INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 11(2):248-258, April 2016.

Sparse Online Learning for Collaborative Filtering

F. Lin, X. Zhou, W.H. Zeng

Fan Lin*
Software School, Xiamen University
China, 308B of General Office at Haiyun Campus, Xiamen University, Xiamen, 361009
*Corresponding author: iamafan@xmu.edu.cn

Xiuze Zhou
Department of Automation, Xiamen University
China, General Office at Haiyun Campus, Xiamen University, Xiamen, 361009
zhouxiuze@foxmail.com

Wenhua Zeng
Software School, Xiamen University
China, 502 of General Office at Haiyun Campus, Xiamen University, Xiamen, 361009
whzeng@xmu.edu.cn

Abstract:
With the rapid growth of Internet information, our individual processing capacity has become over-whelming. Thus, we really need recommender systems to provide us with items online in real time. In reality, a user's interest and an item's popularity are always changing over time. Therefore, recommendation approaches should take such changes into consideration. In this paper, we propose two approaches, i.e., First Order Sparse Collaborative Filtering (SOCFI) and Second Order Sparse Online Collaborative Filtering (SOCFII), to deal with the user-item ratings for online collaborative filtering. We conduct some experiments on such real data sets as MovieLens100K and MovieLens1M, to evaluate our proposed methods. The results show that, our proposed approach is able to effectively online update the recommendation model from a sequence of rating observation. And in terms of RMSE, our proposed approach outperforms other baseline methods.
Keywords: Recommender systems, Collaborative Filtering, Online learning, SOCFI, SOCFII

1 Introduction

With the prosperity of such large-scale online commercial websites and online shopping websites as Amazon [1], Barnes, Netflix, eBay, etc, users are continuously exposed into increasing amount of items. Consequently, the information flow which is increasingly complex and huge makes the user lost, and thereby be tired of the inefficient search. In order to deal with this problem, and also to predict the user's unknown preferences based on some user's preferences we have studied [2,3], a modern technique named Collaborative Filtering(CF) is put forward.CF has become widely used as one of the most successful learning techniques to build real-world recommendation systems.

Consider online e-commerce applications where a user wishes to watch a movie or buy a product, the system offers recommendations using CF techniques in exploiting one's previous preference and that of others. A good recommendation system is extremely beneficial to users in accurately predicting their preferences and providing satisfactory recommendations, and consequently benefiting the company [1]. The fundamental assumption of CF is that if two users rate many items similarly, they will be likely to rate other items similarly [2].

Despite the successful application in such many fields as book [6], music [7], news [8], etc, all these traditional CF approaches share a common but critical drawback that these approaches have to be re-trained completely from scratch whenever new training data arrives, which is clearly non-scalable for large real recommendation systems in which user's rating data often arrives sequentially and frequently [5, 9]. It takes quite a long time to learn a new model when large numbers of parameters need to be estimated. The popularity of items and the interests of users are always changing over time [10, 11]. Therefore, Recommendation approaches shall take these changes into consideration.

Traditional CF approaches work well on the user-item rating data when the latent factors number k is very small. However, when the k become large, these approaches fail to well deal with the user-item rating data, since the user-item rating data is sparse and of large scale, and many approaches need a large k to get accurate results. Although the batch algorithm of matrix factorization has a high accuracy, we can't stand the high memory cost and time complexity in the real-world recommender system.

Nowadays, the data recommendation system which has a enormous amount of data is characterized as followed [12]: (1) high volume, system need to deal with huge amount of training data; (2) high velocity, new data often arrives very rapidly and sequentially; (3) high dimensionality, the data from users has a large number of features; (4) high sparsity, many feature elements are zero.

To tackle the above challenges, recent years have witnessed some studies for online collaborative filtering [3, 4]. The state-of-the-art Online Collaborative Filtering (OCF) approach avoids the highly expensive re-training cost of traditional batch matrix factorization algorithms by applying the simple online gradient descent (OGD) algorithms to solve the matrix factorization task [3]. Muqeet Ali et al. proposed a parallel collaborative filtering for streaming data by using distributed stochastic gradient descent algorithm [4].

