Digital Library Book Recommendation Model Based on Collaborative Filtering and Cloud Computing

Xiujing Wang*, Chuanyu Zhang, Jinming Wu
Library, Changchun Sci-Tech University, Changchun 130600, China
E-mail: Wangxiujing326217@163.com
*Corresponding author

Keywords: collaborative filtering, cloud computing, digital library, attention mechanism, recommendation model

Received: May 11, 2024

With the rapid development of the Internet, digital resources are growing exponentially. In this context, more libraries are transforming into smart libraries. The demand for detailed project recommendations and real-time updates is becoming increasingly prominent due to information redundancy caused by excessive data. Therefore, this study attempts to build a digital library data platform based on cloud computing. In the traditional recommendation algorithm, the attention mechanism is introduced. A neural collaborative filtering algorithm based on channel attention is proposed, and an improved digital library book recommendation model is designed by combining the two. The test results showed that the average value, optimal value, and standard deviation of the improved algorithm were the lowest among the comparison algorithms. The loss function value was the lowest. The average recommendation accuracy of the model was 92.08%, the recall was 89.88%, and the area under the curve was 0.89. When the number of recommended books was 5, 10, 15, and 20, the recommendation matching degree was 92.06%, 96.27%, 90.03%, and 93.46%, respectively. The coverage rate was above 90%, with an average time consumption of 0.28s. The recommended running time on small, medium, and large data sets was 2.3s, 10.8s, and 21.7s, respectively, and the memory consumed was 119MB, 481MB, and 961MB, respectively. The mean reciprocal ranking for different proportions of experimental population was 0.393. It demonstrates that the proposed book recommendation model for digital libraries has higher accuracy, stable recommendation effects, and less time consumption. The library recommendation model can effectively provide targeted and personalized recommendation services to users.

Povzetek: Razvit je sistem za priporočanja knjig v digitalnih knjižnicah, ki temelji na sodelovalnem filtriranju in računalništvu v oblaku. Z uvedbo pozornostnega mehanizma v tradicionalni algoritem priporočanja je razvit izboljšan model, ki dosega visoko točnost priporočil, zanesljivost in učinkovitost, kar prispeva k boljšemu upravljanju podatkov in optimizaciji v pametnih knjižnicah.

1 Introduction

In recent years, cloud computing, as an emerging commercial computing model, has played a crucial role in both domestic and international software development markets due to its large scale, virtualization, high reliability, and strong scalability. It is the main form of information construction [1]. Cloud computing is essentially an infrastructure. Traditional libraries can rely on information technology to achieve innovation, gradually developing towards digital libraries and smart libraries. Users can access the library at any time through the Internet to obtain the necessary information, without being limited by time and space [2]. However, due to the massive amount of data leading to information redundancy, users often spend great time and efforts searching for highly compliant information, which brings new challenges to book recommendation systems [3]. More scholars have conducted in-depth research on information recommendation systems. However, existing recommendation systems mainly rely on Collaborative Filtering (CF) algorithms and content-based recommendation algorithms. Although these methods can provide recommendation services to users to a certain extent, they have major problems in practical applications, such as data sparsity, cold start, low recommendation accuracy, poor timeliness, and high computational complexity [4]. Moreover, traditional CF algorithms do not consider the time factor, and often cannot timely change recommended items when user interests change, resulting in low timeliness [5]. Therefore, this study attempts to combine the attention mechanism in deep learning to propose a neural CF algorithm based on channel attention. By introducing matching function learning modules, convolutional attention network modules and generalized matrix decomposition modules, it aims to improve the accuracy and efficiency of the recommendation system. The improved CF algorithm is combined with the cloud computing platform to build an

efficient digital library book recommendation model, achieving more accurate and real-time personalized recommendation services. This paper structure has four parts. The first part summarizes relevant research results. The second part proposes a CF recommendation algorithm based on cloud computing and improvement. The third part analyzes the library user recommendation model based on the improved algorithm. The fourth part is the discussion. The last part is the research summary and proposes future research directions.

2 Related works

CF algorithm is a classic recommendation technology that has been deeply applied in practice. Due to its simplicity, high accuracy, and wide applicability of data types, it is widely used in fields such as e-commerce, news, video, and social networks. Zhang et al. built a multi-neural CF attraction recommendation architecture for scenic spot recommendation scenarios in diverse tourism environments. By modeling different tourism backgrounds and trajectory backgrounds, the overall feature representation of tourists was obtained. Then, a neural network was used to project scenic spots into the feature space. Finally, different scenic spots were recommended using this architecture [6]. Lim and Xie designed a new weighted pulse neighborhood regularization three factor classification. Then, a CF algorithm was used to identify and predict unobserved target genes. The results indicated that the model was significantly superior to other algorithms based on matrix factorization, which could be directly applied to tissue-specific data [7]. Zhang et al. built a new CF algorithm for information recommendation in the e-commerce, which incorporated temporal behavior information and decomposed adjacent sets into probability matrices. Each neighbor in the adjacent set had a certain impact on the current user or project. The results indicated that this algorithm was superior to mainstream algorithms [8]. Sun and Zheng first used big data technology and CF algorithm to construct a multi-level music audio database. Then a fast response estimation model was designed to form a new multi-level music audio data query system with discrete dynamic modeling features for complex systems. The results indicated that the search efficiency and accuracy of this audio database were superior to other databases [9].

Cloud computing is a computing mode that provides users with a number of dynamic virtualization resources through the Internet. Due to its large scale, strong scalability, virtualization and other characteristics, the research on cloud computing has been popular in recent years. Narwal and Dhingra built a credit-based resource aware load balancing scheduling algorithm to address the low resource utilization and load imbalance in most existing scheduling algorithms in cloud computing. The designed algorithm weighted tasks were mapped to resources. Compared with existing algorithms, this method improved processing time and completion time by 48.5% and 16.9%, respectively [10]. Li and Yao proposed a new strategy for digital entrepreneurship based on artificial intelligence and cloud computing, which enabled machines to self identify, report, and solve faults. Through the collection of artificial intelligence algorithms and device hardware, customers could make flexible payments on demand. Then some solutions such as warehouse management, data analysis, and financial management were provided to customers, promoting innovation in new enterprises [11]. Aktan and Bulut developed meta-heuristic and hybrid meta-heuristic algorithms to address the task diversity and resource heterogeneity in cloud computing, and combined them with greedy algorithms. The Eurasian results indicated that this algorithm was superior to a single algorithm, improving the average completion time of task groups and the average standard deviation of virtual machine load, effectively allocating resources, and optimizing task objectives [12]. Rani and Garg proposed an energy-saving adaptive particle swarm optimization algorithm for independent task scheduling decisions to improve cloud computing performance. The acceleration coefficient and inertia weight were changed, and mutation operation was introduced to avoid the algorithm getting stuck in local minima. The algorithm had certain advantages in terms of time span and energy consumption [13]. Finally, the study summarizes the research areas, indicator test results and limitations of the above literature review. The results are shown in Table 1.

Table 1: Literature summary

Author(s)	Algorithm	Domain	Key metrics	Limitations
Zhang et al. [6]	Neural CF	Tourism	Accuracy:85% MRR:70%	Data sparsity
Lim and Xie [7]	Tri-factor CF	Bio-informatics	AUC:0.766 Enrichment Factor:4.19	Cold start problem
Zhang et al. [8]	Timing behavior CF	E-commerce	Book RMSE:0.7208 Movie RMSE:0.7302	Data sparsity problem
Sun and Zheng [9]	CF + Big Data	Music	Reliability:82% Time consuming:4s	Scalability issues
Narwal and Dhingra [10]	CB-RALB-SA	Resource utilization	Energy consumption:20.07kWh SLA violation rate:10.12%	High system consumption

Aktan and Bulut [12]	GA+DE+SA+GR	Task scheduling	Average scheduling time:8.0506ms	Weak generality
Rani and Garg [13]	SPV+PSO	Task assignment	Energy consumption:0.01491kWh Total task execution time:1878.6s	Task type limitations

From Table 1, although personalized information recommendation service systems have developed relatively maturely, there are still problems with data sparsity and cold start. Therefore, the study proposes a CF algorithm based on cloud computing and neural networks, which analyzes heterogeneous multi-dimensional data to ensure the information comprehensiveness and targeted recommendations, thereby promoting the development of book recommendation services.

