applied sciences

MDPI

Article

Improving Ant Collaborative Filtering on Sparsity via Dimension Reduction

Xiaofeng Liao [1,2,3,*], Xiangjun Li [4,*], Qingyong Xu [5], Hu Wu [6] and Yongji Wang [6]

1 School of Management, Nanchang University, Nanchang 330031, China
2 Multiscale Networked System, Informatics Institute, University of Amsterdam, 1098 XH Amsterdam, The Netherlands
3 Postdoctoral Research Station of Management Science and Engineering, Nanchang University, Nanchang 330031, China
4 School of Software, Nanchang University, Nanchang 330031, China
5 School of Tourism, Nanchang University, Nanchang 330031, China; xyongle@163.com
6 State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China; wuhuieeeaccess@gmail.com (H.W.); ywang@itechs.iscas.ac.cn (Y.W.)
* Correspondence: xfliao@ncu.edu.cn or x.liao@uva.nl (X.L.); lixiangjun@ncu.edu.cn (X.L.)

Received: 23 August 2020; Accepted: 12 October 2020; Published: 16 October 2020

Abstract: Recommender systems should be able to handle highly sparse training data that continues to change over time. Among the many solutions, Ant Colony Optimization, as a kind of optimization algorithm modeled on the actions of an ant colony, enjoys the favorable characteristic of being optimal, which has not been easily achieved by other kinds of algorithms. A recent work adopting genetic optimization proposes a collaborative filtering scheme: Ant Collaborative Filtering (ACF), which models the pheromone of ants for a recommender system in two ways: (1) use the pheromone exchange to model the ratings given by users with respect to items; (2) use the evaporation of existing pheromone to model the evolution of users' preference change over time. This mechanism helps to identify the users and the items most related, even in the case of sparsity, and can capture the drift of user preferences over time. However, it reveals that many users share the same preference over items, which means it is not necessary to initialize each user with a unique type of pheromone, as was done with the ACF. Regarding the sparsity problem, this work takes one step further to improve the Ant Collaborative Filtering's performance by adding a clustering step in the initialization phase to reduce the dimension of the rate matrix, which leads to the results that $K << \#users$, where K is the number of clusters, which stands for the maximum number of types of pheromone carried by all users. We call this revised version the Improved Ant Collaborative Filtering (IACF). Experiments are conducted on larger datasets, compared with the previous work, based on three typical recommender systems: (1) movie recommendations, (2) music recommendations, and (3) book recommendations. For movie recommendation, a larger dataset, MoviesLens 10M, was used, instead of MoviesLens 1M. For book recommendation and music recommendation, we used a new dataset that has a much larger size of samples from Douban and NetEase. The results illustrate that our IACF algorithm can better deal with practical recommendation scenarios that handle sparse dataset.

Keywords: recommender systems; collaborative filtering; ant colony optimization; ant collaborative filtering; dimension reduction

1. Introduction

Recommender systems allow the user to handle the increasing online information overload problem by providing personalized suggestions based on the user's history and interest. The class of recommendation algorithms can be generally divided into two main categories: content-based and collaborative filtering

(CF) recommendations. Although regarded as one of the most successful recommendation methods, collaborative filtering still faces two big challenges: sparsity and time evolvement.

Among many collaborative filtering algorithms targeting these two problems, the solution based on Genetic Algorithms enjoys favorable characteristics that are difficult to achieve in other solutions. In a recent work [1], an ant collaborative filtering scheme was developed by adopting Ant Colony Optimization. Their method was able to identify most related users and items in the sparse rate matrix by utilizing the pheromone exchange between users and items, and the pheromone evaporation over time. However, the fact reveals that many users may share the same or similar preference towards items, which indicates that it is not necessary to assign a unique type of pheromone to one single user as the ACF in [1] did. To reflect this fact, this work takes one step further by including an additional clustering step for dimension reduction in the initialization phase, which leads to a much lower number of clusters (pheromones) compared with the number of users, i.e., $K << \#users$. With the reduced number of pheromones required, the improved algorithm can improve performance in the case of sparseness.