Unfortunately, these methods are generally based on the first order optimization framework (e.g., online gradient descent) to find the optimal solutions of low-rank matrix factorization. The ignorance of second order information results in the slow convergence of these approaches. Besides, the latent factors number is actually quite likely to be very large which is a difficulty for the first order optimization framework even the framework which has already taken the second order information. To tackle this issue, we propose to solve the following sparse collaborative filtering problem. To address the weakness of these first order or second order online CF approaches and reduce data storage space and increase computing speed, we propose such Sparse Online Collaborative Filtering (SOCF) as First Order Sparse Collaborative Filtering (SOCFI) and Second Order Sparse Online Collaborative Filtering (SOCFII). Our proposed approach is able to effectively online update the recommendation model from a sequence of rating observation. The Sparse Online Collaborative Filtering (SOCF) takes consider the latent factors of the low rank matrix and online second order optimization method. The key idea of SOCF is to not only update the user and item weight vectors at each round, but also estimate their distribution and take full account of large latent factors. Because of full account of this case, SOCF converges significantly faster and thus achieves much lower values of RMSE and MAE than those of the regular first order algorithms when receiving the same amount of rating observations.

The rest of the paper is organized as follows. Section 2 introduces the background information and presents the problem formulation. Section 3 exhibits the proposed Sparse Collaborative Filtering algorithm, which takes first order and second order information into consideration. Section 4 presents our experimental results and analysis. Section 5 draws conclusions and discusses the future work.

2 Background Information and Related Work

In this section, we introduce some background information, related works and the problem we're going to solve.

Collaborative filter (CF) and content-based filtering are two strategies widely used in recommendation systems for recommending items for users. CF makes prediction by using only the user-item interaction indormation without additional information or domain knowledge, so it has a wider application. The key idea of the CF is that users who have similar preferences in the past are likely to have similar preferences in the future.

Memory-based algorithms show good performance on accuracy, but they cannot handle scalability and sparsity problems of data. All these CF algorithms achieved good results without using additional information. In order to solve the data sparsity problems, many model-based CF methods have been proposed. Model-based CF techniques aim at building a model to represent user rating data, and use that model to predict user preference for a specific item. For example, the Singular Value Decompositon (SVD) obtains the main factors to reduce dimensimality. Hofmann converts the Latent semantic model from information retrieval to collaborative filtering. These models not only reduce the dimensions of the user-item matrix and smooth out the noise information, but also help the algorithm to alleviate scalability of data.

Recommendation systems provide an effective way for information filtering to discover useful information according to the historical preferences expressed by users [15]. At present, CF approaches, as the most widerly used method in recommendation system, can be generally grouped into two types: model-based CF and memory-based CF. Model-based CF approaches provide item with recommendation by first developing a model of user ratings, and then predicting user's preference for a specific item through reained model [13–16]. While memory-based approaches predict rating of users according to all user ratings [1, 7, 20]. Generally, the Model-based CF approaches is more accurate than memory-based approaches [21].

Matrix factorization is one of the most popular and the state-of-the-art methods of model-based CF approaches, which was used by the winner of the Netflix prize [22–25]. SVD puts the items that are highly relevant and apparent together as a Singular factor, and breaks up the vector into a small order approximation matrix. One user with one item represents a vector in this space and the rating that a user assigns to an item is the dot product of their feature vectors [26]. The key idea of latent factor model assumes that the similarity between users and items is discovered by the lower-dimension data. The system minimizes the regularized squared error on the set of known ratings to learn the factor vectors [27]. Low rank matrix factorization is considered to be a very effective method and achieves good results in practice [20, 28].

We will base our matrix factorization study on the collaborative filtering, which is traditionally defined as:

$$\sum_{(a,b)\in A} l(r_{a,b}, U_a, V_b) + \frac{\lambda}{2}\left(\sum_{a=1}^{m} \|U_a\|^2 + \sum_{b=1}^{n} \|V_b\|^2\right)$$

Where $A \subseteq \{(a,b)|r_{a,b}$ is known$\}$, $l(r,U,V) = (r - U^T V)^2$, $U_a, V_b \in R^k$ and λ is a regularization parameter. A is the user-item rating set we have known.

Although the matrix factorization technique can obtain high accuracy, we cannot stand with it for a long time to run. Stochastic gradient descent algorithm requires an iterative many times until convergence. In practical, the model-based approaches have to be retrained completely for new records when new users' rating data arrives sequentially and frequently [6]. In contrast to traditional collaborative filtering algorithms, online learning promptly update the predictive model and able to avoid expensive re-training cost when a new instance appears [7, 8]. In online algorithms, these models need to be retrained when each new data arrives. The online algorithms only need a single iteration which processes the events in the order of time over the training data [30]. Although the online learning algorithm loses some accuracy, it avoids high time complexity and memory cost.