3 Digital library recommendation model design based on CF and cloud computing

The study first combines the attention mechanism to propose a CF algorithm on the basis of channel attention. Then, the improved CF algorithm is combined with cloud computing to build a book recommendation service model for digital libraries.

3.1 Design of recommendation model based on CF

CF is a classic and commonly used recommendation method, which generally determines the user preferences through historical data analysis to find a collection of other users with similar preferences, or analyze project features to seek the most similar project collection [14]. The CF recommendation process is shown in Figure 1.

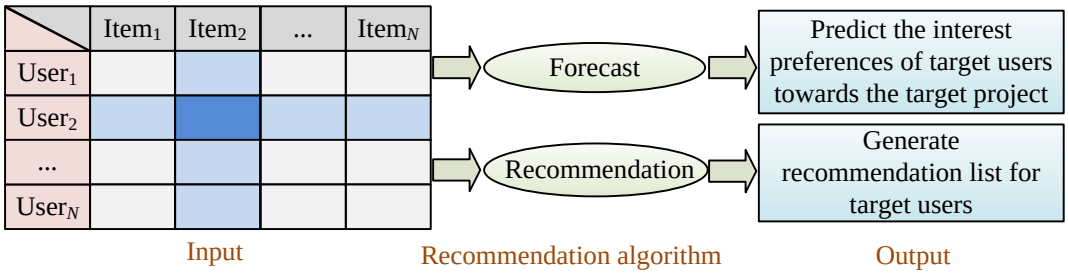


Figure 1: CF recommendation process

Figure 1 shows the entire process of the CF algorithm from inputting user and project features to finally generating recommendation results, including steps such as feature extraction, attention mechanism, matching function, and rating prediction. Each element in the matrix represents the user's rating for the project. Assuming that $User_1$ and $User_1$ liked similar projects in the past, the projects that user $User_1$ likes in the future may also be liked by user $User_2$. The CF algorithm is used to make predictions, thereby generating projects that may be of interest to the target user, and the recommendation list is sorted and recommended to $User_2$. The most commonly used method in CF is matrix factorization, which performs inner product on the feature vectors of users and items. However, with the sharp increase in user visits and reading volume, traditional CF algorithms can no longer meet existing recommendation requirements [15]. Therefore, based on the attention mechanism, the Channel Attention of Neural CF algorithm (CA-NCF) is proposed, which consists of a matching function learning module, a convolutional attention network module, and a generalized matrix factorization module. The embedding matrix calculation of user and project features is shown in equation (1).

$$\begin{cases} p_u = P^T v_u^U \\ q_i = Q^T v_i^I \end{cases} \tag{1}$$

In equation (1), u represents the user. i represents the project. v_u^U and v_i^I represent the feature vectors of users and projects. Its embedding vectors are represented as p_u and q_i. $P \in R^{M \times K}$ and $Q \in R^{N \times K}$ represent the embedding matrices of user features and project features. K, M and N respectively represent the size of the embedding matrix and the feature vectors of users and items. The impact of different feature attributes on user reading behavior is calculated using equation (2).

$$f(p_u, q_i) = \mathrm{ReLU}(W(p_u \square q_i) + b_0) \tag{2}$$

In equation (2), f represents the attention function. $W \in R^{K \times K}$ signifies the weight matrix from the input layer to the hidden layer. p_u represents the user embedding vector, q_i represents the item embedding vector. b_0 signifies the bias vector. The activation function is ReLU. The attention weight obtained from the SoftMax function is then converted into probability distribution. The attention coefficient is calculated, as

shown in equation (3).

$$a(u,i)=\frac{\exp(f(p_u,q_i))}{\sum_{i\in R_u}\exp(f(p_u,q_i))} \tag{3}$$

In equation (3), R_u represents the set of interaction history between user u and item i. $a(u,i)\in R^{K\times 1}$ signifies the attention weights of user u and item i. The matching function learning calculation based on multi-layer perceptual networks is shown in equation (4).

$$\begin{cases} x_0=\left[\dfrac{a(u,i)}{(p_u,q_i)}\right] \\ x_1=\mathrm{ReLU}(W_1^T x_0+b_1) \\ x_2=\mathrm{ReLU}(W_2^T x_1+b_2) \\ \cdots \\ x_n=\mathrm{ReLU}(W_n^T x_{n-1}+b_n) \end{cases} \tag{4}$$

In equation (4), $x_0\in R^{2K\times 1}$ represents the input vector of the multi-layer perception network. $\{x_1,x_2,...,x_n\}$ represents the corresponding input and output vectors in the hidden layer. $\{W_1,W_2,...,W_N\}$ and $\{b_1,b_2,...,b_n\}$ signify the weight matrix and bias vector for each layer. x_n represents the final prediction vector. ReLU represents the activation function used. In the convolutional attention network module, the outer product of user and item embeddings is shown in equation (5).

$$FM=p_u\otimes q_i=p_u q_i^T \tag{5}$$

In equation (5), $FM=R^{K\times K}$. fm represents the hidden state in convolutional blocks. Each element can be represented as $fm_{k_1,k_2}=p_{u,k_1}q_{i,k_2}$. A two-dimensional matrix is obtained through outer product, and then pull the matrix into a vector with dimension K^2 through Flatten. The study adds a Convolutional Neural Network with ECA-Net (E-CNN) module to a convolutional neural network. Firstly, the user item interaction representation is input into the convolutional neural network to train the final layer. The Efficient Channel Attention Network (ECA-Net) is added to the convolutional block, and convolutional kernels are used for convolution. The interaction between user and item information is calculated in the channel, as shown in equation (6).

$$\begin{cases} y=\dfrac{1}{WH}\sum_{u=1,i=1}^{W,H}\chi_{ui} \\ w=Sigmoid(C1D(y)) \end{cases} \tag{6}$$

In equation (6), $W\in R^{K\times K}$ represents the weight matrix from the input layer to the hidden layer. w represents the user-item information interaction between channels. $C1D$ represents the one-dimensional convolution. y represents global average channel pooling. In a convolutional neural network, the input of the last convolutional layer is defined as a three-dimensional tensor, which is combined with the ECA-Net attention network. The overall hidden state of the user and item embedded in the outer product FM is shown in equation (7).

$$\begin{cases} h=(fm_{2u+a},fm_{2i+b}) \\ H^{l-1}=h^{l-1}\cdot t^l_{1-a,1-b,c} \\ fm_{u.i,c}=\sigma(fm^{l-1}_{u,i,c}+fm^l_{u,i,c}) \end{cases} \tag{7}$$

In equation (7), h represents the hidden layer activation value. fm represents the overall hidden state in FM. H^{l-1} represents the activation value of $l-1$-th layer. o_l signifies the bias vector of the l-th layer. a and b signify the interaction matrix parameters of the local regions captured in each layer. $t^l_{1-a,1-b,c}$ represents a convolutional filter. σ represents the ReLU activation function. The final predicted score is shown in equation (8) [16].

$$\hat{y}_{ui}^{E-CNN}=G(FM)\cdot W_G^T \tag{8}$$

In equation (8), FM represents the outer product. $G(FM)$ represents the output vector of the model in multiple intermediate layers. W_G^T represents the re-weighting of user and item weights in the output vector. $\hat{y}_{ui}^{E-CNN}$ represents the prediction vector based on the matching function. In model training, all dimensional interactions can be captured. In the final layer, global connectivity interaction information from the original feature map can be extracted, thereby mining semantic information. Finally, a generalized matrix factorization module is added to CA-NCF to effectively alleviate the data sparsity in deep neural networks. The predicted vector definition is shown in equation (9) [17].

