Hindawi
Security and Communication Networks
Volume 2022, Article ID 1705527, 10 pages
https://doi.org/10.1155/2022/1705527

Research Article

Enhancing Personalized Recommendation by Transductive Support Vector Machine and Active Learning

Xibin Wang,[1,2,3] **Yunji Li**,[1] **Jing Chen**,[4] **and Jianfeng Yang**[1,2]

[1]*School of Data Science, Guizhou Institute of Technology, Guiyang 550003, Guizhou, China*
[2]*Special Key Laboratory of Artificial Intelligence and Intelligent Control of Guizhou Province, Guiyang 550003, Guizhou, China*
[3]*Key Laboratory of Electric Power Big Data of Guizhou Province, Guiyang 550003, Guizhou, China*
[4]*College of Information Engineering, Guizhou University of Traditional Chinese Medicine, Guiyang 550025, Guizhou, China*

Correspondence should be addressed to Xibin Wang; binxiwang@git.edu.cn

Received 26 November 2021; Revised 24 January 2022; Accepted 17 February 2022; Published 9 March 2022

Academic Editor: Chenquan Gan

As an important component of information service networks, personalized recommendation technology provides users with better options and enables them to obtain information anytime and anywhere. Collaborative filtering (CF) is a successful and widely used form of this technology. However, the traditional CF recommendation algorithm is ineffective in environments with frequent entry of new users and high levels of data sparsity. For new users in the system, few or no scores, labels, or other such information is available, leading to the user cold start problem. Simultaneously, data sparsity leads to the selection of unreasonable neighbors, which reduces the recommendation accuracy. In addition, the traditional CF recommendation algorithm ignores the inherent connections between users' preferences and their basic information (such as demographics). Users with similar demographic information are likely to have similar preferences, which can serve as a good basis for finding neighbors. To address the aforementioned problems, we propose a recommendation model that combines active learning (AL) and a semi-supervised transductive support vector machine (TSVM). To enable neighbors to be found quickly and accurately, similar users are clustered together on the basis of their basic information. Then, the TSVM-based classifier is trained on each cluster. To improve the quality of sample labeling and thus the classifier performance, an active learning method based on the distance strategy and a multiclassifier voting mechanism is implemented. Finally, the TSVM-based recommendation model is trained on the labeled samples. The extensive experiments conducted using a real data set from MovieLens demonstrate that the proposed model effectively alleviates the aforementioned cold start and data sparsity problems.

1. Introduction

Information technology and the Internet have developed rapidly, and numerous online forums and e-commerce, social, and consulting service platforms have been established, resulting in the availability of a huge amount of information. However, obtaining effective information can be challenging due to information overload. Recommendation systems can address such problems [1, 2], in which a preference model between users and items is developed by acquiring users' behavior characteristics or preferences, predicting their preferences for unknown or unselected items, and generating a recommendation list. A recommendation algorithm is central to this type of system as it determines the recommendation effect and quality. Current personalized recommendation algorithms include content-based, collaborative filtering (CF), model-based, and hybrid recommendation algorithms [3–6]. Of these, the CF recommendation algorithm is one of the most successful and widely used algorithms. It has the advantages of not relying on the feature information of the item and not being constrained by content analysis technology and thus represents a major development in theoretical and practical terms. However, it requires continuous improvement due to the limitations of cold start (many users with little or no historical data) and data sparsity (limited user-item rating information or few items rated by multiple users) [7, 8].

The user cold start problem occurs when there are few or no scores, labels, or other information about new users in the system. The recommendation function of the system is even more important for such new users. Reference [9] shows that the loyalty of users to the system depends on whether and when the system provides effective personalized services. Users will have a greater reliance on the system and thus improve user loyalty if the recommendation function can be implemented for new users as early as possible. Thus, solving this cold start problem is necessary to improve the quality and efficiency of recommendation systems.

A common solution to the problem of new users is to use non-personalized recommendations; however, a lack of personal information means that the system must accumulate a certain level of data before it can provide recommendations. Another solution is to use user registration information as the basis for recommendations, but the resulting recommendations are likely to be coarse as the information is often limited. A more effective solution entails the use of active inquiry-based models that acquire the required knowledge through communicating with users and therefore provide rapid and accurate results.

The goal of a recommendation system is to satisfy users by providing appropriate recommendations by learning the users' preferences from their operations, which can be achieved through active learning. Active learning should therefore be integrated into recommendation systems [10]. The notion of the highest predicted score in [11] involves predicting the scores of unlabeled items, and the item with the highest predicted score may be the user's favorite item. The notion of the lowest predicted score in [12] is similar, in which the item with the lowest predicted score is selected and the user identifies the least preferred items with a score. An active learning algorithm is proposed in [13] based on matrix decomposition, which selects the sample with the lowest predicted score for users to choose. Reference [14] uses the aspect model to predict the probability that a target user u belonging to an interest group z will give a score r to a specific item i, where the user is a combination of multiple interest groups.

Data sparsity has also become a major problem for recommendation systems as it results in unreasonable neighbors being selected by the target user, which reduces the accuracy of the recommendations [15]. This issue can be addressed using a clustering algorithm to improve the recommendation accuracy. For example, [16] proposes a novel and scalable CCCF method, which improves the performance of CF methods via user-item co-clustering. Users and items are clustered into several subgroups, and each includes a set of like-minded users and a set of items they share an interest in. A hybrid approach is proposed by [17], which combines a content-based approach with CF under a unified model called co-clustering with augmented matrices (CCAMs). This method is based on information-theoretic co-clustering but further considers augmented data matrices, including user profiles and item descriptions. Reference [4] proposes a novel recommendation model based on a time correlation coefficient and an improved K-means with cuckoo search. The clustering method can cluster similar users together for further quick and accurate recommendations. The novel method of [18] applies clustering algorithms to the latent vectors of users and items, which can capture the interests that are common to the clusters of users and of items in a latent space. Some scholars have used matrix factorization (MF) techniques and singular value decomposition to solve this problem. For example, [19] proposes neural variational matrix factorization, which is a novel deep generative model that incorporates side information (features) of both users and items to effectively capture the corresponding latent representations to generate more accurate CF recommendations. To alleviate the effects of data sparsity, [20] proposes a framework that involves two efficient matrix factorizations, a dynamic single-element-based CF integrating manifold regularization (DSMMF) and a dynamic single-element-based Tikhonov graph regularization nonnegative MF (DSTNMF). A novel imputation-based recommendation method is proposed by [21] to solve the problem of data sparsity in SVD-based methods.

