Computing and Informatics, Vol. 43, 2024, 240–260, doi: 10.31577/cai_2024_1_240

INTELLIGENT FUSION RECOMMENDATION ALGORITHM FOR SOCIAL NETWORK BASED ON FUZZY PERCEPTION

Lulu Dong*, Ning Ma

Anhui Open University, Hefei
Anhui, 230022, China
e-mail: `donglulu_1@163.com`

Hao Wang

School of Computer Science and Information Engineering
Hefei University of Technology, Hefei
Anhui, 230601, China

Abstract. In order to improve the effect of intelligent fusion recommendation under the background of social network, this paper combines the fuzzy perception algorithm to research the intelligent fusion recommendation algorithm of social network. Moreover, this paper proposes a task offloading scheme that relies on V2V communication to utilize idle computing resources in a "resource pool". In addition, this paper formulates the computational task execution time as a min-max problem to reduce the storage overhead to optimize the total task execution time. Numerical results show that the proposed scheme greatly reduces the task execution time. The introduced particle swarm optimization algorithm also proves the convergence speed and accuracy of the optimization problem. The research verifies that the intelligent fusion recommendation algorithm for social network based on fuzzy perception has good social network data fusion effect and can effectively improve the effect of intelligent fusion recommendation.

Keywords: Fuzzy perception, social network, fusion, intelligent fusion recommendation, algorithm

* Corresponding author

1 INTRODUCTION

Compared with popular commodities, long-tail commodities have greater marginal utility and can bring higher profits to the company. For recommendation systems, the key to increasing profits lies in the development of long-tail markets. Literature [1] further explains that competitors sell popular products at the same price, resulting in very low profits for such products, while long-tail products are just the opposite. Once successfully purchased, the merchant's revenue will increase. Second, long-tail product sales bring better satisfaction to users. Due to the one-stop shopping convenience effect – by providing customers with the convenience of one-stop access to mainstream objects and niche objects – it can improve consumer satisfaction and make repeat visits possible. The importance of long-tail product recommendations, widely accepted by people, the literature [2] pointed out that the recommendation of long-tail items is an important evaluation index of the recommendation system effectiveness. For long-tail items, the difficulty of recommendation lies in the more prominent degree of data sparseness. From the perspective of the existing research progress, it mainly focuses on clustering, multi-objective optimization, bipartite graph and elimination of popularity bias, etc. Literature [3] clusters long-tail items based on item attributes, and it increases the number of scores available in long-tail recommendations by sharing the scores of long-tail items in the same category. The existing prediction model is used for recommendation. Literature [4] is based on the clustering method grouping similar items, and using this dense data representation for an association rule mining algorithm to improve the quality of cross-selling recommendations. The objective optimization method takes multi-objective optimization as the starting point, and constructs an optimization function according to certain principles, such as accuracy, diversity, popularity, novelty, etc. In addition, it is also considered to use additional information (such as user reviews, user and object attributes, text mining) to mine the relationship between the user's literature and long-tail items. Literature [5] adds additional user or object attributes to the bipartite graph. The dimension generates a tripartite graph, and the purpose is to use this dimension to make random walks have a higher probability of reaching long-tailed items. The existing main methods to eliminate popularity bias are ranking adjustment and unbiased learning. Literature [6] performs a post-event reordering. Literature [7] uses an improved version of xQuAD (Explicit Query Aspect Diversification) to solve the problem of prevalence bias, enabling system designers to tune the system to achieve the desired compromise goal. Both methods are heuristic designs, aiming at intentionally boosting the scores of less popular items, but they lack a theoretical basis for effectiveness. Literature [8] adopts a causal embedding model, using unbiased uniform data to guide the model to learn unbiased embeddings, forcing the model to discard item popularity. However, obtaining such unified data requires random exposure of items to users, which has the risk of damaging user experience. Although the above methods can improve the recommendation effect of long-tail items to a certain extent, they do not consider the effect of users' friends on the recommendation of long-tail items or critical use.

Social recommendation is a recommendation algorithm based on social choice and social influence effect. Social choice means that people tend to contact people with similar attributes, and due to social influence, related individuals in a social network influence each other to become more similar [9]. How to effectively integrate the information between the two has become the core problem of social recommendation. The social recommendation based on neighbors [10] is an intuitive approach, that is, using the number of common friends between two users in the social network to calculate the similarity between users, combine it with the traditional similarity, and then perform the nearest neighbor recommendation or introduce it into the matrix factorization recommendation method. The above fusion strategy involves a large number of users, which brings certain difficulties to online learning, and the matrix factorization method of the fusion social trust network is due to its scalability. It is widely used because of its high flexibility. Some scholars share the user latent feature matrix with the social network and the rating matrix, and combine the two organically and then enter the matrix decomposition process [11, 12]. From the perspective of the existing research progress, the starting point of the above methods is to pursue the recommendation accuracy more, while the social recommendation is not only the pursuit of accuracy, but it can increase the click rate of unpopular items through the trusted recommendation of friends. This paper organically integrates the social network as an important factor affecting long-tail recommendation into the recommendation method to improve the recommendation effect of long-tail items on the premise of maintaining accuracy. The relevant unknown parameters in the model can be obtained by the sub-inference method, and in such a way the prediction function can be realized.

Literature[13] proposed Collaborative Social Topic Regression (CSTR) and applied it to celebrity recommendation. CSTR is a hierarchical Bayesian model that models user behavior information, recommended item semantic information, and recommended item relationship information. At the same time, it also analyzes the difference between the social relationship between celebrities and the social relation- ship between ordinary users. In the literature [13], CSTR simply decomposes the celebrity relationship information, and does not fully exploit the celebrity relationship information. The social relationship decomposed features and the user record matrix decomposed features share a feature vector, which is inconsistent with the actual situation, because ratings and social relations are nonlinearly correlated and distributed differently. The social relations of celebrities consist of followers. There is a certain similarity factor between the behaviors of followers [14]. Celebrities follow other celebrities and browse microblogs written by other celebrities, which can reflect the interests of celebrities. Celebrities are also followed by other celebrities, and the microblogs he writes will be seen by celebrities who follow him, thereby affecting the behavioral characteristics of other celebrities. This article is called the "writing characteristics" of celebrities.

The characteristic reflected in the social relations of celebrities is composed of "read features" and "write features". The analysis of item social relations provides a theoretical basis for decomposing the physical explanation of item relations.