The remainder of this paper is structured as follows. Section 2 investigates the dimension reduction work in general settings and in swarm intelligence recommendation settings. Section 3 details the design and algorithm of the improved solution. Section 4 presents and analyzes the experiment results. Section 5 concludes the work and discusses specific issues, as well as future works.

2. Related Work

We first examine how some conventional dimension reduction techniques are generally used in collaborative filtering recommendation. Next, we mainly focus on the utilizing of dimension reduction upon recommendations adapting genetic algorithms.

2.1. Dimension Reduction in General Settings

Dimension reduction on an original rating matrix is usually used to solve the sparsity problem. The underlying reason for dimension reduction is that the factors that govern user choices are less significant than the dimension of the rating data.

The dimension reduction methods used in model-based recommendations mainly consist of three types [2]: the probabilistic methods [3], the matrix factorization methods [4,5], and the clustering method [6,7].

The first type of method assumes the presence of some latent variables, to which both the users and the items belong. The objective of this type of method is to learn the probability-based similarity between users and items that belong to the same, or a similar, latent variable. A typical representative of this type of method is the PLSA model [8]. An example work is found in [9], which utilizes probabilistic methods for collaborative filtering recommendations. The benefit of adopting a probabilistic framework is that all items and users can be explicitly and simultaneously represented within the framework.

The second type of method, i.e., the matrix factorization method, mainly derives a matrix with restricted ranks from the original matrix, is often considered as a dimension reduction technique and is widely applied in collaborative filtering problems [10]. One typical example of this type of method is the Non-negative Matrix Factorization (NMF) [11].

Another example model in this type of method is the Singular Value Decomposition (SVD). In [12], SVD was used to reduce dimensions to improve performance for recommender systems. In [13], SVD was used to reduce dimensions and find the most similar users and items in each cluster. The work in [14] is another example, which applies Latent Semantic Indexing (LSI) to construct a user profile based on both collaborative and content features.

The third type of method, i.e., the clustering-based method, adopts clustering methods in collaborative filtering. An example is the work in [15], where the authors adopt a clustering procedure on the user profile in the collaborative filtering implementation.

Another example is the work in [16], which uses clustering techniques for dimension reduction, where the k-means algorithm and the SVD method are combined to cluster similar users.

2.2. Dimension Reduction in Swarm Intelligence Recommendation Settings

Particle swarm optimization is broadly regarded as an effective solution for NP-complete global optimization problems because, due to their nature-inspired spirit, they can provide good suboptimal solutions at a reasonable cost [17]. The following are some of the most recent works integrating particle swarm optimization with dimension reduction techniques such as clustering for various applications.

Similar to our work, the work in [18] is a combination of K-Means clustering and particle swarm optimization. The K-means clustering method is used to cluster all the influential factors into several disjointed sub-regions with simpler patterns. After that, a particle swarm intelligence method is applied for their ELM optimization task. The authors of [19] also present a combination of the clustering method, where the spectral clustering method is used with swarm optimization for text document clustering.

Another example of combining particle swarm optimization with the hybrid clustering method is found in [20], which enjoys the benefit of the global convergence ability by particle swarm optimization and the local exploitation of hard clustering algorithms. An employment of the particle swarm optimization algorithm for the task of clustering time-series data is reported in [21].

To the best of our knowledge, although, in a broader sense, several particle swarm optimization algorithms have been applied to recommendation tasks, few works integrate dimension reduction techniques and ant colony optimization for intelligence recommendation systems.

A closely related example is the work in [22], where they use a genetic algorithm to condense the matrix, and this assists in searching for relevant users and items. The idea is that the reduced user-item matrix may eliminate the sparsity problem via increasing the likelihood that different users rate common items.

3. Improve Ant Collaborative Filtering with Dimension Reduction

Although the ACF algorithm in [1] can solve the sparsity problem with the help of a pheromone scheme, it can still be further improved as it reveals that it is not necessary to allocate for each user a unique type of pheromone, as some users sharing a similar preference could be allocated a same type of pheromone. This section takes one step further based on the work in [1], to solve the sparsity problem. More specifically, an additional dimension reduction step is integrated into the initial phase to improve the ACF algorithm's performance when facing extreme sparseness.