Building upon these studies, we propose a new personalized recommendation model in which active learning is integrated with semi-supervised learning and apply the cluster analysis method to solve the cold start and data sparsity problems. Users with similar rating patterns generally have similar interest preferences in recommendation systems and can be classified together on the basis of their item ratings. Thus, the basic information of new users (e.g., registration and demographic information) can determine the user group they belong to, and the system can accordingly provide more accurate item recommendations, to some extent solving the cold start problem. Semi-supervised learning and active learning methods can then be used to label the users' item preferences, which not only alleviate the data sparsity problem but also solve the cold start problem.

Our study makes four main contributions to the literature. First, we use cluster analysis due to the sparsity of a new user's query list, through which we can classify new users and obtain more reliable preference information, thus solving the problem of cold start.

Second, the label information of user-item association data, i.e., rating information expressed by users about items, is scarce, so we use the semi-supervised transductive support vector machine (TSVM) as the benchmark classifier, together with an active learning strategy based on distance, to label the association data.

Third, the quality of data is considered in active learning and fewer but higher quality items are selected for inquiring users, which address the problems of cold start and data sparsity. Specific inquiry information such as item scores can also be selected in active learning, which can supplement the scarce data for interest-related aspects, thus helping to ensure that the interest model is comprehensive.

Finally, as active learning does not rely on any similarity between users or items, the recommendations are not limited to similar modules, which expands the choices presented to users along with their cognitive domains and better portrays user preferences.

2. Semi-Supervised Transductive Support Vector Machine

Joachims et al. propose a transductive support vector machine learning algorithm (TSVM), which is the same as the traditional SVM learning method for binary classification problems, especially suitable for small sample training sets [22, 23]. In the TSVM training process, the test data (unlabeled sample) set is also considered together, and then, the classification error of the test data set is minimized. In other words, TSVM tries to assign different labels to unlabeled samples and find the classification hyperplane with the largest interval on all samples.

The principle of TSVM algorithm is as follows [24]:

Given a set of independently and identically distributed labeled training samples:

$$\begin{aligned} &D_l = \{(x_1, y_1), \ldots, (x_i, y_i)\} \in R^n \times R, \\ &i = 1, \ldots, l, \\ &y_i = \{-1, +1\}. \end{aligned} \tag{1}$$

Another set of unlabeled samples from the same distribution is as follows:

$$D_u = \{x_{l+1}, \ldots, x_{l+u}\}. \tag{2}$$

The learning objective of TSVM is to predict the unlabeled samples in D_u and give the prediction labels, so that

$$\begin{aligned} &\min\left(y_1, \ldots, y_n, w, b, \xi_1, \ldots, \xi_l, \xi_{l+1}, \ldots, \xi_{l+u}\right) \\ &\frac{1}{2}\|w\|^2 + C_1 \sum_{i=1}^{l} \xi_i + C_2 \sum_{i=i+1}^{l+u} \xi_j \\ &s.t. : \forall_{i=1}^{l} : \; y_i\left[w \cdot x_i + b\right] \geq 1 - \xi_i; \xi_i \geq 0 \\ &\forall_{i=l+1}^{l+u} : \; y_j\left[w \cdot x_j + b\right] \geq 1 - \xi_j; \xi_j \geq 0, \end{aligned} \tag{3}$$

where (w, b) determines a classification hyperplane; $\xi_i (i = 1, 2, \cdots, l)$ is the slack variable of labeled samples; $\xi_j (i = l+1, l+2, \ldots, l+u)$ is the slack variable of unlabeled samples; C_1 and C_2 are the impact factors of labeled and unlabeled samples specified by the user; and $C_2\xi_j$ is the "influence term" of the unlabeled sample x_j in the objective function.

The training process of TSVM algorithm is as follows [25]:

Step 1: set parameters C_1 and C_2, train labeled samples by inductive learning, and get an initial classifier. Set the estimated number N of positive samples in unlabeled samples.

Step 2: use the initial classifier to calculate the value of the decision function for all unlabeled samples. Label the first N unlabeled samples with large value of decision function as positive samples, and label the remaining unlabeled samples as negative samples. Set C_{temp} as a temporary impact factor.

Step 3: retrain the SVM model on all the labeled samples. For the newly generated classifier, according to the principle of reducing the objective function (3) as much as possible, exchange the labels of each pair of samples until there are no samples that meet the exchange conditions; otherwise, repeat the process.

Step 4: increase the value of C_{temp} uniformly, and return to Step 3. When $C_{\text{temp}} \geq C_2$, terminate the algorithm and return the labels of all unlabeled samples.

3. Active Learning

Aiming at the shortcoming of supervised learning that a large number of labeled samples must be used to construct a learner, active learning is proposed. In particular, in the case of a very large amount of data, the cost of labeling each sample will be very high [26]. The goal of active learning is to obtain a higher classification accuracy rate when the training data are limited and then these samples are labeled, which can not only reduce the training cost but also improve the classification effect of the learner.

Generally, the process of active learning includes two steps: establishing a basic classifier and selecting appropriate samples, where the basic classifier is obtained using supervised learning algorithm to learn and train on the labeled sample set; the process of sample selection is to select the samples with the largest value from the unlabeled sample set based on a certain sample selection strategy, then label them by domain experts or users, and add the labeled samples into the training set. Repeating the above two steps can gradually improve the performance of the classifier until the termination condition is met.

According to the different problem scenarios and unlabeled sample selection ways, active learning strategies can be divided into three types: membership query synthesis, stream-based selective sampling, and pool-based sampling [27]. Their differences are shown in Figure 1.

