0 - \theta_1\left(D^{-(1/2)}WD^{-(1/2)}\right)\right]x. \tag{14}$$

Further simplify, so that each convolution kernel has only one learnable parameter. Making $\theta_0 = \theta_1 = \theta$, then, you get

$$\begin{aligned} g_{\theta'} * x &= \sum_{k=0}^{1} \theta'_k T_k\left(\tilde{L}\right)x = \left[\theta_0 - \theta_1\left(D^{-(1/2)}WD^{-(1/2)}\right)\right]x \\ &= \left[\theta\left(D^{-(1/2)}WD^{-(1/2)} + I\right)\right]x. \end{aligned} \tag{15}$$

$D^{-}\ (1/2)\ WD^{-(1/2)}\ + I$ has the eigenvalue with range [0,2]. If this operator is used in the deep neural network model, repeated application of this operator will lead to numerical instability (divergence) and gradient explosion/disappearance. In order to solve this problem, a renormalization trick is introduced, so $D^{-(1/2)}WD^{-(1/2)} + I = D^{-(1/2)}WD^{-(1/2)}$ and the final formula is as follows:

$$g_{\theta'} * x = \left[\theta\left(D^{-(1/2)}WD^{-(1/2)}\right)\right]x. \tag{16}$$

*3.2.6. Learnable Parameter Optimization.* The input $x^u \in R^{|V|\times c}$ and $x^i \in R^{|I|\times 1}$ of users and projects are transformed into a $C$-dimensional graph signal $x^u \in R^{|V|\times c}$ and $x^i \in R^{|I|\times c}$, and then, consider extending the convolution filter into a convolution filter matrix $\theta^i \in R^{C\times F}$ with $C$ input channels and $F$ filters. The final spectral convolution operation is as follows:

$$\begin{bmatrix} x^u_{\text{new}} \\ x^i_{\text{new}} \end{bmatrix} = g_{\theta'} * x = \sigma\left(D^{-(1/2)}WD^{-(1/2)}\begin{bmatrix} x^u \\ x^i \end{bmatrix}\Theta'\right), \tag{17}$$

where $x^u \in R^{|V|\times c}$ and $x^i \in R^{|I|\times c}$ represent the convolution results learned from the spectral domain of the user and the project using the $F$ filter, respectively.

*3.2.7. Multilayer Stack.* Given the user vector $x^u$ and item vector $x^i$, the new graph signal $x^u_{\text{new}}$ is the convolution result learned from the spectral domain through the parameter matrix $\Theta \in R^{C\times F}$ in formula (16). First, the user vector $x^u_0$ and item vector $x^i_0$ are randomly initialized as inputs and the $k$-layer depth spectrum CF can be expressed as follows:

$$\begin{bmatrix} x^u_K \\ x^i_K \end{bmatrix} = \text{mul}\left(\cdots\text{mul}\left(\begin{bmatrix} x^u_0 \\ x^i_0 \end{bmatrix}; D, W, \Theta'_0 \cdots; D, W, \Theta'_{k-1}\right)\right), \tag{18}$$

where $\Theta'_{k-1} \in R^{F\times F}$ is the filter parameter matrix of layer $K$ and $x^u_K$ and $x^i_K$ represent the convolution filtering result of layer $K$.

In order to take advantage of the characteristics of all layers from Chebyshev-SCF, they are further connected to the final potential factors of users and projects, namely,

$$\begin{aligned} \Phi^u &= [X^u_0, X^u_1, \cdots, X^u_k], \\ \Phi^i &= [X^i_0, X^i_1, \cdots, X^i_k]. \end{aligned} \tag{19}$$

Among the abovementioned, $\Phi^u \in R^{|u|\times(C+KF)}$, $\Phi^i \in R^{|i|\times(C+KF)}$.

