Postprint of Social Network Clustering Algorithm Based on Graph Clustering and Ant Colony Algorithm

Authors: Ye Xiaoying, Wan Mei, Tang Rong, Xie Yun, Chen Guihong, Li Qiang

Date: 2019-05-10T00:00:00+00:00

Abstract

To address the directedness and diversity of social relationships in social networks, a social network clustering algorithm based on graph clustering and ant colony optimization is proposed. First, a directed, non-fully-connected two-dimensional graph model is established for the social network under the constraint of network coverage. Then, the K-medoids algorithm is employed to search for central users of user groups, while the artificial ant colony algorithm is used to search for similarities between each user and the central users in the 2D graph, and users satisfying the similarity threshold are grouped into the same user group. A prediction mechanism for low-activity users is designed to address the network sparsity problem and cold-start problem. Furthermore, the trade-off between clustering accuracy and coverage is balanced through the constraint condition of network coverage. Simulation experimental results demonstrate that the algorithm achieves favorable social network clustering performance and effectively alleviates the sparsity problem and cold-start problem.

Full Text

Preamble

Vol. 37 No. 6
Application Research of Computers
ChinaXiv Cooperative Journal

Clustering Algorithm for Social Networks Based on Graph Clustering and Ant Colony Optimization

Ye Xiaoying[1], Wan Mei[2], Tang Rong[3], Xie Yun[1], Chen Guihong , Li Qiang[1]

[1]Dept. of Computer Science & Technology, Neusoft Institute of Guangdong, Foshan, Guangdong 528225, China
[2]Dept. of Computer Science & Engineering, Guangzhou College of Technology & Business, Guangzhou 510850, China
[3]Jiulongpo District Mental Health Center, Chongqing 400052, China
School of Electronics & Information Technology, Sun Yat-sen University, Guangzhou 510006, China

Abstract: Aiming at the directional and diverse nature of social relationships in social networks, this paper proposes a social network clustering algorithm based on graph clustering and ant colony optimization. First, under the constraint of network coverage, a directed, non-fully-connected two-dimensional graph model is constructed for social networks. Then, the K-medoids algorithm is employed to search for central users of user groups, while the ant colony optimization algorithm searches for similarities between each user and central users in the 2D graph, grouping users that meet the similarity threshold into the same user group. A prediction mechanism for low-activity users is designed to address network sparsity and cold-start problems. Additionally, network coverage constraints are used to balance the trade-off between clustering accuracy and coverage. Simulation experimental results demonstrate that the proposed algorithm achieves favorable clustering performance in social networks and effectively alleviates sparsity and cold-start problems.

Keywords: social networks; data mining; clustering process; ant colony optimization; graph clustering; trust information

0 Introduction

With the widespread adoption of applications such as Weibo, WeChat, Douban Movies, and NetEase Cloud Music, social networks in various domains have developed rapidly. Current social networks exhibit multiple types of social relationships, including friendships, follow relationships, and shared preferences [1]. Both nodes and connections in social networks have diverse attributes, yet traditional network clustering methods primarily consider link density while neglecting the diversity of social networks. Furthermore, the presence of low-activity users in social networks adversely affects clustering effectiveness [2].

In addition to link density-based social network clustering algorithms [3,4], recent approaches have emerged that consider node diversity, strong and weak social ties, and various hidden information. Reference [5] focuses primarily on user interest similarity, calculating similarity based on a Bayesian probability model. In today' s diverse social networks, interest has become a weak association, and factors such as trust propagation, comment information, and rating information should also be considered. Reference [6] proposes a directed network clustering algorithm based on structural similarity that, accounting for the directed interaction nature of social networks, considers nodes' incoming

neighbors and defines direct structural reachability using directed edges. Reference [7] employs particle swarm optimization to optimize social networks, using network structure as the objective function for the swarm and guiding the evolution process through a greedy strategy. However, both [6] and [7] rely solely on direct social relationships in network construction.

Contemporary social networks exhibit diverse associations. Beyond strong relationships, various weak relationships should be considered, including follow relationships, trust propagation [8], comment information, and rating information. Moreover, social networks contain both active and low-activity users, where low-activity users cause sparsity problems that subsequently affect clustering accuracy and coverage [9]. To address these issues, we propose a Graph Clustering and Ant Colony Optimization Social Networks Clustering Algorithm (GC-ACO). A two-dimensional graph is constructed under coverage constraints to ensure a balance between coverage and clustering accuracy. The graph construction process incorporates diverse information including direct trust relationships, trust propagation, and comment information. By combining Pearson similarity with diverse social relationships, we aim to solve the sparsity problem, and we design a prediction mechanism for low-activity users to address the cold-start problem. During the clustering phase, the ACO algorithm searches for users most similar to central users, thereby improving clustering accuracy.