3 Sparse Collaborative Filtering Algorithm

In this section, we introduce our proposed sparse collaborative filtering algorithms, including First Sparse Order Collaborative Filtering (SOCFI) algorithm and Second Order Online Sparse Collaborative Filtering (SOCFII) algorithm.

3.1 First Online Sparse Collaborative Filtering Algorithm

With N users and M items in the user-item rating matrix is broken up into two low rank matrixes. The user-item rating matrix $R \in R^{N*M}$ is broken down into two low rank matrixes. U_a is the a-th row from the user matrix $U \in R^{N*K}$, and V_b is the b-th row form item matrix $V \in R^{M*K}$. The rank of $K \ll \min\{N, M\}$. $r_{a,b}$ is the movie b rated by user a. The predicted score is the dot product of U_a and V_b, i.e., $\widehat{r_{a,b}} = U_a^T V_b$. $|C|$ represents the number of observed ratings. In general, one can define different type of loss function for different purposes. For example, for the Root Mean Square Error (RMSE), i.e., , we define the loss by the square error function as:

$$RMSE = \sqrt{\frac{1}{|C|} \sum_{(a,b)\in C} (r_{a,b} - \widehat{r_{a,b}})^2}$$

We define the loss by the square error function as:

$$l(U_a, V_b, r_{a,b}) = (r_{a,b} - U_a V_b^T)^2$$

And for the Mean Absolute Error (MAE), i.e., $MAE = \frac{1}{|C|} \sum_{(a,b)\in C} |r_{a,b} - \widehat{r}_{a,b}|$, we define the absolute loss function as:

$$l(U_a, V_b, r_{a,b}) = |r_{a,b} - U_a V_b^T|$$

Traditionally, k is treated as the latent factors number. When k is set as a small value, the traditional algorithm will work well. However, when k is set as a large number, the algorithm will fail, which implies the traditional algorithm cannot tackle tasks with large factors number. However, the latent factors number is quite likely to be very large in reality. To tackle this issue, we propose to solve the following sparse collaborative filtering problem,

$$\sum_{(a,b)\in A} l(r_{a,b}, u_a, v_b) + \frac{\lambda}{2}\left(\sum_{a=1}^{m} \|u_a\|^2 + \sum_{b=1}^{n} \|v_b\|^2\right) + \tau \left(\sum_{a=1}^{m} \|u_a\|_1 + \sum_{b=1}^{n} \|v_b\|_1\right)$$

Furthermore, since the data usually comes one by one, we propose to solve the sparse collaborative filtering problems through online learning techniques. Specifically, we will update the two vectors as follows:

$$\boldsymbol{u}_a \leftarrow arg \min_{\boldsymbol{u}} \langle \partial_u l(r_{a,b}, \boldsymbol{u}_a, \boldsymbol{v}_b) + \lambda \boldsymbol{u}_a, \boldsymbol{u}\rangle + \tau\|\boldsymbol{u}\|_1 + \frac{1}{2\eta_t}\|\boldsymbol{u} - \boldsymbol{u}_a\|^2$$

And

$$\boldsymbol{v}_b \leftarrow arg \min_{\boldsymbol{v}} \langle \partial_v l(r_{a,b}, \boldsymbol{u}_a, \boldsymbol{v}_b) + \lambda \boldsymbol{v}_b, \boldsymbol{v}\rangle + \tau\|\boldsymbol{v}\|_1 + \frac{1}{2\eta_t}\|\boldsymbol{v} - \boldsymbol{v}_b\|^2$$

These two updates enjoys closed-form solutions:

$$\boldsymbol{u}_a \leftarrow ST_{\tau\eta t}[(1-\eta_t\lambda)\boldsymbol{u}_a - \eta_t\partial_{\boldsymbol{u}} l(r_{a,b},\boldsymbol{u}_a,\boldsymbol{v}_b)] \tag{1}$$

And

$$\boldsymbol{v}_b \leftarrow ST_{\tau\eta t}[(1-\eta_t\lambda)\boldsymbol{v}_b - \eta_t\partial_{\boldsymbol{v}} l(r_{a,b},\boldsymbol{u}_a,\boldsymbol{v}_b)] \tag{2}$$

Where $ST_v(w) = sign(w) \odot [|w| - v]_+$ and $\odot$ denotes element-wise product.