In terms of loss function, the conventional BPR loss recommended in literature [21] is adopted. BPR is an unbiased pairwise loss function for implicit recommendation, which is different from the pointwise loss function. A triple $(r, j, j')$ is generated through BPR, where item $j$ indicates that it has been liked/clicked/viewed by user $r$, while the item $j'$ indicates the opposite.

By maximizing the preference difference between $j$ and $j'$, given a user matrix $I^u$ and an item matrix $I^i$, the loss function of CBSVD-SCF is as follows:

$$L = \arg\min \left(I^u, I^i\right) \sum_{(r,j,j')} -\ln \sigma(\text{pos} - \text{neg}) + \lambda\left(I^u + I^i\right), \tag{20}$$

where pos and neg represent the potential factors of users and projects and $\lambda$ represents the coefficient of the regularization term. The training data are as follows:

$$\Gamma = \left\{ \left(r, j, j'\right) \middle| r \in U \wedge j \in I_i^+ \wedge j' \in I_i^- \right\}. \tag{21}$$

## 4. Experimental Analysis

*4.1. Experimental Data.* The experimental dataset is Book-Crossing, a book rating dataset written by Cai-Nicolas Ziegler based on data from http://bookcrossing.com. It contains 1.1 million ratings for 270000 books from 90000 users. Ratings range from 1 to 10, including explicit and implicit ratings. The Book-Crossing dataset is one of the least dense datasets and the least dense dataset with a clear rating. The users in the experimental dataset only marked liking for some resources and did not have specific ratings, so here, the interest value of the item that has given positive feedback is set to 1; otherwise, it is 0 and the user similarity can be determined by the Jaccard similarity method or cosine similarity calculation method metric. The formula for setting the user hit rate is as follows:

$$\text{Click through rate} = \frac{\text{number of hits}}{\text{number of total}}. \tag{22}$$

The precision formula is as follows:

$$P = \frac{\text{number of hits in top } k \text{ items}}{k}. \tag{23}$$

nDCG indicates the standard normalized discount cumulative gain, where $r$ is the rank of these items of the list, $n$ is the length of the item list, $f$ is function or algorithm for ranking, $y_r^f$ is used to show whether to click, and $D(r)$ is the discount.

$$\text{DCG}(f) = \sum_{r=1}^{n} y_r^f D(r). \tag{24}$$

Table 1: Parameter settings.

| Variables | Parameter settings |
|---|---|
| Future rewards | 0.4 |
| User activity factor | 0.05 |
| Explore the coefficient $a$ | 0.1 |
| Use the coefficient $n$ | 0.05 |
| Main update cycle TR | 6 h |
| Minor update period TD | 3 h |

Table 2: Offline recommendation algorithm accuracy.

| Method | Click through rate | Normalized discount cumulative gain |
|---|---|---|
| LR | 0.119 | 0.370 |
| FM | 0.143 | 0.441 |
| W&D | 0.149 | 0.449 |
| LinUCB | 0.145 | 0.420 |
| HLinUCB | 0.122 | 0.351 |
| DDQN + $U$ + EG | 0.163 | 0.480 |
| DDQN + $U$ + DBGD | 0.171 | 0.490 |

*4.2. Experimental Design.* In this experiment, we used the grid search method to determine the parameter values, so we can quickly and effectively get the parameter values to achieve the best click through rate. See Table 1 for details.

Our basic model uses DDQN, which includes future rewards, and endows our own components in this model to better play the performance of DDQN, where $U$ represents user activity, EG represents the $\varepsilon$-greedy strategy, and DBGD is the gradient descent method that we mentioned at the beginning of Section 3.