3.1. Ant Collaborative Filtering

Before providing detail of the improved ACF, we explain the basic ACF algorithm for readers to become acquainted the background idea. For more details, the readers are suggested to refer to the work in [1].

In a conventional environment of Ant Colony Optimization, each ant brings different types of signals representing preferences of food items, which are called pheromones. The pheromone is subject to exchange between users and items, and vice versa. The pheromone is also a feature of evaporation, which is suited for modeling the evolution of a user's preferences over time. This scheme of pheromones can be regarded as a communication medium in collaborative filtering.

The intuition behind the ACF algorithm is that one type of pheromone denotes the preference a user has with respect to an item. When a rating is given, the user's pheromone is exchanged with the pheromone of the item. Conversely, the item's pheromone can also be transferred to the user when a rating is given. After a rating period, users with similar preferences become close in regard to the pheromone pattern they have. Similarly, the items with similar pheromones will also share similar pheromones.

3.2. Dimension Reduction Version for ACF

In the ACF algorithm, a unique type of pheromone is initially allocated to each user. However, in practice, it turns out that a lower number of pheromones types are needed, so it is more reasonable to initialize the users with similar preferences with the same type of pheromone. With this in mind, the types of pheromones can be reduced to alleviate the sparsity problem. We call the improved version as Improved Ant Collaborative Filtering (IACF).

First, the users are clustered according to the rating patterns indicated by their pheromones. The parameter K stands for the expected types of pheromones. The user clusters generated are used for pheromone initialization, and various clustering techniques can also be used [23]. Several clustering methods have been tested, including K-means and spectral clustering, and the results show that the typical K-means is more suitable for our dataset, as it finds balance between more variance and overfitting. Thus, the K-means is used in the initialization phase in the IACF algorithms.

The user pheromone is initialized by assigning each single user one type of pheromone valued $k \in [1, K]$, where $K << \#users$. For multiple users within one cluster, a same value of k is assigned to every one of them. This is the difference between the IACF and ACF. The rest of the procedure remains the same. The pheromone for each item is initialized as zero. If user u_i rates item v_j with rating $r_{i,j}$, the pheromones from both the user and the item are exchanged between them.

The complete update of the pheromone for a user and an item are depicted, respectively, in Equations (1) and (2). More specifically, a user will update its pheromone after a rating is given by adding the item's pheromone times by a weight mixed with a rating adjustment and a constant γ. The constant γ is used for controlling the spreading rate. Likewise, the item's pheromone will also be updated after a rating is given by adding the user's pheromone times by a weight, which is also a mix of rating adjustment and the constant γ. The rating adjustment is the gap between the rating r_{ij} and the average rating. The greater the gap is, the stronger the user's preference is with respect to that item. If the rating is much higher than the average, then the transmission will be a strong positive "plus". Conversely, a rating r_{ij} that is much lower than the average, which means that the user and the item are unalike, results in negative values for the pheromones.

$$Ph_{v_j}^{(t+1)} = Ph_{v_j}^{(t)} \times \exp\left(\frac{amount_{v_j,k} + \lambda}{Max_{k\in K}(amount_{v_j,k}) + \lambda} - 1\right) + (r_{i,j} - \overline{r}_{u_i}) \times \gamma \times Ph_{u_i}^{(t)} \tag{1}$$

$$Ph_{u_i}^{(t+1)} = Ph_{u_i}^{(t)} \times \exp\left(\frac{amount_{u_i,k} + \lambda}{Max_{k\in K}(amount_{u_i,k}) + \lambda} - 1\right) + (r_{i,j} - \overline{r}_{v_j}) \times \gamma \times Ph_{v_j}^{(t)} \tag{2}$$

The IACF algorithm is presented in Algorithm 1. The main difference between ACF and IACF is, at the pheromone initialization phase, where a clustering step is added, the training phase is kept unchanged with Algorithm 1 in [1].