3.2 Second Online Sparse Collaborative Filtering Algorithm

In order to improve the convergence speed, we propose a second order online collaborative filtering algorithm. The key idea of SOCFII is to not only update the user and item weight vectors at each round, but also estimate their distribution, i.e., mean and covariance matrix. In the second order online collaborative filtering, where u_a and v_b are assumed satisfy Gaussian distributions. The objective functions are

$$D_{KL}\left(N(\mu_{\boldsymbol{u}_a},\sum\nolimits_{\boldsymbol{u}_a})\|N(\mu_{\boldsymbol{u}_a,t},\sum\nolimits_{\boldsymbol{u}_a,t})\right) + \eta l(r_{a,b},\boldsymbol{u}_a,\boldsymbol{v}_b) + \frac{\lambda}{2}\boldsymbol{v}_b^T\sum\nolimits_{\boldsymbol{u}_a}\boldsymbol{v}_b$$

And

$$D_{KL}\left(N(\mu_{\boldsymbol{v}},\sum\nolimits_{\boldsymbol{v}_b})\|N(\mu_{\boldsymbol{v}_a,t},\sum\nolimits_{\boldsymbol{v}_b,t})\right) + \eta l(r_{a,b},\boldsymbol{u}_a,\boldsymbol{v}_b) + \frac{\lambda}{2}\boldsymbol{u}_a^T\sum\nolimits_{\boldsymbol{v}_b}\boldsymbol{u}_a$$

Where K_{KL} is KL divergence.

In this way, the algorithm significantly outperform first order algorithm. However, the latent number k has to be set as a small value. To solve this issue we proposed the following two online objective functions:

$$\begin{aligned}C_{\boldsymbol{u}_a}(\mu_{\boldsymbol{u}_a},\sum\nolimits_{\boldsymbol{u}_a}) = {} & D_{KL}\left(N(\mu_{\boldsymbol{u}_a},\sum\nolimits_{\boldsymbol{u}_a})\|N(\mu_{\boldsymbol{u}_a,t},\sum\nolimits_{\boldsymbol{u}_a,t})\right) + \eta\langle\partial_{\boldsymbol{u}} l(r_{a,b},\mu_{\boldsymbol{u}_a,t},\boldsymbol{v}_b),\mu_{\boldsymbol{u}_a}\rangle \\ & + \frac{\lambda}{2}\boldsymbol{v}_b^T\sum\nolimits_{\boldsymbol{u}_a}\boldsymbol{v}_b + \eta\tau\|\mu_{\boldsymbol{u}_a}\|_1\end{aligned}$$

And

$$\begin{aligned}C_{\boldsymbol{v}_b}(\mu_{\boldsymbol{v}_b},\sum\nolimits_{\boldsymbol{v}_b}) = {} & D_{KL}\left(N(\mu_{\boldsymbol{v}_b},\sum\nolimits_{\boldsymbol{v}_b})\|N(\mu_{\boldsymbol{v}_b,t},\sum\nolimits_{\boldsymbol{v}_a,t})\right) + \eta\langle\partial_{\boldsymbol{v}} l(r_{a,b},\mu_{\boldsymbol{v}_b,t},\boldsymbol{v}_b),\mu_{\boldsymbol{v}_b}\rangle \\ & + \frac{\lambda}{2}\boldsymbol{u}_a^T\sum\nolimits_{\boldsymbol{v}_b}\boldsymbol{u}_a + \eta\tau\|\mu_{\boldsymbol{v}_b}\|_1\end{aligned}$$

These objectives linearize the loss functions and introduce sparsity regularization. We can solve these two objectives in two steps:

$$\begin{aligned}\mu_{\boldsymbol{u}_a,t+1} &= arg\min_{\mu_{\boldsymbol{u}_a}} C_{\boldsymbol{u}_a}(\mu_{\boldsymbol{u}_a},\sum\nolimits_{\boldsymbol{u}_a}) \\ &= arg\min_{\mu_{\boldsymbol{u}_a}} \frac{1}{2}(\mu_{\boldsymbol{u}_a}-\mu_{\boldsymbol{u}_a},t)^T\sum\nolimits_{\mu_{\boldsymbol{u}_a,t}}^{-1}(\mu_{\boldsymbol{u}_a}-\mu_{\boldsymbol{u}_a,t}) \\ &\quad + \eta\langle\partial_{\boldsymbol{u}} l(r_{a,b},\mu_{\boldsymbol{u}_a,t},\boldsymbol{v}_b),\mu_{\boldsymbol{u}_a}\rangle + \eta\tau\|\mu_{\boldsymbol{u}_a}\|_1\end{aligned}$$