We compare the baseline algorithm implemented in this project with the following five algorithms:

(1) Logistic regression (LR), also known as logistic regression analysis, is a generalized linear regression analysis model, which belongs to supervised learning in machine learning. Its derivation process and calculation method are similar to the regression process, but in fact, it is mainly used to solve the two-classification problem (also can solve the multiclassification problem). The model is trained by given $n$ groups of data (training set), and one or more groups of data (test set) are classified after the training. Each group of data is composed of $P$ indicators

(2) Factorization Machines (FM) are a machine learning algorithm based on matrix decomposition proposed by Steffen Rendle. At present, it is widely used in advertising the prediction model and the effect is much stronger than LR. We can think that this model combines the advantages of SVM and the decomposition model. Similar to SVM, FM has a

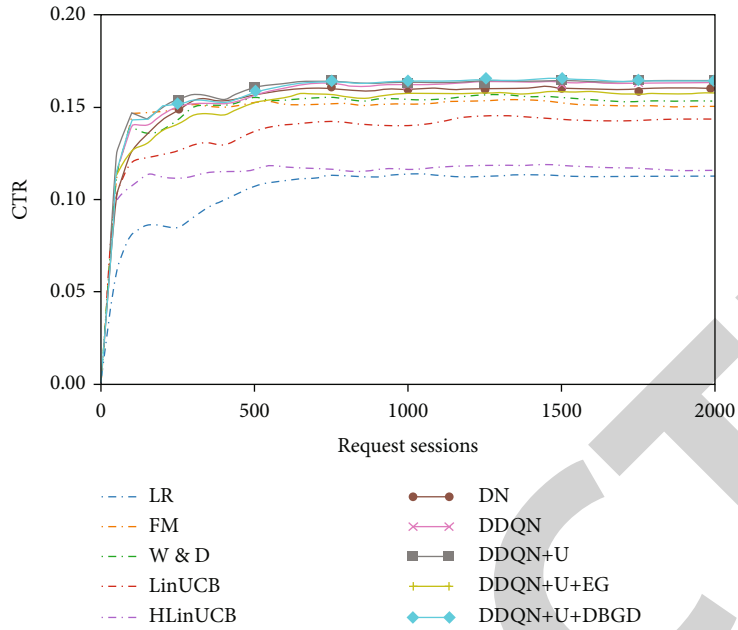


Figure 3: Offline cumulative click through rate for different methods.

wide range of predictors, which can be compatible with any real-valued eigenvectors. And FM uses decomposed parameters to model the relationship between all variables

(3) The Wide & Deep (W&D) model designs a framework that integrates the shallow wide model and deep model for joint training, which comprehensively utilizes the memory ability of the shallow model and the generalization ability of the deep model, so as to realize the accuracy and scalability of the single model to the recommendation system

(a) Wide model: it can use crossfeatures to efficiently achieve memory ability and achieve the purpose of accurate recommendation, but limited by the training data, the wide model cannot achieve generalization that has not appeared in the training data

(b) Deep model: embedded models such as FM and DNN can realize the generalization ability of the model by learning low-dimensional dense vectors, including the generalization recommendation of the unseen content. However, when the query item matrix is too sparse, the model will be overgeneralized and many unrelated synonyms are recommended, so the accuracy cannot be guaranteed

(4) LinUCB originated from ridge regression and achieved strategic balance with the help of the UCB theory. Moreover, LinUCB has a very mature and excellent theoretical foundation, which has been well applied in actual production and widely recognized in the industry

(5) HLinUCB is an excellent recommendation algorithm. Compared with LinUCB, HLinUCB can use hidden features to further model users' rewards. However, for the current problem background of educational resource recommendation, HLinUCB can only directly apply the features of educational resources and other features between users and services will be used as hidden features to participate in modeling

### 4.3. Analysis of Experimental Results

*4.3.1. Offline Evaluation.* We first apply the algorithm to the offline dataset, but the offline dataset has several prominent shortcomings. First, it is unable to record the change of users' preference for resources under the effect of different recommendation algorithms. Second, because the number of resources available to users is very limited, the exploration methods of various algorithms have not been fully utilized. Based on these two points, we only provide the comparison results of the accuracy of various algorithms.