Algorithm 1: Training procedure for IACF

```
Input: $Ratings, U, V, K$
Output: Updated pheromone value for users and items
// A leading clustering step
Using K-means to cluster users into $K$ clusters according to their preference indicated by their pheromones.
//Initialization
for $user \in U$ do
    // Assign each user one type of pheromone
    $\mathbf{Ph_{user}} = \{Ph_k :\in [1.0, k]\};$
end
for $item \in I$ do
    // For simplicity, assign each item with a pheromone of value 0
    $\mathbf{Ph_{item}} = \{\};$
end
//Training
for $rating \in Ratings$ do
    $u_i = rating[user];$
    $v_j = rating[item];$
    $value = rating[value];$
    // Update user pheromones
    for $Ph \in \mathbf{Ph_{u_i}^{(t)}}$ do
        //Update user pheromone with exchange and evaporation effect
        $Ph = Ph \times \exp(\frac{amount_{Ph}+\lambda}{Max_{Ph_{u_i}}+\lambda} - 1) + \gamma \times (value - \bar{r}_{v_j}) \times Ph_{v_j}^{(t)}$
        // Cut off
        if $abs(Ph) < \sigma$ then
            $Ph = 0;$
        end
        $\mathbf{Ph_{u_i}^{(t+1)}} \leftarrow Ph;$
    end
    for $Ph \in \mathbf{Ph_{v_j}^{(t)}}$ do
        // Update item pheromone with exchange and evaporation effect
        $Ph = Ph \times \exp(\frac{amount_{Ph}+\lambda}{Max_{Ph_{v_j}}+\lambda} - 1) + \gamma \times (value - \bar{r}_{u_i}) \times Ph_{u_i}^{(t)}$
        // Cut off
        if $abs(Ph) < \sigma$ then
            $Ph = 0;$
        end
        $\mathbf{Ph_{v_j}^{(t+1)}} \leftarrow Ph;$
    end
end
```

3.3. Complexity Analysis

For both IACF and ACF, the time complexities at the batched training phase are $O(K \times \#ratings)$, in which K is the number of clusters, which represents the maximum types of pheromones carried by all users and items. Typically, $K << \#users$. The complexity at online updating is only $O(K)$. Because $K << \#users$, the IACF is faster than ACF. The complexity for the clustering step at the initial phase is $O(n * K * I * d)$, where n is the number of users, which represents the number of types of pheromones; I is the number of iterations, which in our case is a constant; and d is the number of attributes, which in our case is 1.

In the recommendation phase, both ACF and IACF have a time complexity $O(\#users + \#items)$ for the rating-based recommendation, and a time complexity for the ranking-based recommendation, and they are both $O(\#items)$.

4. Experiments

Most collaborative filtering algorithms operate with the rating prediction manner, i.e., predicting what the user would rate the item and then recommend the items with the highest prediction scores. Another operation manner is to generate a top N list of items that the user is most likely interested in, where N is the length of the final recommendation list, which could typically have a value of 5, 10, 20, etc.

For comparison with the ACF algorithm, we take the same experiments settings with that in [1]. We conduct experiments on both scenarios: (1) Rating-based Recommendation and (2) Top-N Ranking.

4.1. Parameter Settings

The parameters include (1) the number of clusters, (2) the transmission rate γ, and (3) the evaporation rate λ.

The transmission rate γ controls the pheromones exchanging speed between a user and an item, and has a value of $\gamma \in [0,1]$. When $\gamma = 0$, no pheromone exchange occurs when a rating is given by a user to an item. A greater γ indicates faster exchanges of pheromones. $\gamma = 1$ indicates a full exchange of pheromones between the user and the item for each rating. For simplicity, in our experiments, γ is set to 1.

The evaporation speed of the pheromone of the user or the item is affected by the evaporation rate $\lambda \in [0,1]$. A greater λ indicates a slower evaporation of pheromones, which means weaker time effects on the user's preference towards an item. As the influence of pheromone evaporation in real ant systems is unclear, the λ is empirically set according to [24]. The optimal value is set at 0.2, as illustrated in Figure 1, where the value along the x-axis represents the evaporation rate, and the value along the vertical axis represents the mean Root Mean Square Error (RMSE) over 10 runs. The results indicate a high sensitivity regarding the evaporation rate.