$$\sum\nolimits_{\boldsymbol{u}_a,t+1} = arg\min_{\sum_{\boldsymbol{u}_a}} C_{\boldsymbol{u}_a}(\mu_{\boldsymbol{u}_a},\sum\nolimits_{\boldsymbol{u}_a})$$

$$\text{i.e.,} \sum\nolimits_{\boldsymbol{u}_a,t+1} = \sum_{\boldsymbol{u}_a,t} - \frac{\sum_{\boldsymbol{u}_a,t}\boldsymbol{v}_b\boldsymbol{v}_b^T\sum_{\boldsymbol{u}_a,t}}{1/\lambda + \boldsymbol{v}_b^T\sum_{\boldsymbol{u}_a,t}\boldsymbol{v}_b} \tag{3}$$

And

$$\begin{aligned}
\mu_{\boldsymbol{v}_b,t+1} &= arg\min_{\mu_{\boldsymbol{v}_b}} C_{\boldsymbol{v}_b}(\mu_{\boldsymbol{v}_b}, \sum\nolimits_{\boldsymbol{v}_b}) \\
&= arg\min_{\mu_{\boldsymbol{v}_b}} \frac{1}{2}(\mu_{\boldsymbol{v}_b} - \mu_{\boldsymbol{v}_b,t})^T \sum\nolimits_{\mu_{\boldsymbol{v}_b,t}}^{-1} (\mu_{\boldsymbol{v}_b} - \mu_{\boldsymbol{v}_b,t}) \\
&= + \eta\langle \partial_{\boldsymbol{v}} l(r_{a,b}, \mu_{\boldsymbol{v}_b,t}, \boldsymbol{v}_b), \mu_{\boldsymbol{v}_b}\rangle + \eta\tau\|\mu_{\boldsymbol{v}_b}\|_1
\end{aligned}$$

$$\begin{aligned}
\sum\nolimits_{\boldsymbol{v}_b,t+1} &= arg\min_{\sum_{\boldsymbol{v}_b}} C_{\boldsymbol{v}_b}(\mu_{\boldsymbol{v}_b}, \sum\nolimits_{\boldsymbol{v}_b}) \\
\text{i.e.,} \sum\nolimits_{\boldsymbol{v}_b,t+1} &= \sum_{\boldsymbol{v}_b,t} - \frac{\sum_{\boldsymbol{v}_b,t} \boldsymbol{u}_a \boldsymbol{u}_a^T \sum_{\boldsymbol{v}_b,t}}{1/\lambda + \boldsymbol{v}_b^T \sum_{\boldsymbol{v}_b,t} \boldsymbol{v}_b}
\end{aligned} \tag{4}$$

In practice, it is computationally expensive to get $\mu_{\boldsymbol{u}_a,t+1}$, $\mu_{\boldsymbol{v}_b,t+1}$ and update $\sum_{\boldsymbol{u}_a,t+1}$, $\sum_{\boldsymbol{v}_b,t+1}$. To solve this issue, we can set the covariance matrices as diagonal, which will produce the following closed-form solutions for the two mean vectors:

$$(\mu_{\boldsymbol{u}_a,t+1})_i = ST_{\eta\tau(\sum_{\boldsymbol{u}_a,t})_{i,i}} \left[(\mu_{\boldsymbol{u}_a,t})_i - \eta(\sum\nolimits_{\boldsymbol{u}_a,t})_{i,i}(\partial_{\boldsymbol{u}} l(r_{a,b}, \mu_{\boldsymbol{u}_a,t}, \boldsymbol{v}_b))_i\right] \tag{5}$$

And

$$(\mu_{\boldsymbol{v}_b,t+1})_i = ST_{\eta\tau(\sum_{\boldsymbol{v}_b,t})_{i,i}} \left[(\mu_{\boldsymbol{v}_b,t})_i - \eta(\sum\nolimits_{\boldsymbol{v}_b,t})_{i,i}(\partial_{\boldsymbol{v}} l(r_{a,b}, \mu_{\boldsymbol{v}_b,t}, \boldsymbol{v}_b))_i\right] \tag{6}$$