$$z(p_u,q_i)=p_u\ \square\ q_i \tag{9}$$

In equation (9), p_u represents the user embedding vector. q_i represents the item embedding vector. $\square$ represents the element wise product of the user embedding vector p_u and the item embedding vector q_i. $z(p_u,q_i)$ represents the prediction vector. The calculation of mapping the prediction vector to the prediction layer is shown in equation (10) [18].

$$\hat{y}_{ui}^{MF} = \sigma_o(W_o^T z(p_u, q_i)) \tag{10}$$

In equation (10), σ_o signifies the activation function. W_O represents the connection weight matrix. $z(p_u, q_i)$ represents the prediction vector. $\hat{y}_{ui}^{MF} \in R^{K\times 1}$ represents the prediction layer. After calculating all three modules, the fusion model of the three is shown in equation (11) [19].

$$\begin{cases} C_1 = W_o^T(x_n, z(p_u, q_i)) \\ C_2 = concat(C_1, W_G^T(FM)) \\ \hat{y}_{ui} = Sigmoid(C_2) \end{cases} \tag{11}$$

In equation (11), $\hat{y}_{ui}$ represents the interaction prediction score of user u for item i. $z(p_u, q_i)$ represents the prediction vector. In the connection layer, the Sigmoid function is used as the activation function for the fully connected layer. By combining the three modules in equation (11), the user prediction score of the model is obtained through learning. In model learning, the loss function calculation is shown in equation (12) [20].

$$L = -\sum_{(u,i)\in\gamma^+\cup\gamma^-} y_{ui}\log\hat{y}_{ui} - (1-y_{ui})\log(1-\hat{y}_{ui}) \tag{12}$$

In equation (12), γ^+ and γ^- represent positive samples and negative samples in user interaction. y_{ui} represents the true value. 1 indicates that user u has interacted with item i. If there is no interaction, it is represented as 0. The predicted value $\hat{y}_{ui}$ represents the possibility of the user u being related to item i. $\hat{y}_{ui} \in [0,1]$. L represents the loss function. Based on the above calculations, the architecture of the CA-NCF module is shown in Figure 2.

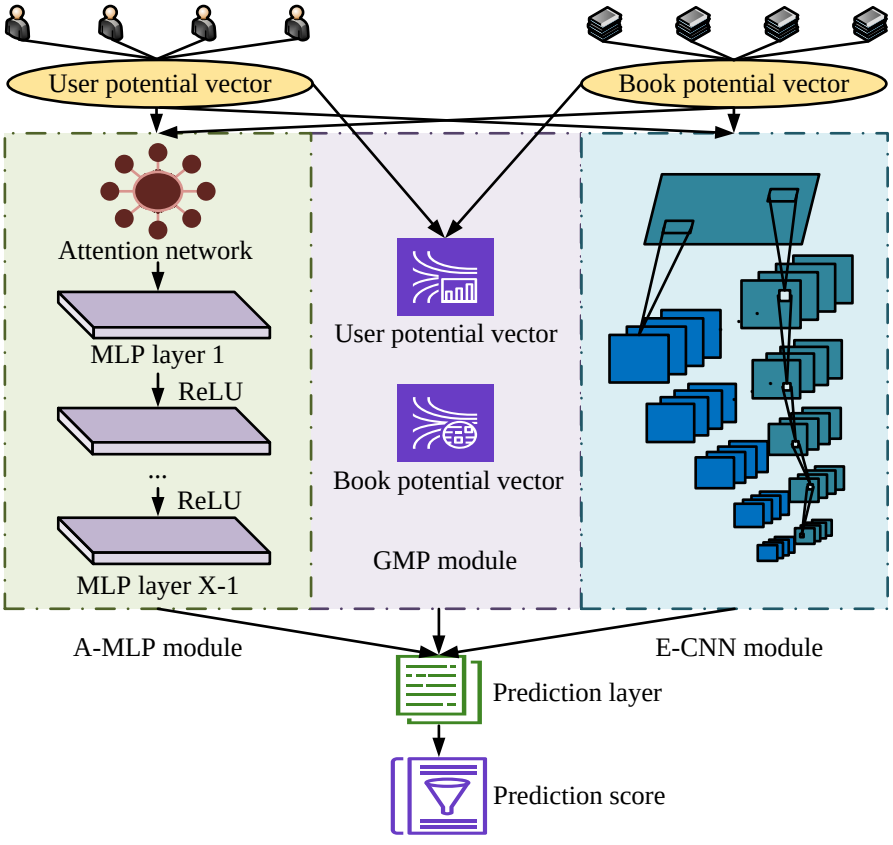


Figure 2: Architecture of CA-NCF module

In Figure 2, the CA-NCF model consists of E-CNN, a matching function learning module with an attention mechanism, and a generalized matrix decomposition module. The combination of the three can effectively improve the learning and generalization capabilities of the model. Firstly, the CA-NCF algorithm uses the outer product method to obtain the latent vector between users and items. The interaction matrix is used as the interaction mapping, which contains matrix decomposition and other pairwise correlation interaction information. CA-NCF can incorporate the correlation between user and item embeddings into the model to

accurately capture the pairwise relationship between embedding dimensions and dimensions. Secondly, the E-CNN module obtains the latent vector between items, that is, the interaction information between users and items in cross-channels, which can also obtain richer semantic information. Finally, the generalized matrix decomposition module combines the user latent vector with the item latent vector, which can effectively alleviate the data sparsity problem generated by the interaction between users and items. According to the above steps, after the three parties jointly complete the calculation of user project matching degree, the generated results are input into the prediction layer to obtain the predicted book recommendation score. Therefore, the pseudo code of the CA-CF algorithm is shown in Figure 3.

Pseudocode: CA-NCF Algorithm

```
# Input: User-item interaction matrix R, latent dimension size latent_dim, number of training epochs epochs, learning rate learning_rate

# Initialize parameters
Initialize user latent factor matrix P
Initialize item latent factor matrix Q
Initialize channel attention weight matrix W_ca
Initialize neural collaborative filtering weight matrix W_ncf

# Channel attention mechanism
function channel_attention(H):
    # Global average pooling
    H_avg = GlobalAveragePooling(H)

    # Attention weights
    attention_weights = ActivationFunction(H_avg * W_ca)

    # Weighted representation
    H_weighted = H * attention_weights

    return H_weighted

# Neural collaborative filtering part
function neural_collaborative_filtering(user_latent, item_latent):
    # Concatenate user and item latent representations
    concatenated_input = Concatenate(user_latent, item_latent)

    # Fully connected neural network to predict score
    predicted_score = ActivationFunction(concatenated_input * W_ncf)

    return predicted_score

# Loss function
function loss_function(predicted_score, true_score):
    prediction_loss = MeanSquaredError(predicted_score, true_score)

    return prediction_loss

# Training process
function train(R, epochs, learning_rate):
    for epoch in range(epochs):
        total_loss = 0

        # Iterate over all user-item pairs
        for each user i, item j in R:
            # Get user and item latent representations
            user_latent = P[i]
            item_latent = Q[j]

            # Channel attention mechanism
            user_latent_weighted = channel_attention(user_latent)
            item_latent_weighted = channel_attention(item_latent)

            # Calculate predicted score
            predicted_score = neural_collaborative_filtering(user_latent_weighted, item_latent_weighted)

            # Get true score
            true_score = GetTrueScore(i, j)

            # Calculate loss
            loss = loss_function(predicted_score, true_score)
            total_loss += loss

            # Backpropagation to update parameters
            UpdateParameters(P, Q, W_ca, W_ncf, learning_rate, loss)

        Output("Epoch {}, Loss: {}".format(epoch, total_loss))

# Main function
function main():
    # Initialize user-item interaction matrix
    R = GetUserItemInteractionData()

    # Set parameters
    latent_dim = 50
    epochs = 100
    learning_rate = 0.01

    # Train the model
    train(R, epochs, learning_rate)

# Execute main function
main()
```

Figure 3: Pseudo code of CA-NCF

Figure 3 is the pseudo code of CA-NCF. Firstly, the parameters are initialized, including the latent factor matrix of users and items, channel attention weights, and neural CF weights. Secondly, the algorithm uses the

channel attention mechanism to enhance the representation between users and items, and uses the attention weights to capture key features. Then, the neural CF algorithm is weighted to represent the predicted score. During the training process, the mean square error is used to calculate the loss after traversing the users and items. The back propagation is used to update the parameters. Finally, CA-NCF performs the model training process to optimize the model, so as to improve the recommendation performance.