Literature [15] uses Latent Dirichlet Allocation (LDA) to process celebrity description information, but LDA is not ideal for sparse texts [16]. Recently, the research on recommendation systems combined with deep learning has become a hot topic. Literature [17] uses the deep learning method known as stacked denoising autoencoder (SDAE) to replace the LDA model to extract item content features in CTR, and integrates the probability matrix decomposition of the rating information, and proposes a hierarchical Bayesian model Collaborative Deep Learning (Collaborative Deep Learning, CDL). Literature [18] proposes a collaborative filtering model framework which uses the deep learning technology to obtain nonlinear features of user-item interactions, combined with the matrix factorization technology which can obtain linear features of user-item interactions, and it integrates into a system with a stronger generalization ability. However, the above methods do not incorporate the important auxiliary information of social relations into the model.

This paper combines the fuzzy perception algorithm to study the intelligent fusion recommendation algorithm of social network, and improves the recommendation effect under the background of various information fusions.

2 FUZZY PERCEPTION INFORMATION TASK PROCESSING

2.1 Modeling of Optimization Problems

Recommended objects can choose between full local computing and offload computing. The fully local computing mode is the recommended object selection to compute all computing tasks locally without offloading them to service objects. The offload computing mode is to execute part of the computing tasks locally, select some or all of the service objects within a hop from the "resource pool", and offload the other part to the selected service objects. According to the different computing resources of the service object, the service time that can be provided is different. It is necessary to select the task execution mode and task offloading strategy for the recommended objects (that is, which service objects are selected, and what proportion of computing tasks are to be offloaded for each service object). Moreover, it is necessary to reduce the execution time of the computing task to the greatest extent, and meet the requirements of the quality of service (QoS) of the computing task, as shown in Figure 1.

This section takes computing task execution time as the objective function to model the optimization problem, and designs the optimal task offloading mode and offloading assignment strategy for recommended objects under the constraints of mobility between objects and the maximum tolerance value of computing task execution delay. The IDMIIM model only controls the first recommendation object on each network line, and how the remaining tail recommendation objects are transmitted is controlled by the IDM model. For a single object transmitted over a straight network line. The position of object α at time t is $X_\alpha(t)$, and the speed is $V_\alpha(t)$, and the position and speed of the former recommended object $\alpha-1$ at time t are $X_\alpha(t)$ and C, respectively. Capacity of object α is l_α.

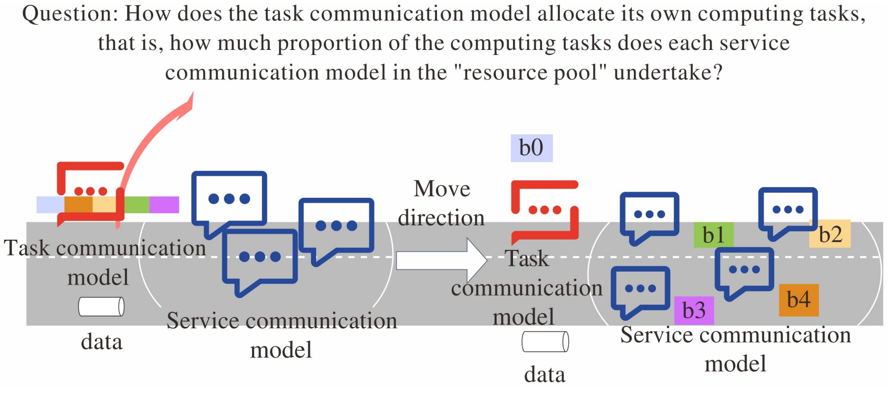


Figure 1. Schematic diagram of the task offloading strategy

It should be emphasized that the velocity here is a vector in physics. We assume that the transmission to the right is a positive direction, then the speed value is greater than 0, indicating that the current object is transmitted to the right. If the velocity value is less than 0, then the current object is transferred to the left under the IDM model, the current acceleration of the object is determined by the current speed $v_\alpha(t)$, the recommended distance $s_\alpha(t) := x_\alpha(t) - x_{\alpha-1} - l_\alpha$ from the previous recommended object, and the speed difference $\Delta v_\alpha(t) := v_\alpha(t) - v_{\alpha-1}(t)$ from the previous recommended object. The relationship is as follows:

$$\frac{dv_\alpha(t)}{dt} = a_\alpha \left[1 - \left(\frac{v_\alpha(t)}{v_\alpha^+}\right)^\delta - \left(\frac{s^+(v_\alpha(t), \triangle v_\alpha(t))}{s_\alpha(t)}\right)^2\right]. \tag{1}$$

Among them, α is the maximum acceleration of the object α,v_α is the acceleration parameter used to adjust the acceleration behavior. $s_\alpha(t)$ is the expected speed when there is no recommended object in front of the object, that is, $s_\alpha(t) \to \infty$: $s'_\alpha + s''(v_\alpha(t), \triangle v_\alpha(t))$ is the ideal minimum distance to the previous recommended object which can be obtained by the following formula:

$$s^*(v_\alpha(t), \triangle v_\alpha(t)) = s'_\alpha + s''_\alpha\sqrt{\frac{v_\alpha(t)}{v_\alpha^*}} + v_\alpha(t)T + \frac{v_\alpha(t)\triangle v_\alpha(t)}{2\sqrt{a_\alpha b_\alpha}}. \tag{2}$$

Among them, s'_α and s''_α are object blocking distance, T is the safety interval time, and b_α is the maximum deceleration of object α. According to the literature, the above formula can generally be simplified by setting s''_α to 0, then the above formula can be simplified to:

$$s^*(V_\alpha(t), \triangle v_\alpha(t)) \approx s'_\alpha + v_\alpha(t)T + \frac{v\alpha(t)\triangle v_\alpha(t)}{2\sqrt{a_\alpha b_\alpha}}. \tag{3}$$

We integrate the above formula with an interval of $\triangle t = 0.4\,\mathrm{s}$ to obtain the calculation formula of velocity and displacement:

$$\begin{aligned} v_\alpha(t+\triangle t) = v_\alpha(t) + \left[\frac{dv_\alpha(t)}{dt}\right] \cdot \triangle t x_\alpha(t+\triangle t) \\ = x_\alpha(t) + \triangle t \cdot v_\alpha(t) + \frac{1}{2}\left[\frac{dv_\alpha(t)}{dt}\right] \cdot (\triangle t)^2. \end{aligned} \tag{4}$$

There is no need to worry about whether the value of $\triangle t$ is too large. According to the literature, a smaller $\triangle t$ will not have any effect on the above formula. Next, we divide $[t, t+t_\alpha]$ with $\triangle t$ as the step size, and the displacement calculation formula can be obtained.