Algorithm 1 First Order Sparse Online Collaborative Filtering (SOCFI)

```
Require: a sequence of rating pairs $\{(a_t, b_t, r_{ab}), t = 1, \cdots, T\}$
1: Initialization: initialize a random matrix for user matrix $U \in R^{n\times k}$ and item matrix $V \in R^{m\times k}$
2: for $t = 1, 2 \cdots, T$ do
3:     Receive rating prediction request of user $a_t$ on item $b_t$
4:     Make prediction $\widehat{r_{a_t b_t}} = U_{a_t} V_{b_t}^T$
5:     The true rating $r_{a_t b_t}$ is revealed
6:     The algorithm suffers a loss $l(U_a, V_b, r_{a,b})$
7:     Update $U_{a_t}$ and $V_{b_t}$ by (1), (2), respectively
8: end for
```

4 Experiment

In this section, we present the experimental results to evaluate the performance of our proposed methods by using online the Root Square Error (RMSE) on the data set.

Our experiments are performed on two real data sets: MovieLens100k and MovieLens1M. These two data sets are classic movie rating data sets collected by the MovieLens web site (www.movielens.umn.edu). MovieLens is publicly available data set, and it is widely used to study recommendation systems. The MovieLens100K consists of 100,000 ratings from 943 users on 1,682 movies and the MovieLens1M consists of 1,000,209 ratings from 6,040 users on 3,900 movies.

Algorithm 2 Second Order Sparse Online Collaborative Filtering (SOCFII)

```
Require: a sequence of rating pairs $\{(a_t, b_t, r_{ab}), t = 1, \cdots, T\}$
1: Initialization: initialize a random matrix for user matrix $U \in R^{n\times k}$ and item matrix $V \in R^{m\times k}$, and covariance matrix $\sum_U, \sum_V$ to be item I
2: for $t = 1, 2 \cdots, T$ do
3:     Receive rating prediction request of user $a_t$ on item $b_t$
4:     Make prediction $\widehat{r_{a_t b_t}} = U_{a_t} V_{b_t}^T$
5:     The true rating $r_{a_t b_t}$ is revealed
6:     The algorithm suffers a loss $l(U_a, V_b, r_{a,b})$
7:     Update $U_{a_t}$ and $V_{b_t}$ by (3), (4), respectively
8:     Update $\sum_{U_a}$ and $\sum_{V_b}$ by (5), (6), respectively
9: end for
```

4.1 Compared Algorithm

We compare our methods with other two online algorithm CF algorithms as follow:

(1) OCF: Online Collaborative Filtering for learning a rank-k matrix factorization by using online gradient descent [3];

(2) DA-OCF: Dual-Averaging method of probabilistic matrix factorization for Online Collaborative Filtering by absorbing previous rating information in an approximate average gradient of the loss [9];

(3) SOCFI: First Order Sparse Online Collaborative Filtering;

(4) SOCFII: Second Order Sparse Online Collaborative Filtering by setting covariance matrices as diagonal to simplify the calculation.

Our experiments are conducted on two real data sets, i.e. MovieLens100k and MovieLens1M, which are classic movie rating data sets collected by the MovieLens web site (www.movielens.umn.edu). MovieLens, as a publicly available data set, is widely used to study recommendation systems. The MovieLens100K consists of 100,000 ratings from 943 users on 1,682 movies while the MovieLens1M consists of 1,000,209 ratings from 6,040 users on 3,900 movies.

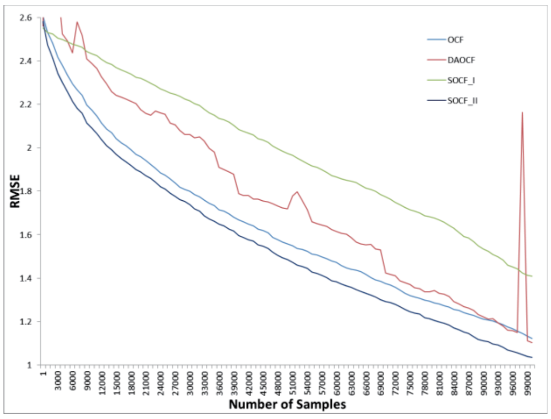
The rank parameter k of matrix u and v is set to four cases: 5, 10 and 50, respectively. After all the parameters are set, all the experiments are conducted 10 times randomly for each data set. To make a fair comparison, the learn arte r of all algorithms is set to 0.01, and λ parameter in DAOCF is set to 0.006, which was suggested to achieve the best performance according to reference [31].