In the active learning strategy, it is mainly to determine which unlabeled sample has the largest amount of information or the most uncertain to be inquiry, and this inquiry strategy is the focus of research. Reference [28] proposes a new semi-supervised learning framework, which combines the active learning of the Gaussian random field with harmonic function, and selects unlabeled samples based on the value of the energy function. In [29], a combination of active learning and semi-supervised learning is proposed for sequence labeling, which can greatly reduce the cost of manual labeling. It only labels unlabeled samples with high uncertainty, and other sequences and subsequences are automatically labeled.

4. Sample Labeling Method Based on Distance Measurement and a Multiclassifier Voting Decision

4.1. TSVM Algorithm Analysis. We identified the following shortcomings in the TSVM algorithm through our analysis:

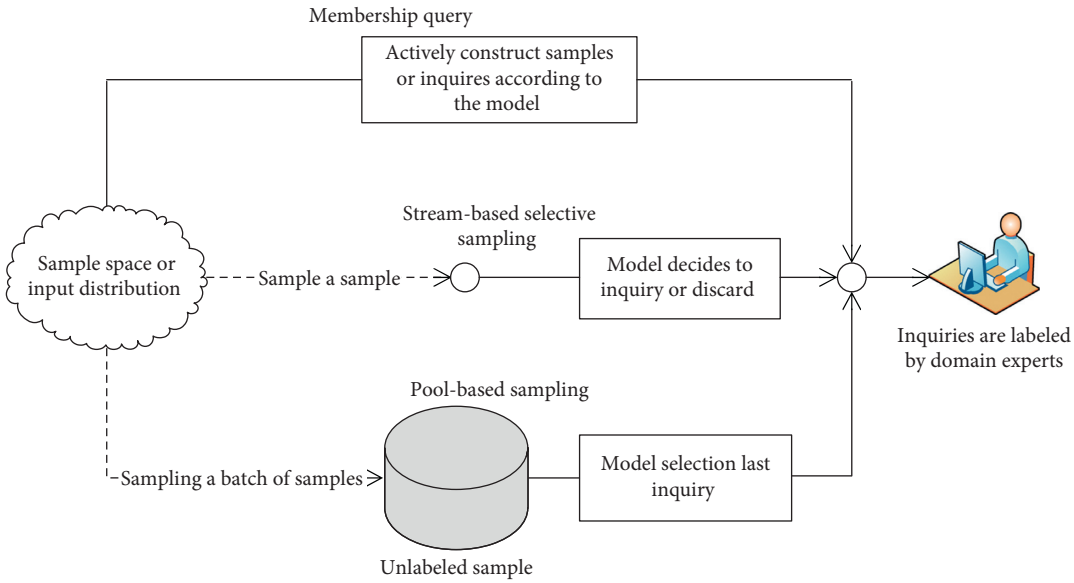

Figure 1: Three main active learning scenarios.

(1) If there are u unlabeled samples, then to achieve an accurate solution to the problem, all possible classification results for the sample sets must be searched for, that is, 2^u results, which belongs to a typical NP problem. If the unlabeled sample u is relatively small, the solution of this algorithm is completely possible. However, when the unlabeled sample u is large, it is almost impossible to find its exact solution, so it must be solved using labeled samples and various approximate optimization algorithms.

(2) The algorithm labels samples in pairs and selects those samples most likely to be support vectors in the boundary area for labeling each time, and the labeling speed is very slow.

Thus, to solve the above problems, we propose an active learning method that combines multiclassifier voting and collaborative training when labeling samples. The advantages of this method are as follows:

(1) The algorithm does not simply rely on the classification results of a single classifier to determine the samples to be labeled but trains multiple classifiers and uses the voting results of all of the classifiers to determine the samples to be labeled, which can improve the accuracy of labeling.

(2) When training multiple classifiers, the training set must be divided into multiple sub-training sets, and how these are divided determines the performance of the trained classifier. We use the clustering algorithm to divide the training set, which considers all of the geometric and spatial distribution characteristics of the labeled samples. A proportion of samples are then extracted according to the clustering results and the size of each cluster to construct a new training set and train the classifier, which can effectively improve its performance.

(3) In every iteration, each training set is small, so the training time cost of each classifier is relatively low.

(4) We train the obtained labeled samples and the previously labeled samples to obtain the final classifier, which improves not only the training speed of the classifier but also its performance.

4.2. Sample Labeling Based on Distance Strategy and Multiclassifier Voting. We apply the multiclassifier collaborative voting mechanism to label unlabeled samples, which improves the training speed of TSVM and the labeling accuracy of samples. The time complexity of the iterative training is reduced, and multiple classifiers use a voting mechanism to determine the class label of samples, thus improving the labeling accuracy in each iteration.

Figure 2 illustrates the multiclassifier voting decision labeling process. We divide the entire sample set into the labeled sample set L and an unlabeled sample set U. First, we cluster the labeled sample set and extract a specific number of samples from each cluster according to a specific proportion to form k (k is odd and greater than 1) subsample sets as the training sets, which guarantees that each training set is different. Second, k initial classifiers $C_1, C_2, \ldots, C_k$ are trained based on k training sets, following which we use these k classifiers to predict each unlabeled sample and obtain the output $f_1, f_2, \ldots, f_k$. Third, we label unlabeled samples and decide whether to iterate further based on the termination conditions. Four key problems must be solved in this process:

(1) Using the clustering algorithm to distinguish the training samples

(2) Selecting the samples to be labeled

(3) Adding the labeled samples to the corresponding classifier and using them in further iterative training