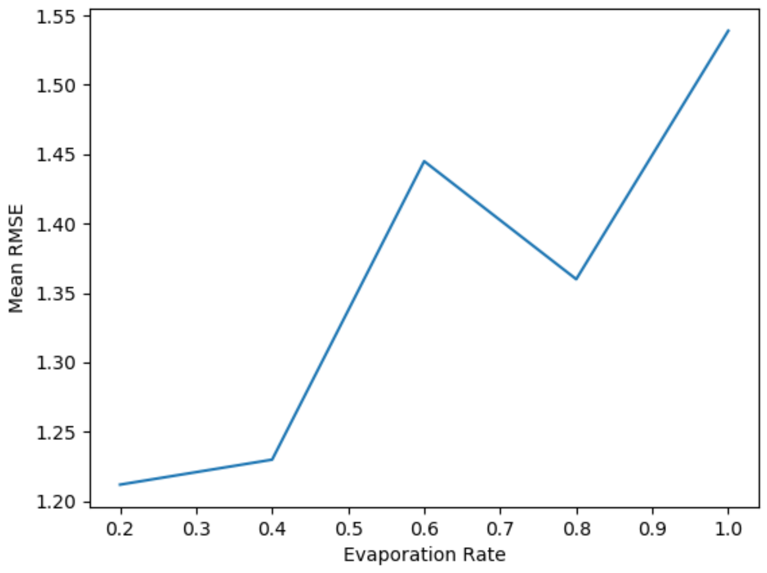

Figure 1. Empirical evaluation of Root Mean Square Error (RMSE) over different value of evaporation rate λ.

The optimal cluster number found is 20. The number of clusters is determined with the elbow method. A quick running of the Yellowbrick [25] implementation of the elbow method suggests that the number of clusters at 20 gives a balance between more variation and over-fitting.

4.2. Rating-Based Recommendation

To test the IACF's effectiveness regarding sparsity, we used a larger dataset Movies 10M (https://grouplens.org/datasets/movielens/10m/) than the Movies 1M used in [1]. MovieLens 10M contains 10 million ratings given to 10,000 movies from 72,000 users. The ratings are given on a five-star scale; 90% of the ratings are kept as a training set, and 10% are kept as a testing set.

The evaluation metric adopted is RMSE as defined below. Statistically, RMSE is the standard deviation of the residuals. It represents the square root of the second sample moment of the differences between predicted values and observed values, so it is extensively used in measuring the accuracy of prediction. Clearly, the smaller the value is, the better the prediction is.

$$RMSE = \sqrt{\frac{1}{n} \sum_{(u_i,v_j) \in TestSet} (r_{i,j} - estimated\_rating)^2}.$$

For this scenario, we only compare IACF directly with ACF to see the improvement brought about by the dimension reduction in the initialization phase. The comparison results in Table 1 indicate that the IACF runs much faster than the ACF with better RMSE points. IACF is faster because much fewer types of pheromones are assigned due to the clustering procedure in the initial phase. The users are clustered into K clusters according to their rating pattern, i.e., the desired number of types of pheromones. Typically, we have $k << \#users$. For the batched training phase, the time complexities for both ACF and IACF are $O(K \times \#ratings)$. For the online update, the update complexity is only $O(K)$. In the recommendation phase, for the rating-based recommendation, the time complexity is $O(\#users + \#items)$, but the computations could be significantly reduced if the user neighbors and item neighbors are explicitly maintained in memory.

Table 1. Rating-based recommendation results evaluation.

Algorithm	RMSE	Time (s)	*p*-Value
ACF	1.853 ± 0.02	1340.3	0.3316
IACF	1.012 ± 0.02	137.2	

Additionally, the statistical difference is examined. The average RMSE for ACF and IACF algorithms over 5 runs is 1.853, with a standard deviation of 4.32 and 1.012, respectively. Our alternative hypothesis is that IACF does not improve the performance regarding RMSE. A value of 0.05 is used for alpha, which means the results are significant if the *p*-value is below 0.05. The z-score and consequent *p*-value produced based on these measurements are −0.4353 and 0.3316, respectively. Based on the *p*-value, the null hypothesis can be rejected.