$$\begin{aligned} \triangle x_\alpha(t_\alpha) &= x_\alpha(t+\triangle t_\alpha) - x_\alpha(t) \\ &= [x_\alpha(t+\triangle t_\alpha) - x_\alpha(t)] + [x\alpha(t+\triangle t_\alpha) - x\alpha(t+\triangle t_\alpha)] + \ldots \\ &\quad + [x_\alpha(t+[t_0/\triangle t]\triangle t) - x_\alpha(t+[t_0/\triangle t]-1)\triangle t] \\ &\quad + [x_\alpha(t+\triangle t_\alpha) - x_\alpha(t+[t+\triangle t_\alpha])\triangle t]. \end{aligned} \tag{5}$$

2.2 Communication Model

The service provided externally can be understood as the input data volume data required by the computing task. The service objects in the "resource pool" will listen on the control channel. If it is the selected service object, it will configure the physical layer and media access control (MAC) layer according to the message in the WSA, switch to the corresponding service channel, and join the WBSS to receive the services it provides. It should be noted that the switching between the control channel and the service channel adopts an immediate channel access mode. This switching mode allows long-time IM access on designated channels without regard to time slot boundaries, as shown in Figure 2.

We assume that the recommended object establishes WBSS, the time it takes to publish messages such as WSA on the control channel is *data*, and the time it takes for the recommended object to transmit the amount of input data required for computing tasks is t^i_{SCH}, then t^i_{SCH} can be expressed as:

$$t^i_{SCH} = \frac{data}{Blog(1+\frac{PG_Q}{\sigma^2})}. \tag{6}$$

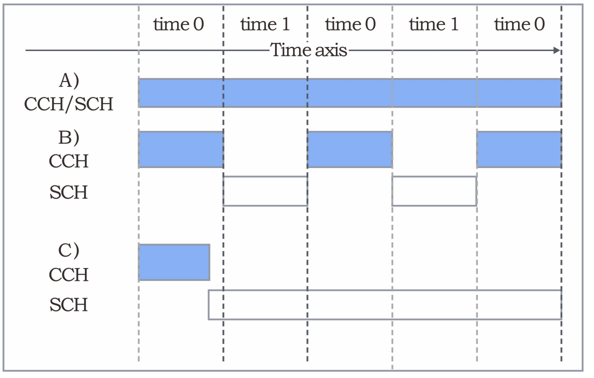


Figure 2. Schematic diagram of handover between CCH and SCH

Among them, B represents the bandwidth of the SCH to be used by the WBS established by the recommended object, P represents the transmission power used by the recommended object announced in the WSA message, G_i represents the channel gain between the recommended object and the service object Vidle, and σ^2 represents the noise power.

For the sake of simplicity in analyzing the problem, the time for the recommended object and the service object to access the control channel and the service channel through the CSMA/CA protocol is ignored. Moreover, the time spent in the channel switching process, such as the guard interval, is also negligible. Therefore, the total time from the input data volume data required by the recommendation object to transmit the calculation task to the service object Vidle, can be expressed as follows:

$$t_i^{tr} = t_{CCH} + t_{SCH}^i. \tag{7}$$

Because the calculation result after the calculation task is processed is generally small, the time for the service object to return the calculation result to the recommendation object is generally negligible.

2.3 Computational Model

1. Calculate the local execution time of the task. When the recommended object selects the fully local computing mode, the computing task execution time is:

$$t_{local} = C/f_0. \tag{8}$$

Among them, f_0 is the computing power of the recommended object.

2. Calculate the execution time of task offloading. When the recommendation object chooses the offloading computing mode, it is assumed that the proportion of computing tasks undertaken by each service object is $b_i \in 0, 1$. When $b = 0$, it

means that the recommended object does not select the service object Vidle to participate in the offloading calculation. When $0 < b_i < 1$, it indicates that the service object Vidle participates in the offload calculation, where $b_i \neq 1$. Because the recommendation object and the service object are ordinary smart objects, there is not much difference in computing power. If $b_i = 1$, it means that the recommendation object offloads the entire computing task to the needs to attach the communication cost required to offload computing tasks to the service object, which is not worth the loss. By defining b, the entire task offloading strategy can be represented by $b = [b_1, b_2, b_3, \ldots, b_N]$. If the computing power of each service object is assumed to be f_i, then the time required for each service object to perform the computing task assigned to it is:

$$t_i^c = \frac{b_i^c}{f_i}. \tag{9}$$

Among them, f_i is the computing power of the service object Vidle. Combined with the communication model described in the previous section, the total time spent by the service object Vidle to process the computing tasks it undertakes is:

$$t_i = t_i^c + t_i^{tr}. \tag{10}$$

Under the above task offloading strategy, the proportion of computing tasks that the recommended object needs to undertake is:

$$b_o = 1 - \sum_{i=1}^{N} b_i. \tag{11}$$

Because the recommendation object itself has all the input data volume data required by the computing task, in the offload computing mode, the total time that the recommendation object needs to spend is:

$$t_0 = c(1 - \sum_{i=1}^{N} b_i)/f_0. \tag{12}$$

The system supports parallel computing, and the computing tasks assigned to each service object in the offload computing mode, including the computing tasks that the recommendation object itself needs to undertake, are all computed in a parallel manner. Therefore, the total time required for the entire computing task is:

$$t_{\triangle edge} = max(t_i), \quad i = 0, 1, 2, \ldots, N. \tag{13}$$

2.4 The Establishment of the Optimization Problem

In order to minimize the execution time of computing tasks, we can establish the following optimization model:

$$minimizeT = s \cdot t_{local} + (1 - s) \cdot t_{edge}. \tag{14}$$

Subject to

C1: $\sum_{i=1}^{N} b_i < 1 (0 \le b_i \le 1)$,
C2: $t_i \le t_{res \cdot i} (i = 1, 2, \ldots, N)$,
C3: $|\triangle X_i| \le |x_0(t) - x_i(t)| + \sqrt{dis^2 - (y_b(t) - y_i(t))^2}$.

Among them, $s = 1$ indicates that the recommended object chooses the completely local computing mode, and $s = 0$ indicates that the recommended object chooses the offloading computing mode. $\sum_{i=1}^{N} b_i$ represents the maximum stay time of the service object Vidle in the "resource pool". $(x_0(t), y_0(t))$ and $(x_i(t), y_i(t))$ respectively represent the position of the recommended object and the service object Vidle, at time t, and $\triangle x_i$ represents the relative displacement of the recommended object and the service object Vidle after time t.