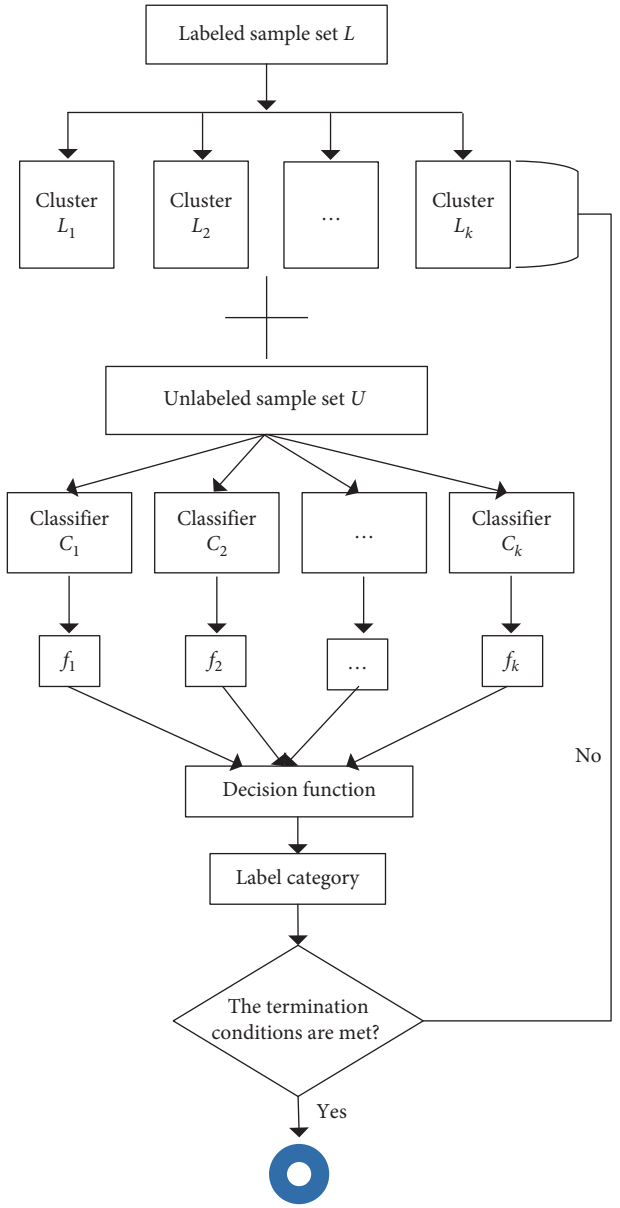

Figure 2: Framework of multiclassifier voting decision labeling.

(4) Determining the iteration termination conditions

4.2.1. Training Set Division Based on the Clustering Algorithm. The basic principles of training set division are as follows: first, the sample distribution in each training set and the differences between the training sample sets ensure that the output result of each set is reasonable; second, the difference between the training sets ensures the reliability of the voting results of multiple classifiers. The clustering method is used to construct the training set to ensure that the differences among the initial classifiers are as large as possible. k clusters are formed through cluster analysis, which is as similar as possible within the cluster and as dissimilar as possible between clusters. Samples are then randomly selected from each cluster in a specific proportion to form each training set.

We use the K-means clustering algorithm to cluster the initial training set. The specific steps are as follows:

Step 1: suppose the labeled sample set is $L = \{x_1, x_2, \ldots, x_l\}$, the number of clusters is K, the iteration round is r, and the initial value is 0; then, the initial K cluster centers are set as $(C_1^r, C_2^r, \ldots, C_K^r)$.

Step 2: assume that the set corresponding to the i-class sample is L_i^r. For any sample $x_j, j = 1, \ldots, l$, if the distance between x_j and the cluster center C_j^r is the shortest, as shown in equation (4), then the sample x_j is added to the L_j^r class.

$$\left\|x_j - C_j^r\right\| \le \left\|x_j - C_k^r\right\|, \quad k = 1, 2, \ldots, K. \tag{4}$$

Step 3: recalculate K cluster centers as follows:

$$C_i^r = \frac{1}{o_i} \sum_{x_j \in L_i^r} x_j, \quad i = 1, 2, \ldots, K, \tag{5}$$

where $o_i \in L_i^r$.

Step 4: define the clustering criterion function and calculate the clustering error as follows:

$$E(t) = \sum_{i=1}^{K} \sum_{x_j \in L_i^r} \left\|x_j - C_i^r\right\|. \tag{6}$$

Step 5: determine whether the stop condition is met. If $|E(t-1) - E(t)|$ is less than the preset error value, the final clusters and cluster centers are $L_i = L_i^r$, $C_i = C_i^r, i = 1, 2, \ldots, K$; otherwise, $r = r + 1$ is set, and go to Step 2.

4.2.2. Sample Labeling Strategy Based on Distance Strategy and the Multiclassifier Voting Mechanism. The size of the data set of the training samples is considered when selecting the samples to be labeled. Common labeling methods include the boundary-based sample method (e.g., pairwise sample labeling in TSVM), which has high labeling accuracy but very low labeling speed, and the region-based labeling method, which can simultaneously label multiple samples and thus has a high labeling speed but may have low labeling accuracy. We propose a multiclassifier voting decision labeling method that selects samples for labeling that belong to more than m classifier boundary regions at the same time; i.e., the minority is subordinate to the majority. If m samples meet equation (8), they are labeled as positive, and if m samples meet equation (9), they are labeled as negative.

Assuming that the classification hyperplane is $f(x)$ and the unlabeled sample set is $U = \{x_{l+1}, x_{l+2}, \ldots, x_{l+u}\}$, the distance from the sample x_i to the classification hyperplane is expressed as follows:

$$d(x_i) = \frac{\left|\omega^T x_i + b\right|}{\|\omega\|} = \frac{y\left(\omega^T x_i + b\right)}{\|\omega\|} = \frac{y f(x_i)}{\|\omega\|}, \quad i = l+1, \ldots, l+u. \tag{7}$$

We select unlabeled samples that are most likely to be support vectors for labeling and increase the labeling speed using a compromise between the pairwise labeling and region labeling methods to label the unlabeled samples. In each iteration process, unlabeled samples meeting the first p maximum of equation (8) are selected and labeled as positive, while those meeting the first q minimum values of equation (9) are then selected and labeled as negative.

$$\max\left(f\left(x_i\right)\right), s.t. \frac{\omega^T x_i + b}{\|\omega\|} \geq d\left(x_i\right), \quad i = l+1, \cdots l+u. \tag{8}$$

$$\min\left(f\left(x_i\right)\right), s.t. \frac{\omega^T x_i + b}{\|\omega\|} \leq -d\left(x_i\right), \quad i = l+1, \cdots l+u, \tag{9}$$

where the values of p and q determine the number of samples labeled in each iteration, i.e., the learning speed of transductive learning. When p and q equal 1, we apply the pairwise labeling method. When p and q are greater than 1, p or q samples in the boundary region of the optimal classification hyperplane are labeled in each iteration, and their values can be tuned according to the actual application scenario.