---

1 Two-Dimensional Graph Model

Constructing a two-dimensional graph under coverage constraints can effectively alleviate the sparsity problem in social networks. Similarity measurement effectiveness heavily depends on user comment information; therefore, improving the reliability of similarity measurement enhances clustering reliability and precision. Trust-aware social networks improve clustering accuracy by predicting information for low-activity users. The basic assumption is that users are easily influenced by those they trust highly, though this mechanism tends to reduce coverage. Many researchers have found that users are influenced not only by directly trusted users but also by indirect users, with influence decreasing as the distance between users increases—a theory known as trust propagation.

We design a graph model based on trust and similarity, with the construction algorithm shown in Algorithm 1. Graph nodes represent users; edges represent connections between users; connections are dual-weighted, expressed as tuples (W ,W) = (pcc(u,v), T(u,v)), where pcc denotes similarity measurement and T denotes trust propagation. The algorithm inputs include direct trust information, indirect trust information, Pearson Correlation Coefficient (PCC), and Maximum Propagation distance (MP), outputting a social network graph represented by an adjacency matrix. Trust propagation is calculated based on the shortest path between users, and the setdiff() function cancels existing new connections. The coefficient 1/i in line 6 indicates that the longer the distance

between two users, the smaller the trust value.

Algorithm 1: Social Network Graph Construction Algorithm
Input: PCC graph, trust graph, MP /* MP is the maximum trust propagation distance */
Output: W_graph

1. users = number of users
2. tmp = I_users×users /* initialize temporary user matrix */
3. mt = 0_users×users /* initialize user matrix */
4. for i = 1 to MP do {
5. `tmp = tmp * T`
6. `mt = mt + (1/i) * setdiff(mt, tmp) /* calculate differences between trusted users */`
7. }
8. foreach (u , u) do { // traverse each user pair
9. `if both PCC(u , u ) and MT(u , u ) exist { /* both PCC graph and MT value exist */`
10. `(W , W ) = (PCC(u , u ), 0)`
11. `} else if PCC(u , u ) exists {`
12. `(W , W ) = (PCC(u , u ), 0)`
13. `} else if MT(u , u ) exists {`
14. `(W , W ) = (0, MT(u , u ))`
15. `}`
16. `W_graph(u , u ) = (W , W )`
17. }

[Figure 1: see original paper] shows a social network example with eight users. Figure 1(a) is the PCC similarity graph; (b) is the direct trust relationship graph; (c) is the trust propagation graph; (d) is the final constructed graph model. As shown, the social network graph may not be fully connected. According to six degrees of separation theory [10], the longest distance between two connected users is six hops. Integrating Figures 1(a), (b), and (c) into Figure 1(d) may establish connections for isolated partitions, helping improve coverage. In the figure, u denotes users, and edge values in Figures 1(b) and (c) represent pcc

values between users. User u trusts u , but u does not trust u , so the weight of u →u is (pcc(u ,u), T(u ,u)) = (pcc(u ,u), 1), while the weight of u →u is (pcc(u ,u), 0), making the graph directed.

---

2 Background on Ant Colony Algorithm

Ant Colony Optimization (ACO) [11] is a multi-agent system that can solve problems in a distributed manner with strong global search and local exploitation capabilities. ACO first models a problem as a weighted graph and then searches for optimal paths in the graph. Artificial ants generate feasible solutions through traversal, exchange information with each other, and release pheromones on edges or nodes.

The pheromone release process by ant k in ACO can be defined as:

$$\tau_{ab}^{k} = \begin{cases} (1-\rho)\tau_{ab} + \frac{Q}{L_k} & \text{if ant k uses edge } e_{ab} \\ (1-\rho)\tau_{ab} & \text{otherwise} \end{cases}$$

where n is the number of users in the network and k is the average degree of the network. Assuming sim(a,u) represents the similarity between a and u, the PCC-based similarity calculation is:

$$sim(a,u) = \frac{\sum_{i \in A_{a,u}} (r_i(a) - \bar{r}(a))(r_i(u) - \bar{r}(u))}{\sqrt{\sum_{i \in A_{a,u}} (r_i(a) - \bar{r}(a))^2} \sqrt{\sum_{i \in A_{a,u}} (r_i(u) - \bar{r}(u))^2}}$$

where r (u) is user u' s rating for item i; $\bar{r}(u)$ is user u' s average rating; and A , is the set of items rated by both users a and u. Ultimately, users with similarity above threshold are grouped into the same user group.