For Constraint 1, just like the reason for explaining $b_i \neq 1$ when defining b, in the same way, the value $\sum_{i=1}^{N} b_i$ added up to all the computing tasks allocated to the service object will not be equal to 1, but will only be less than 1. Constraint 2 is to ensure that the service object Vidle does not leave the "resource pool" until it has completed the computational tasks assigned to it. Constraint 3 is to ensure that the service object Vidle keeps within the communication range with the recommended object in the process of computing the computing tasks assigned to it. Because of the movement of the object, the distance $\triangle x_i$ between the recommended object and the service object changes with time, as shown in Figure 3. According to the calculation displacement formula given in the movement model, we can get:

$$\triangle x_i = \triangle x_i(t_i) - \triangle x_0(t_i) = [x_i(t + t_i) - x_i(t)] - [x_0(t + t_i) - x_0(t)]. \tag{15}$$

In order to ensure that the offloading calculation is successful, the service object will be within the one-hop communication range of the recommended object at the beginning, and will not jump out of the communication range of the recommended object. Therefore, we use the communication distance as the radius and draw a circle with the recommended object as the center to limit the movement range of the service object. Because we assume that the object is transmitted on a straight network line, in order to strictly ensure the success of the offload calculation, we constrain the service object and the recommended object with the strictest conditions, that is, both move along a straight line, and the transmission path will not have any radians. According to the cosine theorem, we can calculate that the relative displacement $\triangle x_i$ of the recommended object and the service object cannot exceed $|x_0(t) - x_i(t)| + \sqrt{dis^2 - (y_0(t) - y_i(t))^2}$.

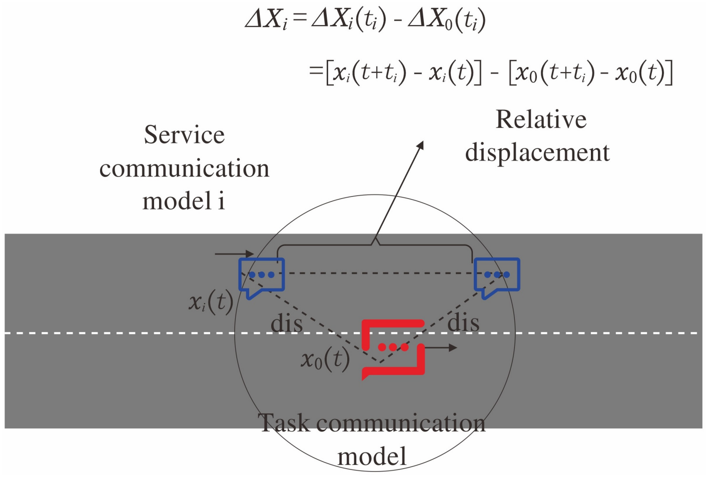


Figure 3. Schematic diagram of Constraint 3

2.5 Solution of Optimization Problems

Considering that all service objects in the "resource pool" participate in uninstallation, the solution is as follows.

Step 1: The algorithm first considers that all computing tasks are calculated locally, that all computing tasks are calculated locally, that is, $s = 1$, then the algorithm can obtain:

$$T = t_{local} = c/f_0. \tag{16}$$

Step 2: The algorithm considers the recommended object to select the unloading calculation mode, that is, $s = 0$, then it can be obtained:

$$T = t_{edge} = \min\{\max(t_i)\}, i = 0, 1, 2, \ldots, N. \tag{17}$$

Step 3: The algorithm compares the size of T obtained in the two calculation modes. If $t_{local} < t_{edge}$, the recommender chooses the fully local computing mode, and if $t_{local} > t_{edge}$, then the recommender chooses the offload computing mode. It can be seen from the above solution ideas that the optimization problem in the second step is the most complicated, and different T values will be obtained by selecting different task offloading strategies $b = \{b_1, b_2, b_3, \ldots, b_N\}$. Here, we adopt the max-min fairness algorithm to solve it.

According to the max-min fairness algorithm, the first step is to ensure that the total time t_i $(i = 1, 2, \ldots, N)$ spent by each service object to process the computing task it undertakes is the same as the time t_0 spent by the recommended object without considering the constraints 2 and 3, that is,

$$t_0 = t_1 = t_2 = \cdots = t_N. \tag{18}$$

According to $t_0 = t_1 = t_2 = \cdots = t_N$, we can get:

$$b_{equal1} \cdot \frac{c}{f_1} + t_1^{tr} = b_{equal2} \cdot \frac{c}{f_2} + t_2^{tr} = \cdots = b_{equalN} \cdot \frac{c}{f_N} + t_N^{tr}. \tag{19}$$

In the formula, b_{equali} represents the proportion of computing tasks that each service object Vidle needs to undertake under the guidance of the maximum and minimum fairness algorithm. According to the above formula, we can get the relationship between b_{equali} and c:

$$b_{equali} = f_i \left(\frac{b_{equal1}}{f_1} + \frac{t_1^{tr} - t_i^{tr}}{c} \right). \tag{20}$$

The time t spent according to the recommended object is the same as the total time c spent by the service object to process the computing tasks it undertakes, that is, $t_0 = t_1$. We can get:

$$\frac{c}{f_0} = \sum_{i=1}^{N} \left(b_{equali} \cdot \frac{c}{f_i} + t_i^{tr} \right). \tag{21}$$

Substituting formula (20) into formula (21), we can get:

$$b_{equali} = \frac{c/f_0 - t_1^{tr} - \sum_{i=1}^{N} \frac{f_i}{f_0}(t_1^{tr} - t_i^{tr})}{\frac{c}{f_1} + \frac{c}{f_0} \sum_{i=1}^{N} \frac{f_i}{f_1}}. \tag{22}$$

Then we can conclude that when the computing task is allocated to the service object for the first time, the time required for each service object to undertake its assigned computing task is:

$$T_{equal} = t_1 = t_2 = \cdots = t_N = b_{equal1} \frac{c}{f_1} + t_i^{tr}. \tag{23}$$

However, according to Constraint 2 and Constraint 3, we can know that each service object may not necessarily provide such a long service time, that is, t_i ($i = 1, 2, \ldots, N$) is limited in size. According to the calculation formula of $\triangle x$ in Constraint 3, we can know that the relationship between $\triangle x_i$ and t_i is a quadratic inequality, which is solvable. We assume that the solution is $0 \leq t_i \leq t_{constraintc3\_i}$. Then, combined with Constraint 2, we can get $0 \leq t_i \leq t_{max\_i}$ ($t_{max\_i} = \min\{t_{res\_i}, t_{constraintc3\_i}\}$). By sorting t_{max_i} from smallest to largest, we get $t_{max\_1} \leq t_{max\_2} \leq t_{max\_3} \leq \cdots \leq t_{max\_N}$. By comparing T_{equal} to it, we get $T_{equal} \leq t_{max\_i}$ ($1 \leq i$). Therefore, it can be concluded from this that the amount value is too large and needs to be adjusted.