4.2.3. Adding Labeled Samples to the Corresponding Classifier. After the samples are labeled, they must be added to the training set for iteration if the stop condition of the algorithm is not met. Instead of adding the samples to all training sets, they are added to those corresponding to classifiers whose output class labels are consistent. For example, if the output results of the five classifiers A, B, C, D, and E are positive class, positive class, positive class, positive class, and negative class, respectively, then adding samples to the training set corresponding to training classifier E is obviously inappropriate. In addition, if the output results of two classifiers A and B meet equation (8) and if the output result of A is 0.05 and B is 0.95, then the probability of A labeling the sample correctly is relatively low compared with B. Thus, the difference in training and the accuracy of the sample labeling can be guaranteed by adding the labeled samples to the training set corresponding to the classifier whose labeled class is consistent with the output result and whose output value is the largest.

4.2.4. Termination Conditions. If in the process of sample labeling and model training the currently labeled sample class is inconsistent with the previously labeled class, then the labeled class must be reset; that is, the sample must be relabeled. The approach in this case is to take the sample as an unlabeled sample again and delete it from the corresponding training set and proceed to the next iteration. If there are no samples that need to be reset in the iteration and there are no unlabeled samples that meet the labeling condition, then the iteration is stopped.

5. Improved TSVM Algorithm Based on Active Learning

The TSVM algorithm is based on multiclassifier collaborative labeling and is designed by combining the TSVM algorithm and the proposed multiclassifier voting decision labeling. The specific steps of Algorithm 1 are as follows:

6. Experiment Evaluation

6.1. Experimental Data Set and Experimental Setting

6.1.1. Data Description. We select the MovieLens data set for the experiment, which comprises movie rating data collected by the GroupLens team of the University of Minnesota. Movies are recommended based on users' scores (1–5) [30]. Four levels of data are provided: 100,000 scores, 1 million scores, 10 million scores, and 20 million scores. We select the first data set of 100,000 real scores for 1682 movies submitted by 943 users, with a sparsity of 94.3%, and a rating range of [1, 5]. The higher the score, the more the user likes that movie. Each user in the data set has rated at least 20 movies. We regard a movie with a score of 3–5 as one liked by the user and is thus labeled as +1 and one with a score of 1–2 as disliked (labeled as −1).

6.1.2. Experimental Setting. The experimental environment of this study is a computer configured with Intel Core i5 1.6 GHZ, 8 G memory, and Windows 10 operating system, and the simulation software is MATLAB 2015b.

In the experiment, we set the parameters, for example, the number of clusters k, of the proposed algorithm through experiments, and all the experimental results are the average results of 5 experiments.

6.1.3. Evaluation Metrics. We use precision and F score to evaluate the performance of the recommendation model. The specific calculation methods are as follows.

Suppose n items are recommended for user u, denoted as $R(u)$. The set of items that user u likes on the test set is $T(u)$. Then, the accuracy and recall are defined as follows:

$$\begin{aligned} \text{precision} &= \frac{\sum_u |R(u) \cap T(u)|}{\sum_u |T(u)|}, \\ \text{recall} &= \frac{\sum_u |R(u) \cap T(u)|}{\sum_u |R(u)|}. \end{aligned} \tag{10}$$

Accuracy and recall are a pair of mutually exclusive indicators that are typically combined, and the F score is used to measure the recommendation quality. The higher the F score, the higher the quality.

$$\text{F score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{11}$$

Input: Labeled sample set L; unlabeled sample set U; the number of classifiers k.
Output: The final classifier TSVM.
Step 1: Apply the K-means algorithm to cluster the labeled sample set L, and extract samples from each cluster according to a specific proportion to form k sub-training sets, which are denoted as $L_1, L_2, \ldots, L_k$.
Step 2: Utilize the SVM algorithm to train k training subsets to obtain k initial classifiers: $C_1, C_2, \ldots, C_k$.
Step 3: Input unlabeled samples into $C_1, C_2, \ldots, C_k$, and obtain k output results: $f_1^i, f_2^i, \ldots, f_k^i$.
Step 4: For any unlabeled sample x_j, if the classification results of a k classifier meet equation (8), then label it as a positive class; if the classification results of a k classifier meet equation (9), then label it as a negative class.
Step 5: If the currently labeled class of x_j is inconsistent with the previously labeled class, cancel the labeling and delete it from the corresponding training set. If the currently labeled class is consistent with the previous and $\max(f(x_j^i)), j = 1, 2, \ldots, k$ is inconsistent, then add the sample to L_j. If the sample is not labeled in the early stage, then find j that meets $\max(f(x_j^i)), j = 1, 2, \ldots, k$, and add the sample to L_j; otherwise, stop the iteration and go to **Step 8**.
Step 6: Repeat steps 4 and 5 until all unlabeled samples are labeled.
Step 7: Train the new training subset and obtain the new classifiers $C_{1\,\mathrm{new}}, C_{2\,\mathrm{new}}, \ldots, C_{k\mathrm{new}}$. If the sub-training sets of the previous and current iterations remain unchanged, the corresponding training should continue to use the classifier from the previous iteration; then, go to **Step 3**.
Step 8: Combine each training subset to form the final training set, and retrain the sample set to obtain the final classifier.

Algorithm 1: TSVM algorithm based on distance strategy and multiclassifier collaborative labeling (DCTSVM).

6.2. Experimental Results

6.2.1. Role of Clustering in Solving the Cold Start Problem for New Users. We assess the performance of the clustering algorithm in alleviating the cold start of new users and compare it with the performance of the following algorithms: user-based collaborative filtering (UserCF) algorithm, cluster-based UserCF algorithm (CUserCF), TSVM algorithm based on the distance metric, proposed multiclassifier voting decision mechanism (DCBTSVM), TSVM algorithm based on the distance metric, and multiclassifier voting decision mechanism without cluster analysis (ALTSVM). Figure 3 presents the performance of these reference algorithms, in terms of their recommendation accuracy, with a different number of clusters.