4.3. Ranking-Based Recommendation

For this scenario, experiments are evaluated on two practical applications: music and book recommendations. The dataset adopted is from the work in [26], which includes Douban book and NetEase music datasets. Douban (http://www.douban.com) is the largest book recommendation website in China. Netease music is a freemium music streaming service developed and owned by NetEase, Inc. The distribution of the training/testing sets are also set to 90% and 10%. Table 2 provides statistics regarding these two datasets.

Table 2. Dataset properties.

	Douban	NetEase
#User	4965	115,994
#Item	41,785	19,980
#Preference	958,425	2,399,639

We evaluated the methods in terms of precision, recall, and F1. The definitions are given below.

$$Precision = \frac{\#\ Hitting}{N}, Recall = \frac{\#\ Hitting}{\#\ Likes}, F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

Precision is a widely used metric in recommender systems. Precision is the percentage of relevant items out of those items selected by the query. The value of precision is $\in [0, 1]$, with a higher value indicating a more accurate prediction [27]. Recall is the percentage of relevant items that are actually retrieved over the total amount of relevant items. F1 is the harmonic mean of precision and recall taking both metrics into account. Several ranking-based CF methods, NBI [28], RSM [29], the BM25-Item algorithm [30], and the classic user-based and item-based CF algorithms are chosen for comparison. Similarly, the distribution of the dataset is 90% for training and 10% for testing. All the results were obtained by averaging five different runs. Table 3 shows the comparison. On the Douban dataset, both the ACF and IACF have better F1 than other methods. The IACF demonstrates better performance than the ACF in terms of precision and running time. On the NetEase dataset, though both the IACF and ACF are not the best in terms of F1, their F1 scores are only a bit lower than the F1 scores of other methods. Again IACF shows better performance over ACF in regard of precision and running time.

Similarly, the statistical difference is examined regarding the IACF and ACF algorithms. The alternative hypothesis is that IACF does not improve the performance in terms of RMSE. A value of 0.05 is chosen for alpha, which means the results are significant if the *p*-value is below 0.05. The standard deviations observed for the experiments that are run on Douban and NetEase datasets are 3.1 and 4.3, respectively. With these measurements and the precision in Table 3, the *p*-values calculated are 0.4962 and 0.4989. There is statistically significant evidence showing that the IACF is an improvement over the ACF.

Table 3. Ranking results comparison.

Dataset	Algorithm	Precision	Recall	F1	Time (s)	*p*-Value
Douban	User-based	0.45	0.67	0.538	289.3	
	Item-based	0.33	0.73	0.455	2212.6	
	NBI	0.09	0.84	0.163	5314.0	
	RSM	0.29	0.78	0.423	2350	
	BM25-Item	0.08	0.89	0.147	2010.0	
	ACF	0.57	0.61	0.589	913.1	0.4962
	IACF	0.70	0.50	0.583	290.4	
NetEase	User-based	0.50	0.71	0.587	901.4	
	Item-based	0.40	0.78	0.529	1024.3	
	NBI	0.03	0.90	0.058	3121.2	
	RSM	0.49	0.71	0.579	802.1	
	BM25-Item	0.28	0.81	0.416	3223.1	
	ACF	0.76	0.36	0.489	728.3	0.4989
	IACF	0.81	0.30	0.438	413.2	

5. Discussion and Conclusions

In this paper, we proposed an improvement to the ACF algorithm [1] to better deal with the sparsity problem by introducing an initial clustering phase on the users according to their preferences represented by pheromones, of which there are far fewer compared to the number of users. The experiments on several larger datasets, including Movielens 10M, Douban book, and NetEase music datasets, demonstrate its excellence over the ACF algorithm in both scenarios, i.e., the rating-based recommendation and the ranking-based recommendation.

As mentioned earlier, K-means was used for clustering implementation because it is more appropriate for our dataset. However, a typical issue for K-means is the determination of the parameter K, i.e., the number of clusters. We empirically determined the number of *K* in our experiments. This could be studied further.