3.2 Construction of the digital library recommendation system based on cloud computing and CA-NCF

After improving the CF algorithm mentioned above, the research further builds a CA-NCF digital library book recommendation model based on cloud computing platform. Cloud computing has three types: public cloud, private cloud, and comprehensive cloud. Its core idea is to unify the processing and calculation of massive resources in the network, forming an intelligent resource database. Through this intelligent database, users can obtain the services and projects they need using a certain method. The current mainstream Software Platform Infrastructure (SPI) model architecture consists of Infrastructure as a Service (IaaS) layer, Platform as a Service (PaaS) layer, and Software as a Service (SaaS) layer, as shown in Figure 4.

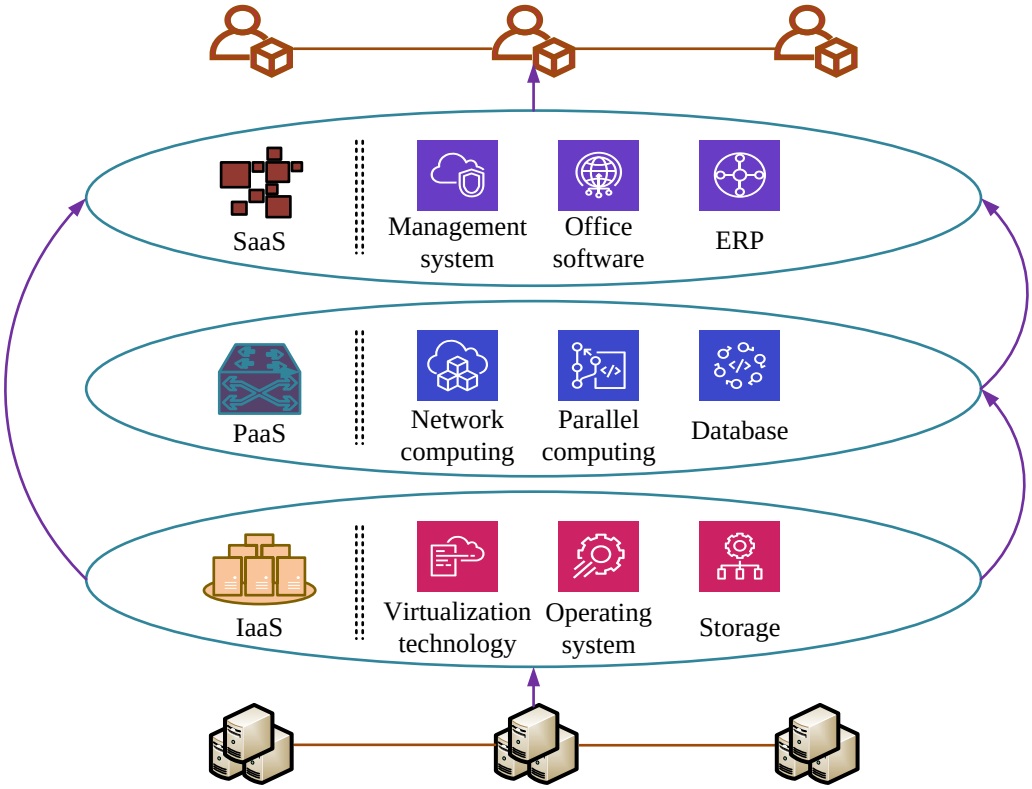


Figure 4: SPI model architecture

In Figure 4, the IaaS is the physical hardware in the cloud computing architecture. The PaaS layer integrates various existing resources. Based on the hardware resources provided by IaaS, real-time monitoring of resources can be carried out, and resources can be opened to SaaS users. At the same time, the PaaS layer also has a promoting effect on the development of the SaaS layer. In response to the large data volume and multi-dimensional data structure in digital libraries, software, platforms, and infrastructure services need to be continuously optimized with flexibility and targeting. The most important of cloud computing technology is data management technology. Currently, the most classic data management technology is Google's Hadoop. The Hadoop is selected as a data management technology for digital library recommendation models. The two core components of Hadoop are Hadoop Distributed File System (HDFS) and MapReduce. The combination of the two can effectively address the storage and distribution problems of digital library data. The specific architecture of HDFS is shown in Figure 5.

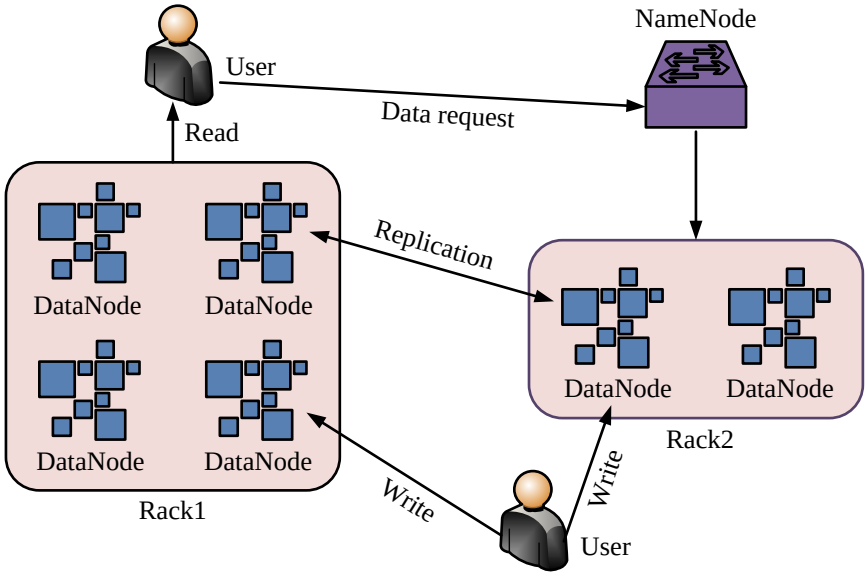


Figure 5: The architecture of HDFS

In Figure 5, HDFS adopts a master-slave structure, with the core server being NameNode, which stores file data and manages file-client access. Its storage is managed by DataNode nodes. NameNode is responsible for building name-spaces and manipulating book related content. Users can use the NameNode node to manipulate file related content, such as opening books, collecting books, sharing books, and rating books. If a user wants to query a certain book, they first need to obtain a drop-down table of the data information block distribution. Secondly, a channel together with the corresponding DataNode is created to read the file content. Under this operation, the file content data will not flow to the Namenod node. MapReduce is a distributed programming and task scheduling model used for parallel operations on large-scale data sets. The data processing flow diagram is shown in Figure 6.