These results show that the clustering algorithm improves the performance of the recommendation model. By analyzing the characteristics of the data set, the model automatically selects neighbors and excavates the potential association relationship of users (i.e., it looks for users that are highly similar), thus helping new users find their own user groups quickly and alleviating the problem of reduced recommendation accuracy due to user cold start. Owing to the use of cluster analysis, the recommendation accuracy of the CUserCF algorithm is better than that of UserCF. Similarly, the recommendation accuracy of DCBTSVM is better than that of ALTSVM.

In the classification-based recommendation system, the number of samples (user-item) in each cluster gradually decreases as the number of clusters k increases, while the number of classifiers trained (equal to the number of clusters k) using the active learning classification model gradually increases. As the number of samples in each cluster decreases, the ability of the trained model to generalize also changes. When $k = 30$, the recommendation accuracy begins to decline, as the labeling of unlabeled samples by all classifiers is inaccurate; this may be related to model overfitting.

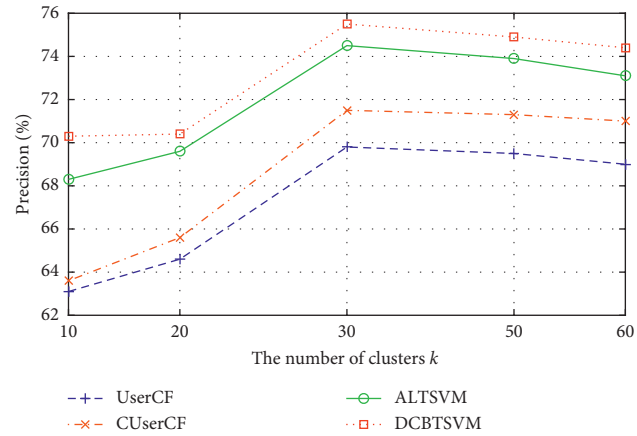

Figure 3: Recommended accuracy of four algorithms with a different number of clusters.

A high level of accuracy (i.e., precision) and the ability to identify as many items of user interest as possible (i.e., recall) are key factors characterizing the performance of a recommendation model. The F score is an important indicator of these abilities. Figure 4 shows that the proposed DCBTSVM method has better F score than the other three methods. The scores are highest when $k = 30$ and when the proportion of training samples accounts for 60% of the whole training sample set.

6.2.2. Active Learning Strategy for Solving Data Sparsity and User Cold Start Problems. To better simulate real online user situations, we divide the users into the MovieLens data set into two groups. We select one group of users and their rated movie data for the initial training set and no longer regard them as new users. Users in the other group are regarded as new users, and their rated movie data are divided into two subgroups. Each user randomly reserves 20 movie scores for

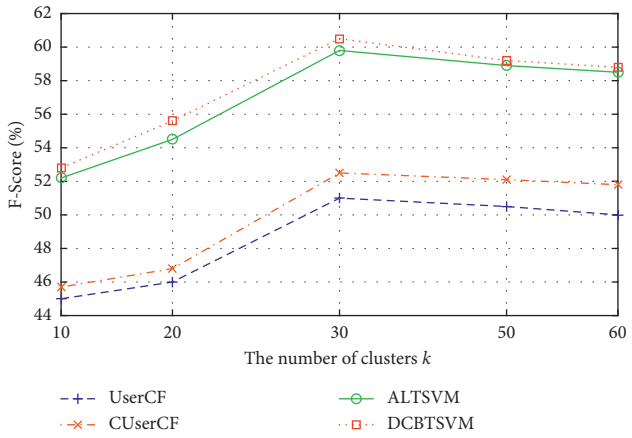

Figure 4: F scores of four algorithms with a different number of clusters.

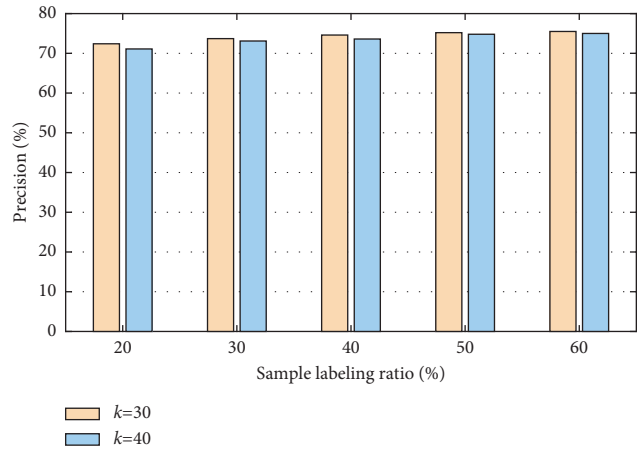

Figure 5: Recommendation accuracy of the DCBTSVM algorithm with different sample labeling ratios when $k = 30$ and $k = 40$, respectively.

the final test set, and the other subgroup is used as an unlabeled sample set. We assume that users can rate any movie. Each time the movie samples are selected from the unlabeled sample set to be labeled, and they are added into the training set to retrain the model.

On the basis of the aforementioned experimental results, the number of clusters k is first set to 30 and then to 40. Figures 5 and 6 show the performance change trend of the proposed DCBTSVM recommendation model as the sample label proportion increases from 20% to 60%. In each iteration, the DCBTSVM model uses the designed active learning strategy to inquire and label unlabeled movie samples. Then, the labeled samples are put into the training set as labeled samples, and the model is retrained, to iterate continuously until the termination condition.

Figures 5 and 6 reveal that the performance of the proposed recommendation algorithm improves as the sample label proportion increases. First, when the number of clusters $k = 30$, the recommendation performance is better than that when $k = 40$. When clustering according to user characteristics, $k = 30$ case yields a wider range of user interests and preferences. For new users, the accuracy rate of being correctly classified into one of the classes is slightly higher; thus, the samples labeled by active learning appear to better reflect the real interests of the users. Second, the preferences of users (both new and old) are better labeled as the proportion of labeled samples increases, thus supplementing the scarce data and making the interest model more effective.