In this experiment, samples were taken according to the design proportion to prevent underfitting and overfitting of the model. The performance results of the recommended algorithm that we designed are shown in Table 2. It can be seen that our algorithm is significantly better than the other five kinds of baseline algorithms. This is inseparable from our modeling process. The model attaches great importance to the interaction between users and resources and can have a more realistic simulation effect. At the same time, we have considered future rewards, which can provide a good basis

Table 3: Online recommendation algorithm accuracy.

| Method | Click through rate | Accuracy | Normalized discount cumulative gain |
|---|---|---|---|
| LR | 0.007 | 0.008 | 0.031 |
| FM | 0.008 | 0.009 | 0.034 |
| W&D | 0.006 | 0.008 | 0.026 |
| LinUCB | 0.008 | 0.010 | 0.037 |
| HLinUCB | 0.008 | 0.013 | 0.043 |
| DDQN + $U$ + EG | 0.010 | 0.011 | 0.032 |
| DDQN + $U$ + DBGD | 0.011 | 0.015 | 0.048 |

for the real-time performance of the recommendation process, which is also more in line with the requirements of the general production environment. However, it is worth noting that in the offline environment, the number of resources available for selection does not meet the requirements of the model, so the effect of combining user activities with resource recommendation and exploration may not be fully displayed here, and the information transmission between the algorithm of the model and users may not achieve the best effect. See Table 2 for details.

The following is mainly about our convergence process of the algorithm model. CTR is the click through rate. In Figure 3, we show the click through rate based on various combinations or individual methods in detail. It can be seen that our algorithm (DDQN + $U$ + DBGD) has more efficient and stable performance to converge to a better click through rate, in which the offline data is used to simulate the information transmission between users and recommended services, and we will update the model every interval.

Our model has also achieved good evaluation results in the online environment. At the same time, in this evaluation process, we compete with some algorithm models in the production environment. Among them, we evaluate the accuracy of the recommendation algorithm, the diversity of the recommendation content, and the suitability of the recommendation method and all the model algorithms focus on recommending quantitative user resources to avoid invalid recommendations caused by too many resources.

As described in Table 3, individual algorithms or various algorithms combined and matched have specific recommendation effects in Figure 3. Our algorithm model can be better than other individual or combined models. It is not difficult to see that our full use of future rewards plays an important role in calculating accurate recommendations for the algorithm model. At the same time, the abovementioned gradient descent method can also reduce the performance loss in the $\varepsilon$-greedy algorithm. See Figure 3 and Table 3 for details.

## 5. Conclusion

In the context of the information explosion brought by the rapid development of the Internet, it is important for enterprises to meet the personalized needs of users. In this paper, we propose a recommendation algorithm based on reinforcement learning, which improves the flexibility of the algorithm and the accuracy of the recommendation. The main work of this paper includes the following aspects:

(1) Background research of the recommendation system is conducted. Firstly, the development history of recommendation systems is sorted out and different recommendation system architectures and key technologies of recommendation systems are introduced; then, the current research status of recommendation systems is introduced; then, typical personalized recommendation algorithms are introduced, including content-based recommendation algorithms, collaborative filtering recommendation algorithms, and hybrid recommendation algorithms; finally, after summarizing the principle of reinforcement learning, finally, after summarizing the principle of reinforcement learning, we analyze the development status of recommendation algorithms based on reinforcement learning

(2) Based on the extensive practical experience of the recommendation algorithm in the fields of finance, Internet, and so on, this paper introduces deep reinforcement learning into the field of the personalized recommendation service of educational resources, simulates the user's portrait better, analyzes the interactive information transmission between users and recommended resources, positively optimizes the model, and uses future rewards to provide an important guarantee for the accuracy of model calculation. Our model has good scalability and can be fully used in other fields except education, which also provides reference and development ideas for the current personalized recommendation service research.

## Data Availability

The experimental data used to support the findings of this study are available from the corresponding author upon request.

## Conflicts of Interest

The authors declared that they have no conflicts of interest regarding this work.