---

3 Algorithm Design

3.1 Generating Appropriate Number of User Groups

Social network coverage is not correlated with the number of groups; therefore, the first group number discovered by the search can be used as the clustering algorithm' s group count.

3.2 Determining Central Users of Groups

The K-medoids algorithm [12] is used to search for central users of user groups. The K-medoids objective function (F) is defined as:

$$F = \sum_{c \in C} \sum_{m \in c} \min_{n \in c} dist(m, n)$$

where C is the set of clusters and dist(m,n) represents the distance between users m and n in the two-dimensional graph. Since each edge in the graph has dual weights, the distance between users is calculated as:

$$dist(u, v) = \sqrt{d_S(u, v)^2 + d_T(u, v)^2}$$

where u and v are target users; d is the similarity distance, calculated as:

$$d_S(u, v) = 1 - W_S(u, v)$$

and d is the trust distance, calculated as:

$$d_T(u, v) = 1 - W_{MT}(u, v)$$

3.3 Similarity Computation and Weighting

After selecting central users, the next step finds user groups with high similarity to these centers. This process mainly includes three steps: ranking processing, weighting processing, and prediction processing.

3.3.1 Initial Ranking Processing This step aims to calculate similarity values between each user and target users (central users) based on trust and comment information, extracting the top-n similar users. If a direct trust relationship exists between users (e.g., friendship, follow relationship), the trust value is calculated directly. If no direct trust relationship exists, hidden trust relationships are extracted, such as comment or rating information. If no direct trust relationship exists between user u and target user a, PCC is used to calculate the trust value between u and a based on comment or rating information. Network nodes represent users, and edge weights represent similarity between users.

Trust-based user similarity calculation [13] is:

$$sim(a, u) = \begin{cases} \frac{sim_T(a,u) \times T_{a,u}}{sim_T(a,u) + T_{a,u}} & \text{if } sim_T(a, u) \neq 0 \text{ and } T_{a,u} \neq 0 \\ sim_T(a, u) & \text{if } sim_T(a, u) \neq 0 \text{ and } T_{a,u} = 0 \\ T_{a,u} & \text{if } sim_T(a, u) = 0 \text{ and } T_{a,u} \neq 0 \\ 0 & \text{if } sim_T(a, u) = 0 \text{ and } T_{a,u} = 0 \end{cases}$$

where T , is the trust value between target user a and user u, calculated as:

$$T_{a,u} = 1 - \frac{d_{a,u}}{d_{max}}$$

where d , represents the trust propagation distance between a and u, and d is the maximum trust propagation distance, set as the average path length in the graph.

3.3.2 Two-Dimensional Graph Model Weighting ACO is used to process top-n users and analyze their importance. First, a two-dimensional user graph is constructed; then, the ant colony traverses the graph to adjust the similarity between each user and target users.

1) **Constructing User Two-Dimensional Graph**: First, select top-n users similar to the target user. Then, construct the two-dimensional graph from Section 1, where nodes represent users and edges with weights represent similarity between users (calculated by Equation (7)), with weights in [0,1]. Figure 2 Figure 2: see original paper shows an example of a social network' s two-dimensional graph; (b) shows the sub-graph of target user 3. Heuristic information and expected information are the two main elements of ACO, with heuristic information defined as the inverse weight value.

2) **Ant Colony Traversal Strategy**: In the initialization phase, the ant colony is randomly placed in the graph, and pheromones are updated during traversal. Ants release appropriate pheromone amounts based on similarity between users and target users, traversing the graph using a routing table. The probability of ant k moving from node i to node j is defined as:

$$p_{ij}^{k} = \frac{\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}{\sum_{m\in\text{unvisited}}\tau_{im}^{\alpha}\eta_{im}^{\beta}}$$

where $\mathcal{N}_i$ is the neighbor set of node i; is the pheromone amount; is the heuristic value; and are parameters controlling the weights of and respectively; = 1/sim(u , u); and m represents unvisited users. This probability function prevents the algorithm from falling into local optima. In social networks, this function can select user sets with high similarity to target users and low redundancy.