First of all, we need to get the amount of tasks assigned to the service object that is overloaded. Taking the service object Vidle as an example, if it is assumed that the maximum amount of tasks that the service object e_1 can undertake is b_{max_1}, its value can be obtained by the following formula:

$$t_{max\_1} = b_{max\_1} \cdot \frac{c}{f_1} + t_1^{tr} \Longrightarrow b_{max\_1} = \frac{t_{max\_1} - t_1^{tr}}{c} f_1. \tag{24}$$

Therefore, the amount of tasks overloaded by the service object Vidle e_1 is $b_{extra\_1} = b_{equal1} - b_{max\_1}$, and the total amount of tasks allocated to the overloaded service object Vidle e_i is:

$$b_{extra} = \sum_{i=1}^{l} b_{extra\_i}. \tag{25}$$

The extra task will continue to be allocated to the service object Vidle e_{l+1}, Vidle e_{l+2}, Vidle e_{l+3}, ..., Vidle e_{k+N}, according to the maximum and minimum fairness algorithm. Similar to the equal distribution above, we can get the following formula:

$$b_{equal2_i} = \frac{f_i}{f_{i+1}} b_{equal2_{l+1}} + \frac{t_{l+1}^{tr} - t_i^{tr}}{c} f_i, \tag{26}$$

$$b_{equal2_{l+1}} \cdot \frac{c}{f_{l+1}} t_{l+1}^{tr} = \frac{c}{f_0} \left(b_{extra} - \sum_{i=l+1}^{l+N} b_{equal2_i} \right), \tag{27}$$

$$b_{equal2_{l+1}} = \frac{b_{extrac/f_0} - t_{i+1}^{tr} - \sum_{i=l+1}^{l+N} \frac{f_l}{f_0} (t_{l+1}^{tr} - t_i^{tr})}{\frac{c}{f_{l+1}} + \frac{c}{f_0} \sum_{i=i+1}^{l+N} \frac{f_i}{f_i+1}}. \tag{28}$$

Among them, b_{equal2_i} represents the proportion of tasks assigned to each service object on service object Vidle e_i $(i = l+1, l+2, l+3, \ldots, l+N)$ by the amount of b_{extra} tasks. So far, the proportion of total tasks assigned to service object Vidle e_i is $\sum_{i=l+1}^{l+N} b_{equal2_i}$.

Allocating b_{extra} task volume to service object Vidle e_{l+1}, Vidle e_{l+2}, Vidle e_{l+3}, ..., Vidle e_N may cause service objects to be overloaded. We also need to check each service object, and compare with its current b according to the maximum amount of computing tasks that each service object can undertake. If there is also overload, we need to follow the above adjustment method to redistribute until all service objects are not overloaded, and get the final task offloading strategy $b = \{b_1, b_2, b_3, \ldots, b_N\}$.

According to the particle swarm optimization algorithm and the optimization problem we need to solve, it is assumed that the particle swarm size is l, that is, there are l particles. We define the position of the j^{th} particle in the $N+1$ dimension space as $B_j^k = (b_{j0}^k, b_{j1}^k, \ldots, b_{jN}^k), j = 1, 2, \ldots, l$ and the velocity as $V_j^k = (v_{j0}^k, v_{j1}^k, \ldots, v_{jN}^k), j = 1, 2, \ldots, l$ during the k^{th} iteration. The fitness function is the optimization problem $T_j^k = t_{edge}(B_j^k), j = 1, 2, \ldots, l$ that needs to be solved. The individual extremum generated by the search of the j^{th} particle from the initial to the current iteration history is defined as $pbest_j^k = (p_{j0}^k, p_{j1}^k, \ldots, p_{jN}^k)$, and the global extremum is defined as $gbest_j^k = (g_{j0}^k, g_{j1}^k, \ldots, g_{jN}^k)$.

The formula for each particle to update its own velocity and position in the iterative process can be expressed as follows:

$$V_j^{k+1} = \omega V_i^k + c_1 r_1 (pbest_j^k - B_j^k) + c_2 r_2 (gbest^k - B_j^k) B_j^{k+1} = B_j^k + V_j^{k+1}. \tag{29}$$

Among them, ω is the inertia coefficient, c_1 and c_2 are learning factors, and r_1 and r_2 are uniformly distributed random numbers that are independent and identically distributed in the $(0, 1)$ interval. (1) When updating the position of the j^{th} particle to obtain B_j^{k+1} according formula (29), we also need to consider three constraints in the optimization problem. For Constraint 1, we must ensure that the proportion of computing tasks allocated to all service objects is not greater than or equal to 1. Well, we first sum the B_j^{k+1} vector. If the summation result is greater than or equal to 1, then for the proportion of the more allocated tasks, we will delete it according to its proportion, taking b_{j1}^{k+1} as an example.

$$B_{j1}^{k+1} = b_{j1}^k - \frac{b_{j1}^k}{\sum_{i=0}^N b_{ji}^k} \left(\sum_{i=0}^N b_{ji}^k - 1 \right). \tag{30}$$

For Constraint 2 and Constraint 3, we can know that the two constraints can be summed up as $0 \leq t_i \leq t_{max\_i}$ ($t_{max\_i} = \min\{t_{resi}, t_{constraintc3\_i}\}$), and then find out that the maximum amount of computing tasks that each service object can undertake is b_{max_i}. We compare the modified b_{ji}^{k+1} in the previous step with its corresponding b_{max_i} in turn. If $b_{ji}^{k+1} > b_{max\_i}$, then we will directly make it equal to b_{max_i}, otherwise it will not be modified. (2) When updating the individual extreme value of a single particle according to the fitness function, we also take into account the number of service objects participating in the offloading calculation. In other words, if the position of the j^{th} particle is updated according to formula (30) during the iteration of step $k+1$, $T_j^{k+1} = t_{edge}(B_j^{(k+1)})$ is obtained. In the traditional particle swarm optimization algorithm, if $T_j^{(k+1)} < T_j^k$, then update the individual extreme value of the j^{th} particle, namely $pbest_j^{k+1} = B_j^{k+1}$, otherwise the individual extreme value remains unchanged, namely $pbest_j^{k+1} = pbest_j^k$. However, we now change the conditions for updating the individual extreme values, that is, not only $T_j^{k+1} < T_j^k$, but also to ensure that the number of b_j^{k+1} greater than 0 in the new B_j^{k+1} does not increase. This ensures that the number of service objects participating in unloading will not increase when the task execution time is reduced to the same extent.