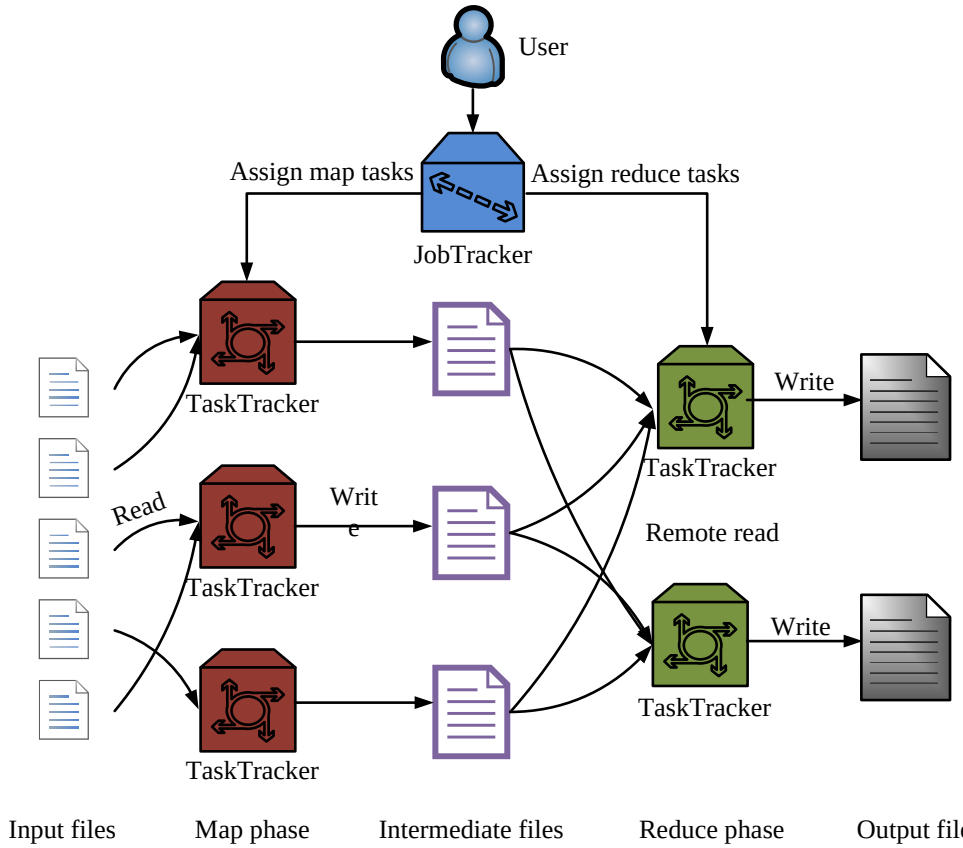


Figure 6: MapReduce data processing flow

In Figure 6, the study uses the C/S framework in the MapReduce module. Firstly, the user submits the task to the JobTracker master node, which divides the file into several data blocks. Then, the map task is assigned to the corresponding TaskTracker node. Subsequently, TaskTracker maps the data blocks of the file and stores the results locally. The R processed results correspond to the number of Reduce. Secondly, the TaskTracker running the map task transfers the information stored in the file to the JobTracker, which then specifies the TaskTracker running the Reduce task. Based on the information transmitted by JobTracker, TaskTracker will uniformly process records with the same keywords, sort them according to keywords, and then perform Reduce task. Finally, the TaskTracker running the Reduce task writes the processing results to the distributed file system. Based on the above design, the recommendation process for building a digital library based on cloud computing platform and improved CA-NCF is shown in Figure 7.

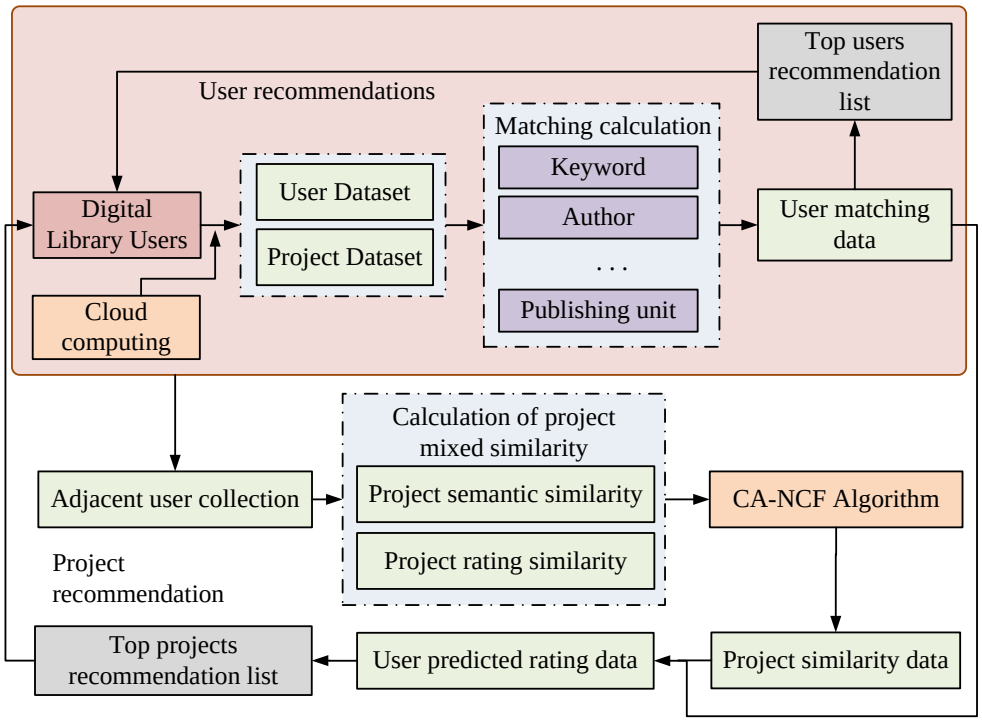


Figure 7: Process of library recommendation model based on CA-NCF

In Figure 7, in the book recommendation process of the digital library, firstly, the cloud computing platform collects and analyzes the behavioral data generated by users through various operations in the digital library, forming user data sets and project data sets, respectively. Subsequently, the user is formed into a set of adjacent users of the target user, and the item data set in the list is obtained. Then, the attribute information of the book is obtained through the cloud computing platform, and effective information is extracted from the main attributes of the book selected by the user, such as keywords, authors, journals, publishing units, and other specific information. These attributes are given corresponding weights according to the user's interest preference. Based on the matching degree value, several users with the highest matching degree score are recommended to original users, completing the user recommendation service. Finally, according to the CA-NCF algorithm, the similarity of each item is matched. All books and user description logs are matched for similarity. The similarity value is calculated and sorted to obtain the user's predicted score for the item. The highest predicted score of several book items is recommended to the user, thus completing the project recommendation service.

4 Experimental analysis of intelligent recommendation algorithm based on cloud computing and improved CF algorithm in digital libraries

To verify the feasibility of the recommendation method based on cloud computing and improved CA-NCF algorithm in digital libraries, this study first tests the performance of the CA-NCF algorithm. Then, a comparative test is conducted on the recommendation service model based on the CA-NCF algorithm. In the comparative test, all algorithms are selected with the same parameters and data sets to ensure the experiment accuracy.

4.1 Comparative testing of CA-NCF algorithm

The Epinions data set is selected for the experiment, which includes 324824 ratings from 5269 readers on 14156 books.

The data set contains a training set and a testing set in a 7:3 ratio. The study selects three commonly used recommendation algorithms, namely Item-based Collaborative Filtering (ICF), User-based Collaborative Filtering (UCF), Recommendation based on Deep Learning (DLCF). They are compared with the proposed CA-NCF algorithm on various evaluation indicators. The optimal solution and average optimal solution of multiple runs are important indicators for measuring algorithm performance. The study selects the mean, standard deviation, and optimal value of 100 run results as evaluation indicators, as displayed in Table 2.

Table 2: Quantitative analysis results of algorithms

Data set	Index	ICF	UCF	DLCF	CA-NCF
Training set	Mean	4.55e-24	3.85e-24	9.50e-04	3.30e-71
	Standard deviation	5.20e-37	9.38e-24	5.01e-02	4.02e-71
	Optimal value	3.88e+01	3.16e+02	4.01e+01	2.58e+01
Testing set	Mean	2.10e+02	1.75e+02	6.30e+01	1.59e+02
	Standard deviation	4.46e+01	6.55e+01	4.59e+01	3.66e+01
	Optimal value	7.28e+04	1.01e+01	1.19e-16	0.16e+01

In Table 2, the mean and optimal values demonstrate the accuracy of the algorithm, while the standard deviation reflects its stability. As shown in Table 1, in the two sets, the mean, optimal, and standard deviation of the CA-NCF were the lowest, indicating that the accuracy and stability of CA-NCF were optimal. Therefore, the improved CA-NCF proposed in the study had adaptability. While considering better convergence ability, the calculation results were more stable. It also indicates that the algorithm has better robustness. The statistical results of the loss function changes during the training are shown in Figure 8.