3) **Pheromone Update Method**: In ACO, pheromones reflect ants' experiences in solving a problem, and pheromone updates reflect solution quality. In this work, node pheromones represent the relevance between users and target users. The pheromone update method for user u is:

$$\tau_i = \tau_i + \sum_{k=1}^{N_{ant}} \Delta\tau_i^k$$

where is the pheromone of user u and Δ is the pheromone amount released by ant k at user i. Δ reflects solution quality, calculated as:

$$\Delta\tau_i^k = \begin{cases} \frac{Q}{\text{cost}(U_k)} & \text{if } i \in U_k \\ 0 & \text{if } i \notin U_k \end{cases}$$

where Q is a constant; U is the set of users traversed by ant k; and cost(U) is the quality of the solution found by ant k. After each iteration, all node pheromones are updated:

$$\tau_i = (1 - \rho)\tau_i$$

where is the pheromone evaporation rate. Solution quality is calculated using the average error metric, with each solution consisting of a user set and its weights.

3.4 Overall Algorithm Design

In the initialization phase, each node' s pheromone is set to constant c, and the ant colony is randomly distributed across graph nodes. Each ant traverses the graph based on Equation (11), possibly selecting different-sized user sets. Ants update user pheromones according to Equation (12). Considering pheromone evaporation, global pheromone updates are performed using Equation (14) at the end of each iteration. These steps repeat until termination conditions are met. Users are then sorted in descending order of importance, and top-k users are selected as the final subset.

Algorithm 2 shows the GC-ACO algorithm pseudocode. Input variables R, T, m, N , NI represent rating information, trust information, number of users, target user, ant colony size, and iteration count, respectively. The algorithm steps are: a) Calculate similarity between target user and other users, selecting those with similarity above threshold for ACO processing; b) Use ACO to assign weights to users—during each ACO iteration, ants traverse the graph to select a similar user set for the target user, outputting a user set with pheromone values; c) Improve clustering effectiveness and coverage for low-activity users through a prediction procedure.

Algorithm 2: Social Network Clustering Algorithm Based on Two-Dimensional Graph and ACO
Input: R, T, m, N , NI
Output: User set similar to target user

/* Preliminary user screening based on rating and trust information */
1. Calculate similarity between target user a and other users // Equation (1)
2. SU = select user set with similarity above threshold

/* Calculate user weights using ACO */
3. Construct user two-dimensional graph
4. Initialize pheromones for graph nodes
5. for i = 1 to NI {
6. Randomly distribute ant colony
7. for j = 1 to N {
8. while (heuristic value of current user is below threshold) {
9. Ant selects next user // Equation (11)
10. }
11. Update pheromones // Equation (12)
12. }
13. Update global pheromones // Equation (14)
14. }

/* Prediction for low-activity users */
15. Sort users in descending order by pheromone
16. Select top-k users
17. For users lacking comment information, predict their ratings for target users based on their most similar users' comments. The prediction method is:

$$\hat{r}_{u,i} = \frac{\sum_{v \in U} w_v \cdot r_{v,i}}{\sum_{v \in U} w_v}$$

where $\hat{r}_{u,i}$ is the predicted rating of target user u for item i; U is the user set selected by ant k; r , is the true rating of v for item i; and w is the pheromone of v. Each solution' s cost is calculated as the error between predicted and true values.

3.5 Complexity Analysis of GC-ACO Algorithm

In step a) of Algorithm 2, since similarity between each user pair depends on total user count, the computational complexity is $O(n^2)$. The second line selects users with similarity above , let $l = |SU|$, constructing a two-dimensional graph with l users. Step b) has NI iterations with complexity $O(NI \cdot N \cdot l^2)$, which can be reduced to $O(NI \cdot l^2)$ with distributed processing. Step c) has complexity $O(l \log l)$. Overall, the algorithm' s total complexity is $O(n^2 + NI \cdot l^2 + l \log l)$.

---

4 Experiments and Results Analysis

Recommender systems are an important application scenario for social network clustering technology. Following the experimental protocols in [5,14], we com-

bine clustering technology with collaborative filtering recommender systems, evaluating social network recommendation effectiveness through recommender system performance. We test the GC-ACO algorithm' s clustering performance on three datasets. The experimental environment is a PC with 8 GB RAM and an i7 8700 processor. Using five-fold cross-validation, each dataset is divided into five subsets, with four randomly selected as training sets and the remaining one as test set in each iteration.