Author Contributions: Conceptualization, X.L. (Xiaofeng Liao); Formal analysis, X.L. (Xiaofeng Liao); Funding acquisition, X.L. (Xiaofeng Liao); Investigation, X.L. (Xiaofeng Liao); Methodology, X.L. (Xiaofeng Liao); Resources, Q.X.; Software, H.W.; Supervision, Y.W.; Validation, X.L. (Xiangjun Li); Writing—original draft, X.L. (Xiaofeng Liao); Writing—review and editing, X.L. (Xiaofeng Liao). All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China grant number 61762062 and 61862042, Science and Technology Innovation Platform Project of Jiangxi Province under grant number 20181BCD40005, Major Discipline Academic and Technical Leader Training Plan Project of Jiangxi Province under grant number 20172BCB22030, Primary Research & Development Plan of Jiangxi Province under grant number 20192BBE50075 and 20181ACE50033, and Jiangxi Province Natural Science Foundation of China under grant number 20192BAB207019.

Acknowledgments: The authors thank the anonymous reviewers for their helpful comments.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Liao, X.; Hu, W.; Yongji, W. Ant Collaborative Filtering Addressing Sparsity and Temporal Effects. *IEEE Access* **2020**, *8*, 32783–32791. [CrossRef]
2. Chao, G.; Luo, Y.; Ding, W. Recent advances in supervised dimension reduction: A survey. *Mach. Learn. Knowl. Extr.* **2019**, *1*, 341–358. [CrossRef]
3. Wu, S.; Liu, J.; Liu, L. Modeling method of internet public information data mining based on probabilistic topic model. *J. Supercomput.* **2019**, *75*, 5882–5897. [CrossRef]
4. Liu, D.; Ye, X. A matrix factorization based dynamic granularity recommendation with three-way decisions. *Knowl.-Based Syst.* **2020**, *191*, 105243. [CrossRef]
5. Lei, C.; Dai, H.; Yu, Z.; Li, R. A service recommendation algorithm with the transfer learning based matrix factorization to improve cloud security. *Inf. Sci.* **2020**, *513*, 98–111. [CrossRef]
6. Sangaiah, A.K.; Fakhry, A.E.; Abdel-Basset, M.; El-henawy, I. Arabic text clustering using improved clustering algorithms with dimensionality reduction. *Clust. Comput.* **2019**, *22*, 4535–4549. [CrossRef]
7. Zhu, X.; Gan, J.; Lu, G.; Li, J.; Zhang, S. Spectral clustering via half-quadratic optimization. *World Wide Web* **2019**, *23*, 1969–1988. [CrossRef]
8. Hofmann, T. Latent Semantic Models for Collaborative Filtering. *ACM Trans. Inf. Syst.* **2004**, *22*, 89–115. [CrossRef]
9. Langseth, H.; Nielsen, T.D. A latent model for collaborative filtering. *Int. J. Approx. Reason.* **2012**, *53*, 447–466. [CrossRef]
10. Koren, Y. Collaborative Filtering with Temporal Dynamics. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '09), San Jose, CA, USA, 12–15 August 2009; Association for Computing Machinery: New York, NY, USA, 2009; pp. 447–456. [CrossRef]
11. Zhang, S.; Wang, W.; Ford, J.; Makedon, F. Learning from incomplete ratings using non-negative matrix factorization. In Proceedings of the 2006 SIAM International Conference on Data Mining, Bethesda, MD, USA, 20–22 April 2006; pp. 549–553.
12. Sarwar, B.; Karypis, G.; Konstan, J.; Riedl, J. *Application of Dimensionality Reduction in Recommender System—A Case Study*; Technical Report; Minnesota University, Department of Computer Science: Minneapolis, MN, USA, 2000.
13. Nilashi, M.; Ibrahim, O.; Bagherifard, K. A recommender system based on collaborative filtering using ontology and dimensionality reduction techniques. *Expert Syst. Appl.* **2018**, *92*, 507–520. [CrossRef]
14. Symeonidis, P. Content-based Dimensionality Reduction for Recommender Systems. In *Data Analysis, Machine Learning and Applications*; Preisach, C., Burkhardt, H., Schmidt-Thieme, L., Decker, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2008; pp. 619–626.