3 INTELLIGENT FUSION RECOMMENDATION ALGORITHM FOR SOCIAL NETWORK BASED ON FUZZY PERCEPTION

Users have explicit or implicit feedback on the previous batch of inference results, and this instant discrimination can further optimize the data model constructed by the recommendation algorithm, thereby improving the accuracy of the recommendation service. Figure 4 shows a basic recommendation service workflow principle. The target user that has not been rated by the target user is used as a candidate, and the score of the target user on the candidate item is inferred by combining the ratings of other users and the similarity of users in the user group. Finally, sorting is performed according to the calculated item scores, and items above the threshold are recommended to users. The following is an example to illustrate it, as shown in Figure 5.

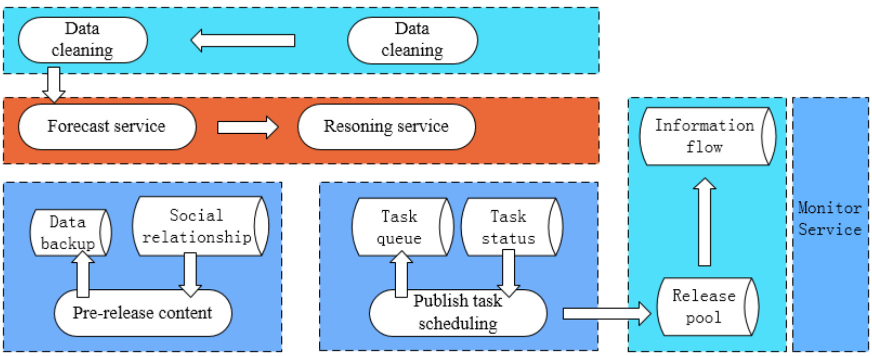


Figure 4. Schematic diagram of basic recommendation service

The highly available recommendation system needs to meet these four characteristics: respond to user requests in real time, record the user feedback accurately and comprehensively, perform the model operations efficiently, and perform pluggable experiments with multiple strategies. All that folowing the information flow architecture design tradition. Figure 6 shows a comprehensive and a brief recommendation system architecture. The concurrent design of data crawling tools is not only about coroutines, but it also divides the different usage scenarios of coroutines and multithreading to manage multithreaded resources more directly. The specific program structure is shown in Figure 7.

The communication rounds of social network nodes are set to 100 rounds, and there are 110 communication nodes and 6 bug nodes in the network. According to the above simulation environment and parameter settings, the adaptive recommendation simulation of the integrated social network is carried out, and the original user behavior data mining results are obtained, as shown in Figure 8. In order to verify the application performance of this method in realizing social network recomendation, simulation experiments are carried out. The

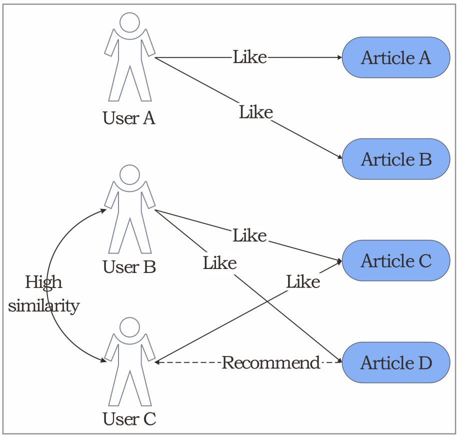


Figure 5. Schematic diagram of user-based collaborative filtering algorithm

experiment is designed by MATLAB, and the simulation analysis of social network recommendation is carried out. Using the method of web crawler, the time domain distribution of network fusion information is obtained, as shown in Figure 9. Taking the data collected in Figure 2 as the research object, the automatic matching design of user identity information and network information is carried out, and the social network intelligent fusion recommendation is realized according to the identity matching results, and the matching output is shown in Figure 10.

The above research verifies that the intelligent fusion recommendation algorithm for social network based on fuzzy perception has good social network data fusion effect, and it can effectively improve the effect of the intelligent recommendation.

4 CONCLUSION

In the era of information explosion, it is difficult for users to efficiently obtain all kinds of information about goods and services that may be of interest, and it is also difficult for merchants to accurately display relevant objects to target users. Therefore, the recommender systems are gradually playing an important role in e-commerce sites such as Amazon. However, the existing recommendation algorithms often ignore the recommendation of unpopular items, that is, the existing methods are more inclined to recommend popular items, unpopular items, namely long-tail items, which have unique value. This paper combines the fuzzy perception algorithm to research the social network fusion intelligent recommendation algorithm to improve the recommendation effect under the background of various information fusion. The research verifies that the intelligent fusion recommendation algorithm for social network based on fuzzy perception has good social network