4.2 Experiment and analysis

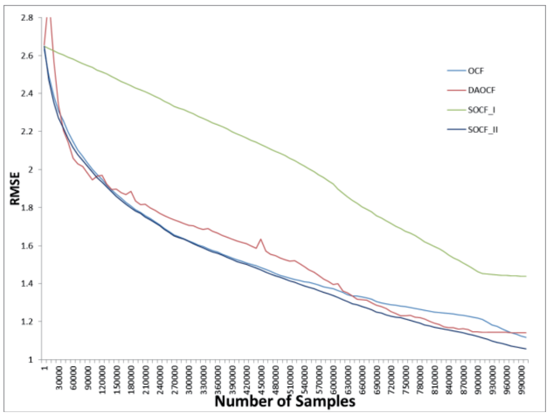
Table 1: The results of MovieLens100K

MovieLens100K	k=5	k=10	k=50
	RMSE	RMSE	RMSE
OCF	1.1218	0.9904	1.5007
DAOCF	1.1038	1.0882	**1.2654**
SOCFI	1.4083	1.2502	1.2886
SOCFII	**1.0355**	**0.9859**	1.3191

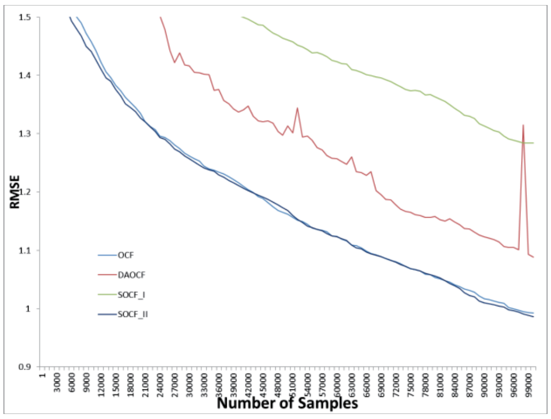
For performance metric, we evaluate the performance of online collaborative filtering algorithms by measuring their scores of online Root Square Error (RMSE) on the test set. The



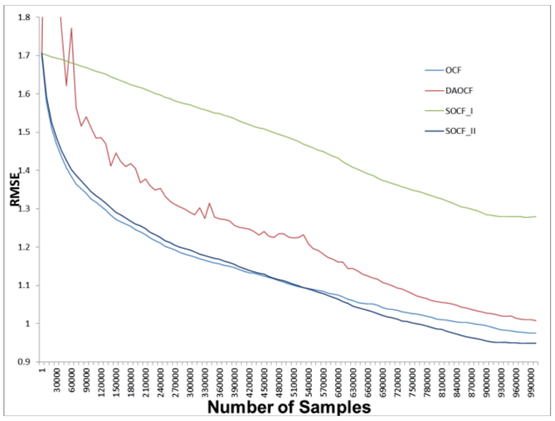
(a) MovieLens100K, K=5



(b) MovieLens1M, K=5



(c) MovieLens100K, K=10



(d) MovieLens1M, K=10

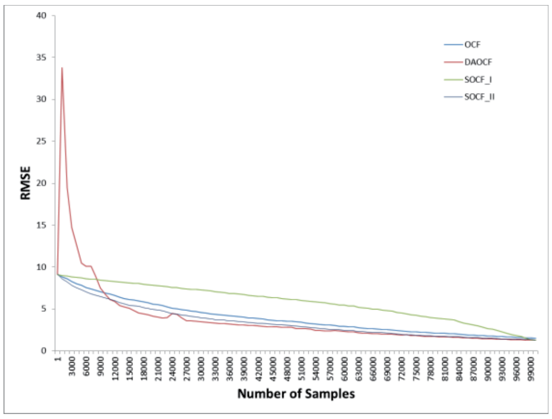
average performance of all approaches is shown in Table 1.

Table 2: The results of MovieLens1M

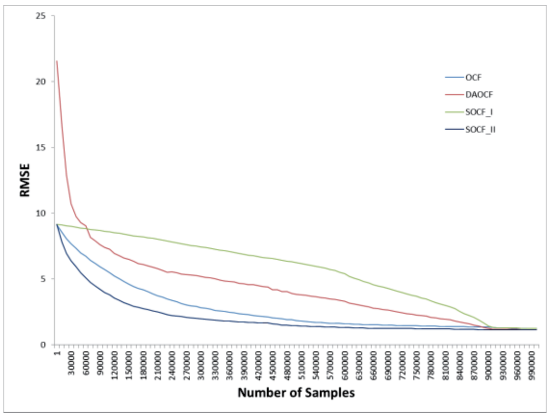
MovieLens1M	k=5	k=10	k=50
	RMSE	RMSE	RMSE
OCF	1.1147	0.9769	1.2345
DAOCF	**1.0136**	1.0077	1.1286
SOCFI	1.4201	1.2702	1.2577
SOCFII	**1.0573**	**0.9486**	**1.1179**

By comparison with OCF and DAOCF approaches, we can find from table 1 that the SOCFII always achieves best results on these data sets, while has smaller RSME values in all cases.