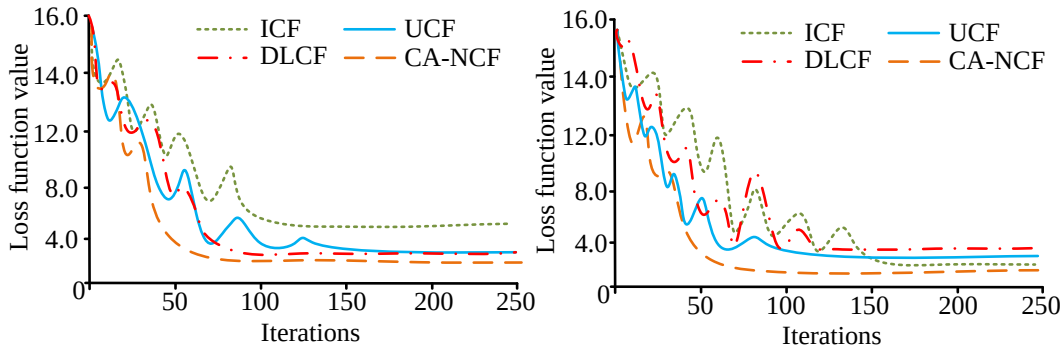


Figure 8: Loss values of different algorithms

Figures 8 (a) and 8 (b) respectively show the loss function values of the four algorithms in the training and testing sets. In Figure 8 (a), when ICF, UCF, DLCF, and CA-NCF algorithms were iterated 82, 134, 74, and 66 times respectively, the four algorithms reached a stable state in the training data set. At this time, the loss function values of the four algorithms were 6.02, 3.92, 3.91, and 2.98, respectively. In Figure 8 (b), when ICF, UCF, DLCF, and CA-NCF algorithms were iterated 144, 86, 198, and 76 times respectively, the four algorithms reached a stable state in the testing data set. At this time, the loss function values of the four algorithms were 3.03, 3.87, 3.97, and 1.98, respectively. In both data sets, the CA-NCF algorithm has the fastest convergence speed, small initial oscillations, and the lowest stability loss function value. The Receiver Operating Characteristic curve (ROC) of each algorithm is shown in Figure 9.

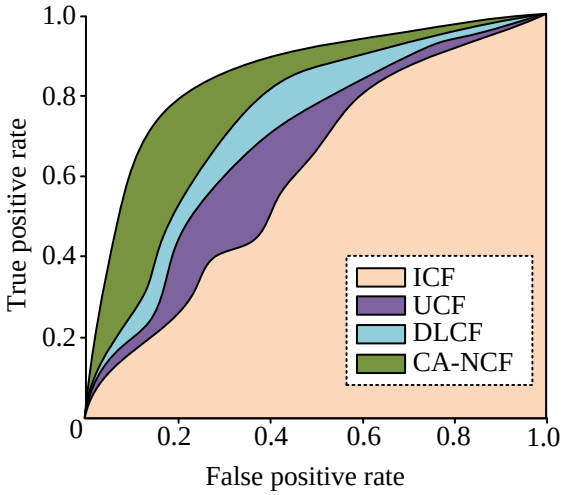


Figure 9: Statistical results of AUC indicators

Figure 9 shows the ROC of each recommendation model. The horizontal axis signifies the false positive rate. The value of the Area Under the Curve (AUC) between the ROC curve and the horizontal and vertical coordinates can measure the classification ability of the model under different thresholds. The larger the AUC, the stronger the model to distinguish between positive and negative samples, which proves that the model performance is better. In the recommendation system, the larger the AUC, the higher the proportion of positive samples in the recommendation results, that is, the items that users are interested in, thereby improving user satisfaction. As shown in Figure 9, the ROC of CA-NCF had the highest AUC value among the other three algorithms. The UCF and DLCF curves were close, and the difference in AUC values between the two was not significant. The ICF algorithm had the smallest area enclosed by the curve and coordinate axis, resulting in the worst model performance. The AUC values of ICF, UCF, DLCF, and CA-NCF recommendation algorithms were 0.51, 0.58, 0.69, and 0.89, respectively. From this, it can be concluded that the CA-NCF provides more accurate and practical recommendations, which can better meet user preferences. Finally, to verify the robustness of the model, the study introduces a cross-validation experiment to test the model performance through a multi-fold cross-validation method. The data set is randomly divided into 10 non-overlapping subsets, 9 subsets are used as the training set each time, and the remaining 1 subset is used as the validation set. The experiment is repeated 10 times to avoid experimental bias caused by data segmentation. The results are shown in Table 3.

Table 3: Ten-fold cross validation results

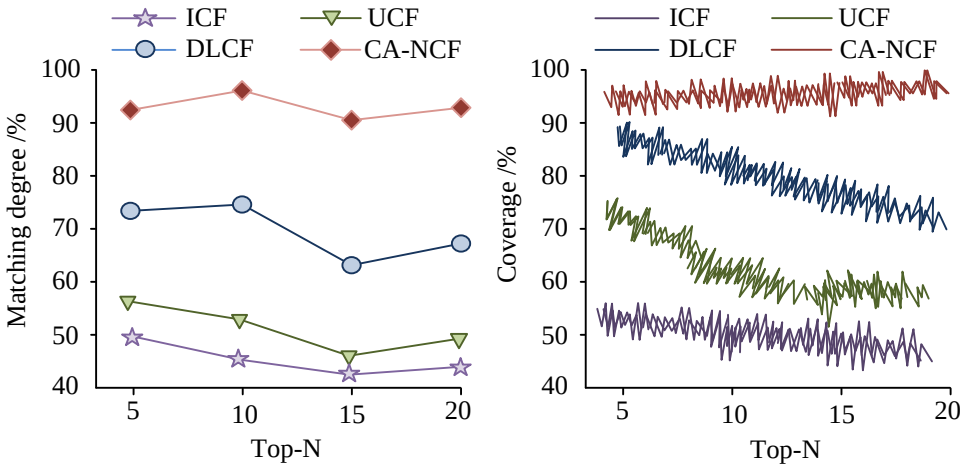
Fold times	Accuracy/%	Recall/%	AUC	MAE
1	91.8	89.5	0.88	0.14
2	91.2	90.4	0.89	0.16
3	91.9	89.8	0.88	0.14
4	92.3	89.9	0.89	0.12
5	91.9	89.6	0.88	0.18
6	92.7	90.1	0.90	0.17
7	92.3	89.7	0.88	0.14
8	91.5	90.2	0.89	0.16
9	92.9	89.7	0.89	0.13
10	92.3	89.9	0.89	0.10

Table 3 shows the accuracy, recall, AUC, and MAE values for each fold of CA-NCF after ten-fold cross validation. After ten-fold cross validation, the test results of the CA-NCF algorithm on different folds were relatively consistent, and the average values of its accuracy, recall, AUC, and MAE were 92.08%, 89.88%, 0.887, and 0.144, respectively. It can be seen that the proposed CA-NCF model has good stability and reliability on different data sets.

4.2 Performance analysis of book recommendation service model based on CA-NCF

After verifying the high accuracy and coverage of the CA-NCF algorithm, each algorithm is further applied to the digital library book recommendation model to

compare the actual application effects of each model on recommendation services. 50 college students aged 18-25 are randomly selected in Sichuan Province. Their book preferences and borrowing habits in the library in the past year are obtained. The final recommended number of books is set to 5, 10, 15, and 20 to test the matching and coverage of book recommendations. When the final number of recommended books varies, the recommendation matching and coverage of each algorithm are shown in Figure 10.



(a) Matching degree of four models (b) Coverage of four models

Figure 10: Matching degree and coverage of four models

Figures 10 (a) and 10 (b) show the recommendation matching and coverage of the four models when the final number of book recommendations is 5, 10, 15, and 20, respectively. In Figure 10 (a), CA-NCF had the highest matching degree among different recommended book numbers. When the final number of recommended books was 5, 10, 15, and 20, the recommended matching of CA-NCF was 92.06%, 96.27%, 90.03%, and 93.46%, respectively. The matching degree of ICF, UCF, and DLCF models was below 80%, and the recommendation effect was poor. From Figure 10 (b), the CA-NCF model had the highest coverage rate of recommendation results. Its coverage rate remained stable at over 90% for different final recommendation numbers. The coverage of ICF, UCF, and DLCF models was all below 90%, with ICF model coverage below 60%. Secondly, when the final number of recommended books is 5, 10, 15, and 20, the recommendation time of each model is shown in Figure 11.