15. Son, N.T.; Dat, D.H.; Trung, N.Q.; Anh, B.N. Combination of Dimensionality Reduction and User Clustering for Collaborative-Filtering. In Proceedings of the 2017 International Conference on Computer Science and Artificial Intelligence (CSAI 2017), Jakarta, Indonesia, 5–7 December 2017; Association for Computing Machinery: New York, NY, USA, 2017; pp. 125–130. [CrossRef]
16. Zarzour, H.; Al-Sharif, Z.; Al-Ayyoub, M.; Jararweh, Y. A new collaborative filtering recommendation algorithm based on dimensionality reduction and clustering techniques. In Proceedings of the 2018 9th International Conference on Information and Communication Systems (ICICS), Irbid, Jordan, 3–5 April 2018; pp. 102–106.
17. Sarmah, D.K. A survey on the latest development of machine learning in genetic algorithm and particle swarm optimization. In *Optimization in Machine Learning and Applications*; Springer: Singapore, 2020; pp. 91–112.
18. Feng, Z.; Niu, W.; Zhang, R.; Wang, S.; Cheng, C. Operation rule derivation of hydropower reservoir by k-means clustering method and extreme learning machine based on particle swarm optimization. *J. Hydrol.* **2019**, *576*, 229–238. [CrossRef]
19. Janani, R.; Vijayarani, S. Text document clustering using Spectral Clustering algorithm with Particle Swarm Optimization. *Expert Syst. Appl.* **2019**, *134*, 192–200. [CrossRef]
20. Gusmão, R.P.; de A.T. de Carvalho, F. Clustering of multi-view relational data based on particle swarm optimization. *Expert Syst. Appl.* **2019**, *123*, 34–53. [CrossRef]
21. Kamalzadeh, H.; Ahmadi, A.; Mansour, S. Clustering time-series by a novel slope-based similarity measure considering particle swarm optimization. *Appl. Soft Comput.* **2020**, *96*, 106701. [CrossRef]
22. Kim, K.; Ahn, H. Collaborative Filtering with a User-Item Matrix Reduction Technique. *Int. J. Electron. Commer.* **2011**, *16*, 107–128. [CrossRef]
23. Rui Xu.; Wunsch, D. Survey of clustering algorithms. *IEEE Trans. Neural Netw.* **2005**, *16*, 645–678.
24. Mavrovouniotis, M.; Yang, S. Adapting the Pheromone Evaporation Rate in Dynamic Routing Problems. In *Applications of Evolutionary Computation*; Esparcia-Alcázar, A.I., Ed.; Springer: Berlin/Heidelberg, Germany, 2013; pp. 606–615.
25. Bengfort, B.; Bilbro, R. Yellowbrick: Visualizing the Scikit-Learn Model Selection Process. *J. Open Source Softw.* **2019**, *4*, 1075. [CrossRef]
26. Yang, N. *Douban Movie and NetEase Music Datasets and Model Code*; The President & Fellows of Harvard College: Cambridge, MA, USA, 2019; doi:10.7910/DVN/JGH1HA. [CrossRef]
27. Powers, D.M. Evaluation: From precision, recall and F-measure to ROC., informedness, markedness & correlation. *J. Mach. Learn. Technol.* **2011**, *2*, 37–63.
28. Zhou, T.; Ren, J.; Medo, M.; Zhang, Y.C. Bipartite network projection and personal recommendation. *Phys. Rev. E* **2007**, *76*, 046115. [CrossRef] [PubMed]
29. Han, P.; Xie, B.; Yang, F.; Shen, R.M. An Adaptive Spreading Activation Scheme for Performing More Effective Collaborative Recommendation. In *Database and Expert Systems Applications*; Andersen, K.V., Debenham, J., Wagner, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2005; pp. 95–104.
30. Wang, J.; Robertson, S.; de Vries, A.P.; Reinders, M.J. Probabilistic relevance ranking for collaborative filtering. *Inf. Retr.* **2008**, *11*, 477–497. [CrossRef]