4.1 Performance Evaluation Metrics

We employ three classic recommender system performance metrics: Mean Absolute Error (MAE), Root Mean Square Error (RMSE), and coverage rate (RC). MAE evaluates prediction accuracy by calculating differences between predicted and true rating values:

$$MAE = \frac{1}{Z} \sum_{(u,j)} |r_{uj} - \hat{r}_{uj}|$$

where Z, $\hat{r}_{uj}$, and r are the number of ratings by user u for item j, estimated ratings, and true ratings, respectively. RMSE also evaluates recommender system performance, measuring absolute error between predicted and true ratings:

$$RMSE = \sqrt{\frac{1}{Z} \sum_{(u,j)} (r_{uj} - \hat{r}_{uj})^2}$$

RC evaluates recommender system performance from another perspective, assessing the system' s ability to mine long-tail items:

$$RC = \frac{\text{Number of predicted ratings}}{\text{Total number of ratings}}$$

4.2 Experimental Datasets

We use three benchmark datasets: FilmTrust, Epinions, and Ciao. FilmTrust is a real-world dataset from a movie recommendation website where users comment on and rate movies, add friends, and share opinions. FilmTrust ratings are real numbers in [0.5, 4]. Epinions includes multiple social relationships, item comments and ratings, and user trust relationships, with integer ratings in [1,5]; trust relationships are binary ("1" for trust, "0" for distrust). Ciao ratings are integers in [1,5]. Table 1 shows information for the three benchmark datasets.

To test our algorithm' s effectiveness on sparsity and cold-start problems, we further divide datasets based on two conditions: a) Cold-start users—extract users with fewer than 5 ratings; b) Sparse items—extract items with fewer than 5 ratings; c) Complete user set. Table 2 shows information for the divided sub-datasets.

4.3 Parameter Settings

Through multiple preprocessing experiments, we selected optimal parameter configurations: maximum iterations = 70, initial pheromone = 0.02, pheromone evaporation coefficient = 0.2. Parameters , , Ω are set to =0.6, =0.4, Ω=1.66, with user neighbor count = 2~30. The number of ants equals the number of users in each dataset.

4.4 Experimental Results

We select two recent social network-based recommender systems and one intelligent optimization-based recommender system for comparison: a) TrustSVD [15]—a trust and rating-based recommender system; b) TrustMF [16]—a trust and matrix factorization-based recommender system; c) Yilmaz [17]—a genetic algorithm-based recommender system.

4.5 Recommendation Performance on Different Datasets

We conduct recommendation experiments on cold-start, sparse, and complete datasets using TrustSVD, TrustMF, Yilmaz, and GC-ACO, recording MAE and RMSE results. Tables 3-5 show results for FilmTrust, Epinions, and Ciao datasets, respectively. GC-ACO achieves better accuracy on FilmTrust and Epinions datasets, outperforming the other three systems. For Ciao, it also achieves good results, though its accuracy on the complete dataset is slightly lower than TrustMF, and on the sparse dataset slightly lower than TrustSVD. Overall, GC-ACO demonstrates good recommendation effectiveness and effectively alleviates cold-start and sparsity problems.

Coverage is an important metric for recommender systems and social networks. Figure 3 [Figure 3: see original paper] shows coverage results for the four systems. All four achieve high coverage, with TrustMF, Yilmaz, and GC-ACO all above 0.9, while our algorithm' s coverage is slightly higher than TrustMF and Yilmaz.

4.6 Convergence Analysis

TrustSVD and TrustMF are not iterative algorithms. Yilmaz is based on genetic algorithms. We compare our algorithm' s convergence with Yilmaz, as convergence is key for iterative algorithms. Figure 4 Figure 4: see original paper-(c) show convergence curves for FilmTrust, Epinions, and Ciao datasets. Our algorithm demonstrates better convergence speed and accuracy than Yilmaz. With strong global search capability, it achieves good accuracy, and with strong local exploitation capability, it achieves fast convergence speed. Equation (11)' s probability function prevents ACO from falling into local optima, enabling strong global search capability. Without using a greedy mechanism, it allows low-probability users to remain selectable. Additionally, preliminary user screening maintains high exploitation capability and reduces solution space. Our algorithm uses rich direct and indirect trust relationships to establish graph

weights, enabling ants to traverse the graph quickly and efficiently in early iterations, thus achieving good exploitation capability and convergence speed.