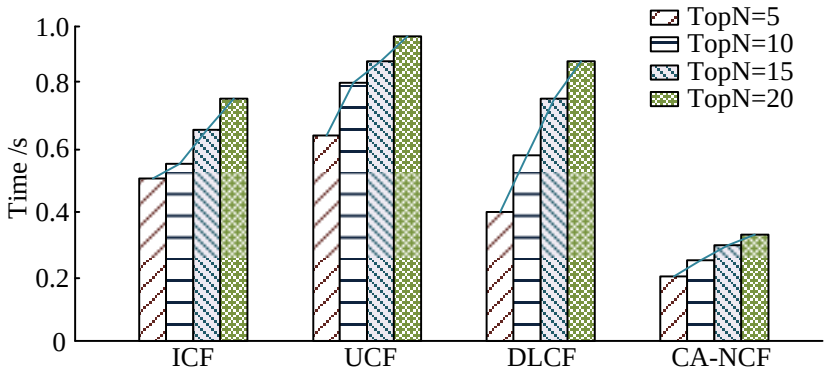


Figure 11: Time consumption of each model for different TopN

In Figure 11, when the final number of recommended books was 5, the time consumption of ICF, UCF, DLCF, and CA-NCF models was 0.51s, 0.63s, 0.40s, and 0.21s, respectively. When the final number of recommended books was 20, the time consumption of the four models was 0.77s, 0.96s, 0.88s, and 0.33s, respectively. The CA-NCF model had the least time consumption under different recommendation numbers. The increase in time consumption was not significant under different recommendation numbers, with a relatively flat trend. The DLCF model had the largest increase in time consumption under different recommendation numbers, with significant fluctuations in the line trend. To further test the effectiveness of the

CA-NCF model in practical applications, this study selects a small-scale data set containing 10,000 user records and 1,000 books, a medium-scale data set containing 50,000 user records and 5,000 books, and a large-scale data set containing 100,000 user records and 10,000 books for further extended experiments. The experimental results are shown in Table 4.

Table 4: Time and memory usage tests on data sets of different sizes

Algorithm	Small-scale data sets		Medium-sized data sets		Large-scale data sets	
	Running time/s	Memory size/MB	Running time/s	Memory size/MB	Running time/s	Memory size/MB
ICF	2.9	147	12.2	529	24.8	1063
UCF	3.1	154	12.5	573	25.3	1184
DLCF	2.8	139	11.2	501	23.6	1089
CA-NCF	2.3	119	10.8	481	21.7	961

Table 4 shows the running time and memory consumption tests of the four models, including ICF, UCF, DLCF, and CA-NCF on small, medium, and large digital library data sets. From Table 4, the recommended running time of the CA-NCF model on small, medium, and large data sets was 2.3s, 10.8s, and 21.7s, respectively, and the memory consumption was 119MB, 481MB, and 961MB, respectively. The running time of the CA-NCF model is the best among the comparison models, and the memory consumption is the smallest, which proves the scalability and computational efficiency of the proposed CA-NCF model, and has good robustness. Then, the study divides 50 college students into 15, 30, and 50 individuals in a ratio of 30%, 60%, and 100% to test its performance in ranking, as shown in Figure 12.

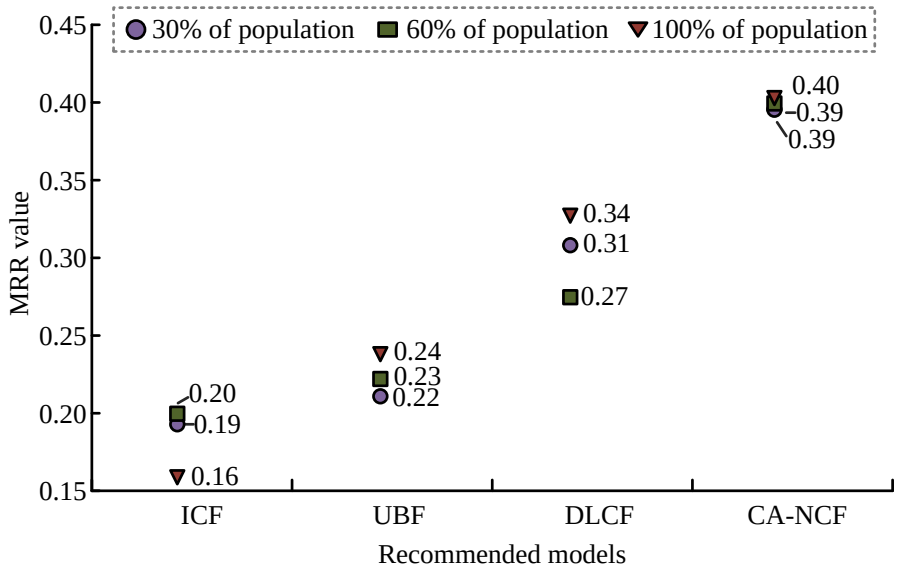


Figure 12: Statistical results of indicators of each model

Figure 12 shows the performance of each model in Mean Reciprocal Rank (MRR). MRR refers to the average of the reciprocal for the first relevant result, which is used to measure the correlation of model recommendation results. The larger the MRR value, the better the model ranking performance. In practical applications, a higher MRR value indicates that users can find more interesting items in the recommendation results generated by the system, which can effectively improve user experience. In Figure 12, the CA-NCF model exhibited the best stability among three different test populations, with all values being the highest. The DLCF model had the worst stability, but the MRR value was better than the CBF and UBF models. The MRR value of ICF was the smallest, and the score performance was the worst among all models. In addition, the study expands the number of test subjects to 100, dividing them into 5 groups. The book recommendation results of different models are evaluated. The results are shown in Table 5.

Table 5: User evaluation form

Model	Group 1	Group 2	Group 3	Group 4	Group 5	Average
ICF	79.2	80.1	76.3	76.2	84.6	79.28
UCF	80.7	88.9	79.7	78.1	75.2	80.52
DLCF	75.9	83.2	82.5	85.4	82.7	81.94
CA-NCF	94.8	94.7	89.9	82.9	88.5	90.16

As shown in Table 5, the scores of the five evaluation groups on the CA-CF model were 94.8, 94.7, 89.9, 82.9, and 88.5, respectively, with an average satisfaction score of 90.16. The average scores of the ICF, UCF, and DLCF models were 79.28, 80.52, and 81.94, respectively, which were lower than the CA-CF model. Combined with the above performance experimental comparison test, the proposed CA-NCF has good overall performance and can provide users with more comprehensive recommendation services based on the potential preferences of users, which is widely praised by users.