It is worth noting that the active learning strategy does not label all unlabeled samples, but labels that are the most valuable and as few as possible. On the one hand, it can not only reduce the labeling cost of samples; on the other hand, it can reduce the time complexity of model training.

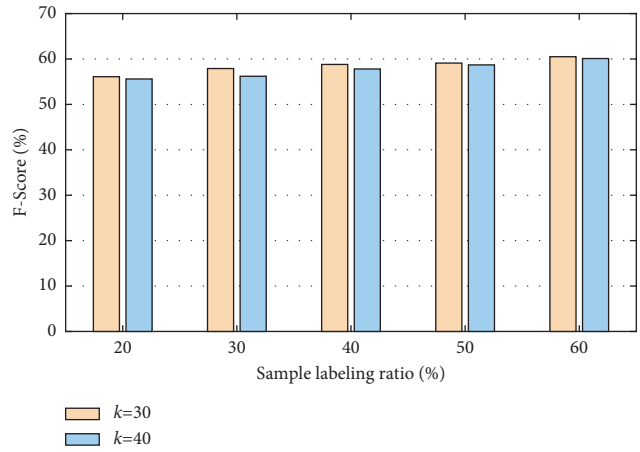

Figure 6: F scores of the DCBTSVM algorithm when $k = 30$ and $k = 40$ with different sample labeling ratios.

6.2.3. Comparison with Other Methods. To verify the effectiveness of the DCBTSVM algorithm, we compare its precision and F score with that of SVM, TSVM, ALTSVM, and UserCF-based models (Figures 7 and 8), where SVM algorithm only uses the labeled samples and performs well in

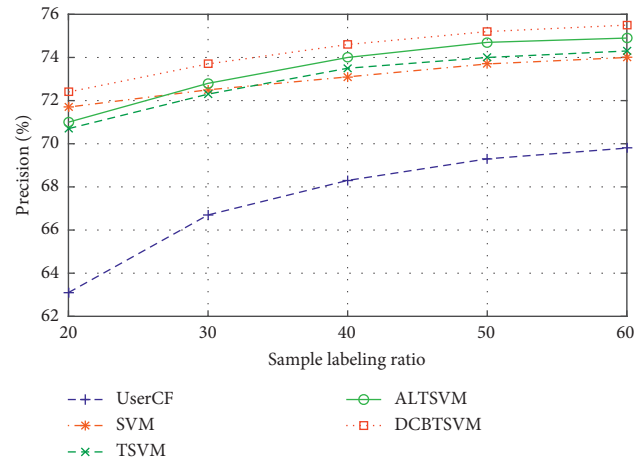

Figure 7: Recommendation accuracy of five algorithms with different sample labeling ratios.

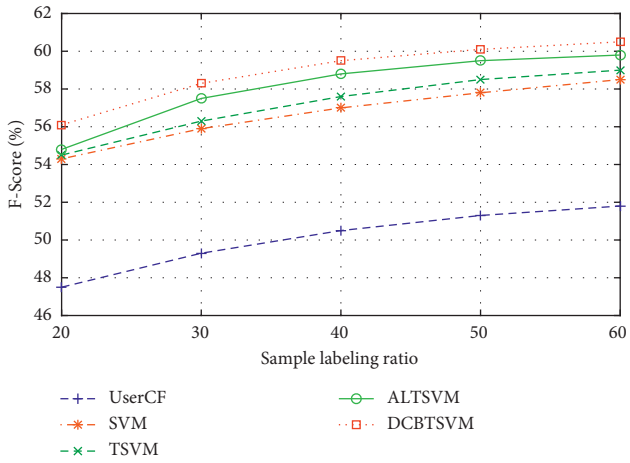

Figure 8: F scores of five algorithms with different sample labeling ratios.

the case of a sufficient number of labeled samples, but the performance will be degraded when the labeled samples are scarce; ALTSVM is the DCBTSVM algorithm without the clustering analysis.

From Figures 7 and 8, it can be found that the precision and F score of the DCBTSVM model are better than those of other recommendation algorithms. Its level of precision and F score also increases with the number of labeled samples, due to the implementation of the active learning strategy. First, the model selects fewer but higher quality samples for labeling, which compensates for the scarce data and enriches the interest model. Second, as it does not rely on the similarity between users or items, the recommendations are not limited to similar modules, and therefore, user choices and cognitive domains can be expanded and their preferences better described. Third, the integration of active learning and semi-supervised learning compensates for the shortcomings of both methods while amplifying their advantages, thus improving the quality of sample labeling. Therefore, the practical value of the method proposed in this study is evident.

7. Conclusion

We propose a novel model-based collaborative filtering algorithm that combines active learning and the semi-supervised transductive support vector machine, which we term DCBTSVM. We use the clustering method to cluster similar users together, thereby alleviating the cold start problem of new users. The sample labeling method design is based on distance measurement and the use of a multiclassifier voting decision to label unlabeled "user-item" association data, thus solving the problems of cold start and data sparsity. The resulting DCBTSVM-based recommendation model is effective and offers personalized and high-quality recommendations. Experimental results obtained using MovieLens data demonstrate that the proposed model provides efficient and accurate recommendations.

Data Availability

The data set used in the experiment of this study is a public data set, which can be freely obtained by researchers. For details, the following website can be reversed: https://grouplens.org/datasets/movielens/.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (grant nos. 72161005, 71901078, and 71964009), Technology Foundation of Guizhou Province (grant nos. QianKeHeJiChu[2020]1Y269 and [2018]1068), High-Level Talent Project of Guizhou Institute of Technology (grant no. XJGC20190929), and Special Key Laboratory of Artificial Intelligence and Intelligent Control of Guizhou Province (grant no. KY[2020]001).

References

[1] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: a survey and new perspectives," *ACM Computing Surveys*, vol. 52, no. 1, pp. 1–38, 2019.

[2] J. Bu, X. Shen, B. Xu, C. Chen, X. He, and D. Cai, "Improving collaborative recommendation via user-item subgroups," *IEEE Transactions on Knowledge and Data Engineering*, vol. 28, no. 9, pp. 2363–2375, 2016.