5 Conclusion

Addressing the directional and diverse nature of social relationships in social networks, this paper proposes a social network clustering algorithm based on graph clustering and ant colony optimization. A two-dimensional graph constructed under coverage constraints ensures a balance between coverage and clustering accuracy. The graph construction process incorporates diverse information including direct trust relationships, trust propagation, and comment information. The algorithm achieves good recommendation effectiveness and alleviates cold-start and sparsity problems. Using rich direct and indirect trust relationships to establish graph weights enables ants to traverse the graph quickly and efficiently in early iterations, resulting in good exploitation capability and convergence speed.

Future work will consider introducing more hidden social information and external information to enhance social network judgment, such as user profiles, comment context, and behavioral trajectories.

References

[1] Hu Changjun, Xu Wenwen, Hu Ying, et al. Review of information diffusion in online social networks [J]. Journal of Electronics & Information Technology, 2017, 39(4): 794-804.

[2] Zhao Shu, Liu Xiaoman, Duan Zhen, et al. A survey on social ties mining [J]. Chinese Journal of Computers, 2017, 40(3): 535-555.

[3] Stovall T R, Kockara S, Avci R. GPUSCAN: GPU-based parallel structural clustering algorithm for networks [J]. IEEE Trans on Parallel & Distributed Systems, 2015, 26(12): 3381-3393.

[4] Zhao Weizhong, Chen Gang, Xu Xiaowei. AnySCAN: an efficient anytime framework with active learning for large-scale network clustering [C]// Proc of IEEE International Conference on Data Mining. Washington DC: IEEE Computer Society, 2017: 665-674.

[5] Tang Ying, Zhong Nanjiang, Sun Kanggao, et al. Clustering and visualization of social network based on user interests [J]. Computer Science, 2017, 44(b11): 385-390.

[6] Chen Jimeng, Chen Jiajun, Liu Jie, et al. Clustering algorithms for large-scale social networks based on structural similarity [J]. Journal of Electronics & Information Technology, 2015, 37(2): 449-454.

[7] Cai Qing, Gong Maoguo, Ma Lijia, et al. Greedy discrete particle swarm optimization for large-scale social network clustering [J]. Information Sciences An International Journal, 2015, 316(9): 503-516.

[8] Wu Jian, Chiclana Francisco, Fujita Hamido, et al. A visual interaction consensus model for social network group decision making with trust propagation [J]. Knowledge-Based Systems, 2017, 122(C): 39-50.

[9] Meng Xiangwu, Liu Shudong, Zhang Yujie, et al. Research on social recommender systems [J]. Journal of Software, 2015, 26(6): 1356-1372.

[10] Liu Hongjie, Lu Hao, Zhang Nan, et al. Theory research based on microblog of six degrees space [J]. Application Research of Computers, 2012, 29(8): 2826-2829.

[11] Eaton Jayne, Yang Shengxiang, Mavrovouniotis M. Ant colony optimization with immigrants schemes for the dynamic railway junction rescheduling problem with multiple delays [J]. Soft Computing, 2016, 20(8): 2951-2966.

[12] Han Xiao, Liu Shufen, Xu Tianqi. Improved K-medoids algorithm based on genetic simulated annealing algorithm [J]. Journal of Jilin University: Engineering and Technology Edition, 2015, 45(2): 619-623.

[13] Moradi P, Ahmadian S. A reliability-based recommendation method to improve trust-aware recommender systems [J]. Expert Systems with Applications, 2015, 42(21): 7386-7398.

[14] Chen Kehan, Han Panpan, Wu Jian. User clustering based social network recommendation [J]. Chinese Journal of Computers, 2013, 36(2): 349-359.

[15] Guo Ging, Zhang Jie, Yorke-Smith Neil. TrustSVD: collaborative filtering with both the explicit and implicit influence of user trust and of item ratings [C]// Proc of the 29th AAAI Conference on Artificial Intelligence. Palo Alto: AAAI Press, 2015: 123-129.

[16] Yang Bo, Lei Yu, Liu Jiming, et al. Social collaborative filtering by trust [J]. IEEE Trans on Pattern Analysis & Machine Intelligence, 2017, 39(8): 1633-1647.

[17] Ar Y, Bostanci E. A genetic algorithm solution to the collaborative filtering problem [J]. Expert Systems with Applications, 2016, 61(1): 122-128.

Note: Figure translations are in progress. See original paper for figures.