5 Discussion

In order to improve the recommendation accuracy of the traditional CF recommendation system, this paper designs a CF algorithm CA-NCF combined with an attention mechanism. A highly efficient digital library book recommendation model was constructed by combining CA-NCF with cloud computing, and its performance was tested. Compared with the new NCF algorithm based on neural CF proposed by Zhang J et al. in the literature [8], NCF mainly captures the nonlinear relationship between users and items through deep learning. Although it has improved the data sparsity, it still has limitations in dealing with feature importance. This study combines channel attention mechanism with CF algorithm, which not only improves the quality of feature representation but also enhances the ability to capture complex user item interactions. The results showed that the CA-NCF algorithm was superior to the NCF method in terms of AUC and time consumption. The AUC value of CA-NCF was 0.89, which was higher than 0.87 of NCF, proving that CA-CF had more advantages in dealing with data sparsity. At the same time, the training time of CA-CF on large-scale data sets was reduced by about 20% compared with NCF. The core of this is that CA-NCF innovatively integrates the attention mechanism in deep learning. At the same time, when the number of books recommended by the CA-NCF algorithm was 5, 10, 15, and 20, the recommendation matching degree was 92.06%, 96.27%, 90.03%, and 93.46% respectively. This is because the matching function learning module, convolutional attention network module, and generalized matrix decomposition module are introduced into the traditional CF algorithm. The neural network learns deeper feature relationships, which effectively improves the accuracy and efficiency of the recommendation results. Compared with the CF algorithm combined with big data analysis proposed by Sun G et al. in the literature [9], this method improves the final recommendation effect by integrating big data technology, but may has low computational efficiency in terms of data scalability and complexity. The CA-NCF algorithm improves the performance and scalability of the model. The experimental results showed that in the test of the time consumption of processing data, the computational time of CA-NCF was reduced by 15% compared with the literature method, which improved the real-time and response speed of the recommendation system. This is because the research combines CA-NCF with a cloud computing platform with excellent operating efficiency. The cloud computing platform can not only store a large amount of user data, but also realize efficient data mining and transmission. In practical applications, when the final number of recommended books was 5, 10, 15, and 20, the running time of CA-NCF was 0.21s, 0.25s, 0.30s, and 0.33s respectively.

In summary, the study proposes an efficient and accurate digital library recommendation system based on CA-NCF by combining the channel attention mechanism with the CF algorithm and connecting it with the cloud computing platform. This not only provides new ideas and technical means for the recommendation system and management of future digital libraries, but also has a positive impact on improving the overall user experience of the library.

6 Conclusion

In order to improve the accuracy of the book recommendation system in the digital library, the study combined deep learning with the CF algorithm to build a recommendation model CA-NCF based on the channel self-attention mechanism. The research results indicated that the model significantly improved feature extraction and application between users and projects, and the recommendation effect was more accurate. The test results showed that the model achieved the lowest values in terms of mean, optimal value and standard deviation. At the same time, in the loss function experiment of the training set and testing set, the lowest loss function value was achieved with the least number of iterations. The AUC under the ROC was the largest, which was higher than comparison models. In practical applications, it shows excellent sorting ability and recommendation accuracy for different numbers of book recommendations, and the calculation time is excellent. In summary, the cloud computing digital library recommendation system based on CA-NCF performs well in performance tests and has achieved excellent recommendation effects in practical applications. Subsequent research can further

incorporate the high-order feature interaction between items into the impact on user selection. Recommendation tests can be conducted on larger and more diverse heterogeneous data sets to improve its practical performance.

References

[1] C. Chen, L. Yang, and X. Jia, "A bit-object-based method for mining maximum frequent patterns in intensive cloud computing data," Web Intelligence and Agent Systems, vol. 19, no. 2, pp. 125-133, 2021. https://doi.org/10.3233/WEB-210461

[2] Y. Zhao, "Image retrieval model analysis of digital library based on texture characteristics," Advances in Mathematical Physics, vol. 2021, no. 3, pp. 2-11, 2021. https://doi.org/10.1155/2021/6014946

[3] C. Plaisant and N. Dementhon, "The historical CHI video project: bringing 1983-2002 treasures to the ACM Digital Library and YouTube," Interactions, vol. 28, no. 2, pp. 70-75, 2021. https://doi.org/10.1145/3447820

[4] S. Zhang, N. Zheng, and D. Wang, "A novel attention-based global and local information fusion neural network for group recommendation," Research on Machine Intelligence, vol. 19, no. 4, pp. 331-346, 2022. https://doi.org/10.1007/s11633-022-1336-1

[5] D. Li, C. Wang, L. Li, and Z. Zheng, "Collaborative filtering algorithm with social information and dynamic time windows," Applied Intelligence, vol. 52, no. 5, pp. 5261-5272, 2022. https://doi.org/10.1007/s10489-021-02519-8

[6] S. Zhang, L. Wang, R. Fei, X. Xu, and W. Li, "Attraction recommendation based on tourism context modeling and multi-neural collaborative filtering algorithm," IEEJ Transactions on Electrical and Electronic Engineering, vol. 18, no. 8, pp. 1280-1295, 2023. https://doi.org/10.1002/tee.23847

[7] H. Lim and L. Xie, "A new weighted imputed neighborhood-regularized tri-factorization one-class collaborative filtering algorithm: application to target gene prediction of transcription factors," IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 18, no. 1, pp. 126-137, 2021. https://doi.org/10.1109/TCBB.2020.2968442

[8] J. Zhang, J. Yang, L.Wang, Y. Jiang, P. Qian, and Y. Liu, "A novel collaborative filtering algorithm and its application for recommendations in e-commerce," Computer Modeling in Engineering and Science, vol. 1, no. 3, pp. 1275-1291, 2021. https://doi.org/10.32604/cmes.2021.012112

[9] J. Zheng, "Construction and application of music audio database based on collaborative filtering algorithm," Discrete Dynamics in Nature and Society, vol. 2022, no. 8, pp. 1026-1035, 2022. https://doi.org/10.1155/2022/1756357

[10] A. Narwal and S. Dhingra, "A novel approach for Credit-Based Resource Aware Load Balancing algorithm (CB-RALB-SA) for scheduling jobs in cloud computing," Data & Knowledge Engineering, vol. 145, no. 5, pp. 2-14, 2023. https://doi.org/10.1016/j.datak.2022.102138

[11] J. Li and M. Yao, "New framework of digital entrepreneurship model based on artificial intelligence and cloud computing," Mobile Information Systems, vol. 2021, no. 7, pp. 2-12, 2021. https://doi.org/10.1155/2021/3080160

[12] M. Aktan and H. Bulut, "Metaheuristic task scheduling algorithms for cloud computing environments," Concurrency and Computation: Practice and Experience, vol. 34, no. 6, pp. 2-14, 2021. https://doi.org/10.1002/cpe.6513

[13] R. Rani and R. Garg, "Energy efficient task scheduling using adaptive PSO for cloud computing," International Journal of Reasoning-based Intelligent Systems, vol. 13, no. 2, pp. 50-58, 2021. https://doi.org/10.1504/IJRIS.2021.114630

[14] H. K. Omar, M. Frikha, and A. K. Jumaa, "Improving big data recommendation system performance using NLP techniques with multi attributes," Informatica, vol. 48, no. 5, pp. 63-70, 2024. https://doi.org/10.31449/inf.v48i5.5255

[15] A. Rani, K. Taneja, and H. Taneja, "Life insurance-based recommendation system for effective information computing," International Journal of Information Retrieval Research, vol. 11, no. 2, pp. 2-14, 2021. https://doi.org/10.4018/IJIRR.2021040101

[16] F. Yang, "Optimization of personalized recommendation strategy for e-commerce platform based on artificial intelligence," Informatica, vol. 47, no. 2, pp. 253-242, 2023. https://doi.org/10.31449/inf.v47i2.3981

[17] W. Feng, J. Liu, T. Li, Y. Zhen, and W. Di, "FAC: A music recommendation model based on fusing audio and chord features (115)," International Journal of Software Engineering and Knowledge Engineering, vol. 32, no. 11, pp. 1753-1770, 2022. https://doi.org/10.1142/S0218194022500577

[18] D. Banik, S. Satapathy, and M. Agarwal, "Advanced weighted hybridized approach for recommendation system," International Journal of Web Information Systems, vol. 19, no. 1, pp. 1-18, 2023. https://doi.org/10.1108/IJWIS-01-2022-0006

[19] O. Du and Y. Li, "Academic collaborator recommendation based on attributed network embedding," Journal of Data and Information Science, vol. 7, no. 1, pp. 37-56, 2022. https://doi.org/https://doi.org/10.2478/jdis-2022-0005

[20] W. Zhou, Z. Huang, C. Wang, and Y. Chen, "A multi-graph neural group recommendation model with meta-learning and multi-teacher distillation," Knowledge-Based Systems, vol. 276, no. 9, pp. 2-17, 2023. https://doi.org/10.1016/j.knosys.2023.110731