[3] Y. Shi, M. Larson, and A. Hanjalic, "Collaborative filtering beyond the user-item matrix," *ACM Computing Surveys*, vol. 47, no. 1, pp. 1–45, 2014.

[4] Z. Cui, X. Xu, F. Xue et al., "Personalized recommendation system based on collaborative filtering for IoT scenarios," *IEEE Transactions on Services Computing*, vol. 13, no. 4, pp. 685–695, 2020.

[5] A. Da'u, N. Salim, I. Rabiu, and A. Osman, "Recommendation system exploiting aspect-based opinion mining with deep learning method," *Information Sciences*, vol. 512, pp. 1279–1292, 2020.

[6] M. Li, Y. Li, W. Lou, and L. Chen, "A hybrid recommendation system for Q&A documents," *Expert Systems with Applications*, vol. 144, Article ID 113088, 2020.

[7] W. Zhang and J. Wang, "A collective bayesian Poisson factorization model for cold-start local event recommendation," in *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1455–1464, Sydney NSW Australia, August 2015.

[8] L. Wang, Y. Liu, and J. Wu, "Research on financial advertisement personalised recommendation method based on customer segmentation," *International Journal of Wireless and Mobile Computing*, vol. 14, no. 1, pp. 97–101, 2018.

[9] Z. Hu and Y. Hu, "Research on collaborative filtering recommendation bottleneck problem," *Wireless Internet Technology*, vol. 9, no. 46, pp. 100-101, 2016.

[10] N. Rubens, M. Elahi, M. Sugiyama, and D. Kaplan, "Active learning in recommender systems," *Recommender Systems Handbook*, Springer, Boston, MA, USA, pp. 809–846, 2015.

[11] M. Elahi, F. Ricci, and N. Rubens, "Active learning strategies for rating elicitation in collaborative filtering: a system-wide

perspective," *ACM Transactions on Intelligent Systems & Technology*, vol. 5, no. 1, pp. 1–33, 2014.

[12] M. Elahi, V. Repsys, and F. Ricci, "Rating elicitation strategies for collaborative filtering," in *Proceedings of the International Conference on Electronic Commerce and Web Technologies*, pp. 160–171, Toulouse, France, 29 August 2011.

[13] R. Karimi, C. Freudenthaler, A. Nanopoulos, and L. Schmidt-Thieme, "Non-myopic active learning for recommender systems based on matrix factorization," in *Proceedings of the 2011 IEEE International Conference on Information Reuse & Integration*, pp. 299–303, Nevada, USA, 3 August 2011.

[14] A. M. Rashid, I. Albert, D. Cosley, and S. K. Lam, "Getting to know you: learning new user preferences in recommender systems," in *Proceedings of the 7th International Conference on Intelligent User Interfaces*, pp. 127–134, San Francisco, CA, USA, 13 January 2002.

[15] Y. Song, L. Zhang, and C. L. Giles, "Automatic tag recommendation algorithms for social recommender systems," *ACM Transactions on the Web*, vol. 5, no. 1, pp. 1–31, 2011.

[16] M. Li, L. Wen, and F. Chen, "A novel Collaborative Filtering recommendation approach based on Soft Co-Clustering," *Physica A: Statistical Mechanics and Its Applications*, vol. 561, Article ID 125140, 2021.

[17] M.-L. Wu, C.-H. Chang, and R.-Z. Liu, "Integrating content-based filtering with collaborative filtering using co-clustering with augmented matrices," *Expert Systems with Applications*, vol. 41, no. 6, pp. 2754–2761, 2014.

[18] N. Mirbakhsh and C. X. Ling, "Leveraging clustering to improve collaborative filtering," *Information Systems Frontiers*, vol. 20, no. 1, pp. 111–124, 2018.

[19] T. Xiao and H. Shen, "Neural variational matrix factorization for collaborative filtering in recommendation systems," *Applied Intelligence*, vol. 49, no. 10, pp. 3558–3569, 2019.

[20] Y. Li, D. Wang, H. He, L. Jiao, and Y. Xue, "Mining intrinsic information by matrix factorization-based approaches for collaborative filtering in recommender systems," *Neurocomputing*, vol. 249, pp. 48–63, 2017.

[21] X. Yuan, L. Han, S. Qian, G. Xu, and H. Yan, "Singular value decomposition based recommendation using imputed data," *Knowledge-Based Systems*, vol. 163, pp. 485–494, 2019.

[22] T. Joachims, "Transductive inference for text classification using support vector machines," in *Proceedings of the 16th International Conference on Machine Learning*, pp. 200–209, San Francisco CA, 27 June 1999.

[23] T. Joachims, "Transductive support vector machines," *Semi-Supervised Learning*, pp. 105–118, MIT Press, Cambridge, MA, 2006.

[24] T. Joachims, *Learning to Classify Text Using Support Vector machines*, Springer Science & Business Media, Berlin, Germany, 2002.

[25] X. Wang, Z. Dai, H. Li, and J. Yang, "A new collaborative filtering recommendation method based on transductive SVM and active learning," *Discrete Dynamics in Nature and Society*, vol. 2020, Article ID 6480273, 15 pages, 2020.

[26] S. Cassel, F. Howar, B. Jonsson, and B. Steffen, "Active learning for extended finite state machines," *Formal Aspects of Computing*, vol. 28, no. 2, pp. 233–263, 2016.

[27] M. Sharma and M. Bilgic, "Evidence-based uncertainty sampling for active learning," *Data Mining and Knowledge Discovery*, vol. 31, no. 1, pp. 164–202, 2017.

[28] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions," in *Proceedings of the ICML 2003 Workshop on the Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*, pp. 58–65, Washington, DC, USA, March 2003.

[29] H. Hassanzadeh and M. Keyvanpour, "A two-phase hybrid of semi-supervised and active learning approach for sequence labeling," *Intelligent Data Analysis*, vol. 17, no. 2, pp. 251–270, 2013.

[30] GroupLens. MovieLensdatasets[EB/OL]. https://grouplens.org/datasets/movielens.2016-11-03/2021-08-10.