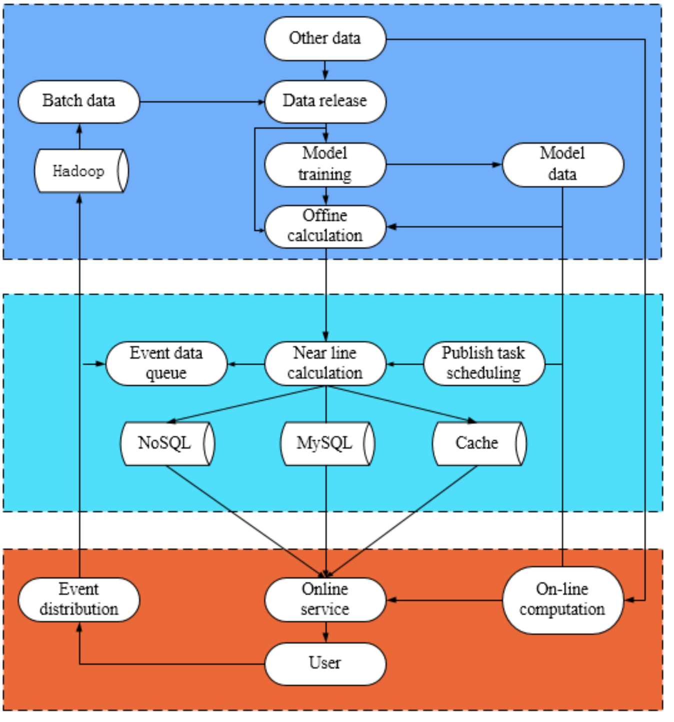


Figure 6. Architecture diagram of the information flow recommendation system

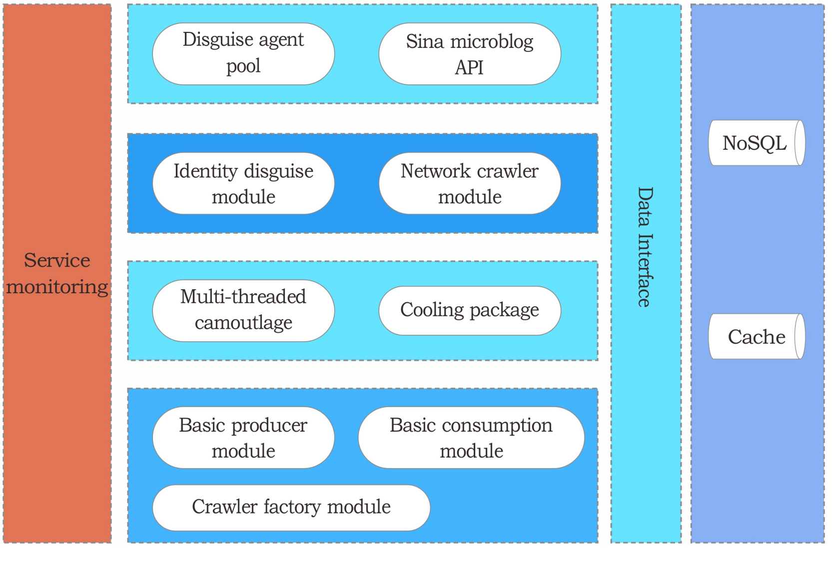


Figure 7. Basic architecture of data crawling tools

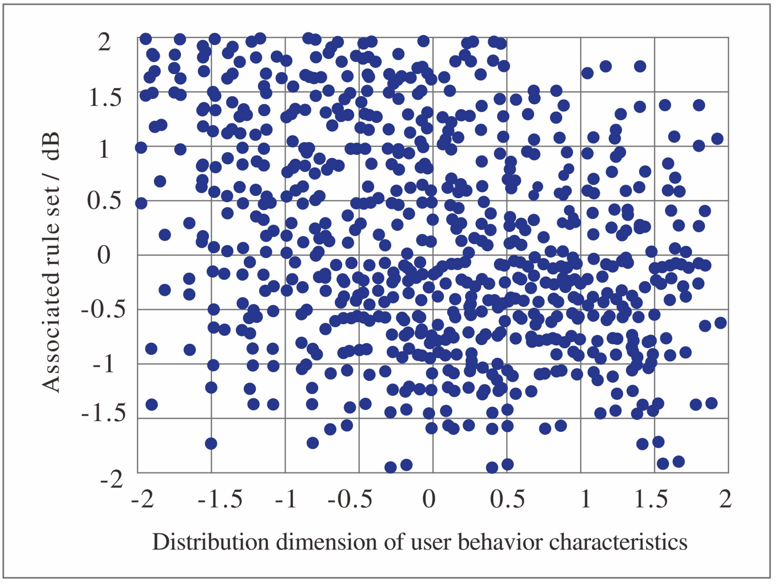


Figure 8. Data mining results of user behavior

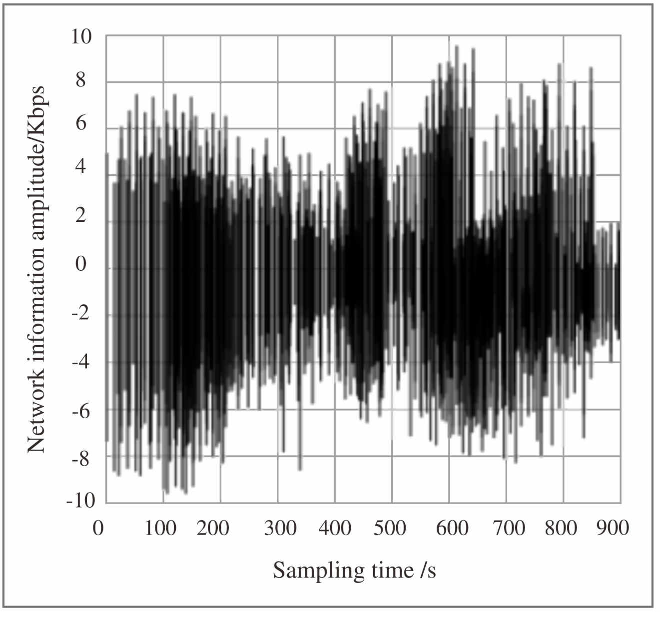


Figure 9. Information crawler sampling

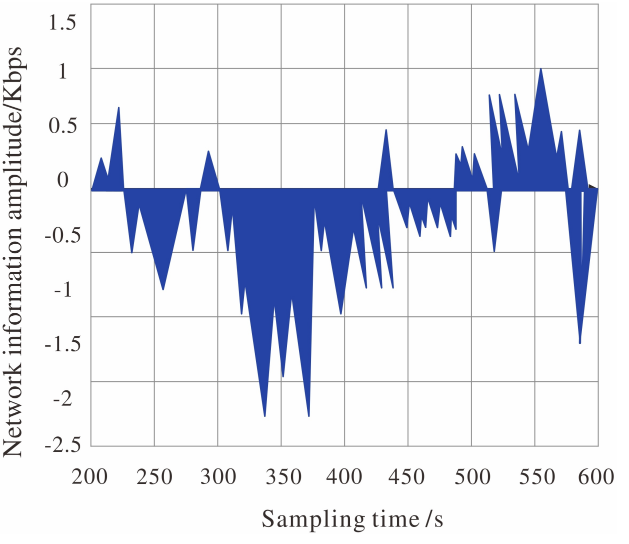


Figure 10. Information matching results of social network recommendation

data fusion effect, and can effectively improve the effect of intelligent recommendation.