## References

[1] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[2] M. Jamali and M. Ester, "Trustwalker: a random walk model for combining trust-based and item-based recommendation," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 397–406, 2009.

[3] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Itembased collaborative fifiltering recommendation algorithms," in *Proceedings of the 10th international conference on World Wide Web*, pp. 285–295, 2001.

[4] M. Defffferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral fifiltering," *Advances in Neural Information Processing Systems*, vol. 29, 2016.

[5] X. Xiong, X. Li, Y. Hu, Y. Wu, and J. Yin, "Handling information loss of graph convolutional networks in collaborative filtering," *Information Systems*, vol. 109, article 102051, 2022.

[6] F. M. Harper and J. A. Konstan, "The MovieLens datasets," *Acm Transactions on Interactive Intelligent Systems*, vol. 5, no. 4, pp. 1–19, 2016.

[7] R. M. Bell and Y. Koren, "Lessons from the Netflix prize challenge," *ACM SIGKDD Explorations Newsletter*, vol. 9, no. 2, pp. 75–79, 2007.

[8] S. Berkovsky, R. Taib, and D. Conway, "How to recommend? User trust factors in movie recommender systems," in *International Conference on Intelligent User Interfaces*, 2017.

[9] Y. Fangyong, W. Hong, and W. Jihua, "An algorithm for relieving date sparsity in reciprocal recommendation system," *Journal of Jinan University (Natural Science Edition)*, vol. 31, no. 1, pp. 48–54, 2017.

[10] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: a comprehensive review," *Computational Social Networks*, vol. 6, no. 1, pp. 1–23, 2019.

[11] R. Guo, X. Li, Y. Hu, Y. Wu, X. Xiong, and M. Qu, "A simple graph convolutional network with abundant interaction for collaborative filtering," *IEEE Access*, vol. 9, pp. 77407–77415, 2021.

[12] Q. Dai, X. M. Wu, L. Fan et al., "Personalized knowledge-aware recommendation with collaborative and attentive graph convolutional networks," *Pattern Recognition*, vol. 128, article 108628, 2022.

[13] G. Shen, Z. Zhao, and X. Kong, "GCN2CDD: a commercial district discovery framework via embedding space clustering on graph convolution networks," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 1, pp. 356–364, 2022.

[14] Z. Li, Z. Liu, J. Huang et al., "MV-GCN: multi-view graph convolutional networks for link prediction," *IEEE Access*, vol. 7, pp. 176317–176328, 2019.

[15] G. Chen, J. Wu, W. Yang, A. K. Bashir, G. Li, and M. Hammoudeh, "Leveraging graph convolutional-LSTM for energy-efficient caching in blockchain-based green IoT," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 3, pp. 1154–1164, 2021.

[16] M. Zhao, Q. Deng, K. Wang et al., "Bilateral filtering graph convolutional network for multi-relational social recommendation in the power-law networks," *ACM Transactions on Information Systems (TOIS)*, vol. 40, no. 2, pp. 1–24, 2022.

[17] S. M. Shah and V. S. Borkar, "Q-learning for Markov decision processes with a satisfiability criterion," *Systems & Control Letters*, vol. 113, pp. 45–51, 2018.

[18] T. Ma, B. Dong, and H. Qv, "Spatial first hyperspectral image classification with graph convolution network," *IEEE Access*, vol. 10, pp. 39533–39544, 2022.

[19] F. Gama, E. Isufi, G. Leus, and A. Ribeiro, "Graphs, convolutions, and neural networks: from graph filters to graph neural networks," *IEEE Signal Processing Magazine*, vol. 37, no. 6, pp. 128–138, 2020.

[20] S. Pu, Y. Wu, X. Sun, and X. Sun, "Hyperspectral image classification with localized graph convolutional filtering," *Remote Sensing*, vol. 13, no. 3, p. 526, 2021.

[21] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," 2012, http://arxiv.org/abs/1205.2618.