5 DECLARATIONS

Funding:

1. The research was funded by the Key Project of Natural Science of Anhui Province "Research on Support Model and Realization Mechanism for Adaptive Learning Based on Educational Data Mining" (No. 2023AH051144).
2. The research was funded by the School-Level Project "Research on Personalized Recommendation Based on Interest Modeling" (No. JY22-04).

Conflict of interest: There is no conflict of interest among the authors.

Data availability: All data generated or analysed during this study are included in the manuscript.

Code availability: Not applicable.

Author' s contributions: Lulu Dong, Ning Ma and Hao Wang contributed to the design and methodology of this study, the assessment of the outcomes and the writing of the manuscript.

REFERENCES

[1] Yang, F.: A Hybrid Recommendation Algorithm-Based Intelligent Business Recommendation System. Journal of Discrete Mathematical Sciences and Cryptography, Vol. 21, 2018, No. 6, pp. 1317–1322, doi: 10.1080/09720529.2018.1526408.

[2] Bai, B.—Fan, Y.—Tan, W.—Zhang, J.: DLTSR: A Deep Learning Framework for Recommendations of Long-Tail Web Services. IEEE Transactions on Services Computing, Vol. 13, 2020, No. 1, pp. 73–85, doi: 10.1109/TSC.2017.2681666.

[3] Sigova, M. V.—Klioutchnikov, I. K.—Klioutchnikov, O. I.: Long-Tail Data-Driven Recommendations - Innovative Solutions for Financial Recommender Systems. 2021 International Conference Engineering Technologies and Computer Science (EnT), IEEE, 2021, pp. 92–97, doi: 10.1109/EnT52731.2021.00023.

[4] Zhang, L.—Priestley, J.—DeMaio, J.—Ni, S.—Tian, X.: Measuring Customer Similarity and Identifying Cross-Selling Products by Community Detection. Big Data, Vol. 9, 2021, No. 2, pp. 132–143, doi: 10.1089/big.2020.0044.

[5] Johnson, J.—Ng, Y. K.: Enhancing Long Tail Item Recommendations Using Tripartite Graphs and Markov Process. Proceedings of the International Conference on Web Intelligence (WI '17), ACM, 2017, pp. 761–768, doi: 10.1145/3106426.3106439.

[6] Lu, X.—Xu, Y.—Tian, Y.—Cetiner, B.—Taciroglu, E.: A Deep Learning Approach to Rapid Regional Post-Event Seismic Damage Assessment Using Time-Frequency Distributions of Ground Motions. Earthquake Engineering and Structural Dynamics, Vol. 50, 2021, No. 6, pp. 1612–1627, doi: 10.1002/eqe.3415.

[7] Yigit-Sert, S.—Altingovde, I. S.—Macdonald, C.—Ounis, I.—Ulusoy, Ö.: Supervised Approaches for Explicit Search Result Diversification. Information Processing and Management, Vol. 57, 2020, No. 6, Art. No. 102356, doi: 10.1016/j.ipm.2020.102356.

[8] Huang, Y.—Chai, Y.—Liu, Y.—Shen, J.: Architecture of Next-Generation E-Commerce Platform. Tsinghua Science and Technology, Vol. 24, 2019, No. 1, pp. 18–29, doi: 10.26599/TST.2018.9010067.

[9] Tang, J.—Hu, X.—Liu, H.: Social Recommendation: A Review. Social Network Analysis and Mining, Vol. 3, 2013, pp. 1113–1133, doi: 10.1007/s13278-013-0141-9.

[10] Xin, M.—Wu, L.: Using Multi-Features to Partition Users for Friends Recommendation in Location Based Social Network. Information Processing and Management, Vol. 57, 2020, No. 1, Art. No. 102125, doi: 10.1016/j.ipm.2019.102125.

[11] He, G.: Enterprise E-Commerce Marketing System Based on Big Data Methods of Maintaining Social Relations in the Process of E-Commerce Environmental Commodity. Journal of Organizational and End User Computing (JOEUC), Vol. 33, 2021, No. 6, pp. 1–16, doi: 10.4018/JOEUC.20211101.oa16.

[12] Xu, J.—Hu, Z.—Zou, J.: Personalized Product Recommendation Method for Analyzing User Behavior Using DeepFM. Journal of Information Processing Systems, Vol. 17, 2021, No. 2, pp. 369–384, doi: 10.3745/JIPS.01.0069.

[13] Ding, X.—Jin, X.—Li, Y.—Li, L.: Celebrity Recommendation with Collaborative Social Topic Regression. In: Rossi, F. (Ed.): Proceedings of the Twenty-Third Inter-

national Joint Conference on Artificial Intelligence (IJCAI '13). AAAI Press, 2013, pp. 2612–2618, https://www.ijcai.org/Proceedings/13/Papers/385.pdf.

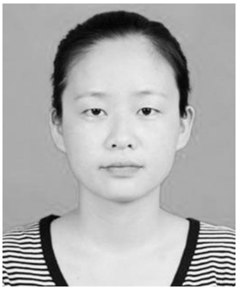
[14] Zimmerman, J.—Parameswaran, L.—Kurapati, K.: Celebrity Recommender. 2002, doi: 10.1184/R1/6469880.v1.

[15] Wang, H.—Hong, Z.—Hong, M.: Research on Product Recommendation Based on Matrix Factorization Models Fusing User Reviews. Applied Soft Computing, Vol. 123, 2022, Art. No. 108971, doi: 10.1016/j.asoc.2022.108971.

[16] Liu, H.—Yang, L.—Ling, C.—Wu, X.: Collaborative Social Deep Learning for Celebrity Recommendation. Intelligent Data Analysis, Vol. 22, 2018, No. 6, pp. 1375–1394, doi: 10.3233/IDA-173674.

[17] Wang, H.—Wang, N.—Yeung, D. Y.: Collaborative Deep Learning for Recommender Systems. Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15), 2015, pp. 1235–1244, doi: 10.1145/2783258.2783273.

[18] Wu, L.—Sun, P.—Hong, R.—Ge, Y.—Wang, M.: Collaborative Neural Social Recommendation. IEEE Transactions on Systems, Man, and Cybernetics: Systems, Vol. 51, 2021, No. 1, pp. 464–476, doi: 10.1109/TSMC.2018.2872842.



Lulu Dong is Associate Professor at Anhui Open University. Her research interests include data mining, natural language processing, machine learning, artificial intelligence, etc.

Ning Ma is Professor at Anhui Open University. His research interests include data mining, cloud computing, artificial intelligence, etc.

Hao Wang is Professor at the Hefei University of Technology. His research interests include distributed intelligent systems, robots, data mining, etc.