When k is set to a small value, such as 5 or 10, both of the traditional algorithm DAOCF and the SOCFII we proposed perform very well in small scale data sets and large scale data sets. In small scale data sets in which k is set to a large value, such as 50, the DAOCF works better than SOCFII. However, when the data sets become large ones, SOCFII achieves best results. Due to the SOCFI algorithm's lost on second order information, the SOCFI may not perform very well in the case of a very small k on both small scale data sets and large scale data sets. However, in reality, the latent factors number k stands a good chance to be very large. When we turn the k into a large value, adapting to the real situation, the SOCFI achieves better results on these



(e) MovieLens100k, K=50



(f) MovieLens1M, K=50

Figure 1: Performance of online collaborative filtering approaches on the MovieLens100k and MovieLens1M data sets.

data sets. This shows that our proposed method which exploits the confidence information can better handle data online and is suitable to large-scale dynamic collaborative filtering scenario.

To further evaluate our approaches for online learning, Figure 1 shows more details. We observe that the curve of the DAOCF is always fluctuant instead of smooth. This is disadvantageous for stable output, because sometimes users will be recommended with item which involve severe and numerous errors. The curves of SOCFII and OCF are very similar. However, the former is slightly better than the latter.

5 Conclusion and Future Work

In this paper, we propose two approaches to deal with the user-item ratings for online collaborative filtering. We focus on the online matrix factorization problem which consists of learning the basis set of users in order to adapt it to online CF recommender systems. A user's interest and an item's popularity are always changing over a long period of time. So, recommendation approaches should take such changes into consideration.

Then, an empirical study has been conducted on two benchmark data sets, namely, *MovieLens100K* and *MovieLens1M.* These experimental results demonstrate that our online algorithm achieves more accuracy performance than other online algorithms while dramatically boosting efficiency. Our approaches are suitable to large-scale dynamic collaborative filtering scenario.

Now, many collaborative filtering algorithms are unable to capture the latest change of user preferences over time. In the future, we will focus our works on the improving of prediction accuracy and the accelerating of the speed of our approaches.

Acknowledgment

The Project was supported by the National Natural Science Foundation of China (No. 61402386, No. 61305061 and Grant No. 61402389). And we wish to thank the anonymous reviewers who helped to improve the quality of the paper.

Bibliography

[1] G. Linden, B. Smith, and J. York (2003), Amazon. com recommendations: Item-to-item collaborative filtering, *Internet Comput. IEEE*, 7(1): 76–80.

[2] M. D. Ekstrand, R. Join T., and K. Joseph A.(2011), Collaborative Filtering Recommender Systems, *Found. Trends® Human–Computer Interact.*, 4(2): 81–173.

[3] Z. Wang and H. Lu (2014), Online Recommender System Based on Social Network Regularization, *Neural Inf. Process.*, 487–494, Nov. 2014.

[4] J. B. Schafer, J. Konstan, and J. Riedl (1999), Recommender systems in e-commerce, *Electronic Commerce*, 158–166.

[5] K. Dohyun and Y. Bong Jin (2005), Collaborative filtering based on iterative principal component analysis, *Expert Syst. Appl.*, 28(4): 823–830.

[6] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl (2004), Evaluating collaborative filtering recommender systems, *ACM Trans. Inf. Syst. TOIS*, 22(1): 5–53, 2004.

[7] B. Sarwar, G. Karypis, J. Konstan, and J. Reidl (2001), Item-based collaborative filtering recommendation algorithms, *Proceedings of the 10th international conference on World Wide Web*, 285–295.

[8] J. A. Konstan, B. N. Miller, D. Maltz, J. L. Herlocker, L. R. Gordon, and J. Riedl (1997), GroupLens: applying collaborative filtering to Usenet news, *Commun. ACM*, 40(3): 77–87.

[9] J. Wang, S. C. H. Hoi, P. Zhao, and Z.-Y. Liu (2013), Online multi-task collaborative filtering for on-the-fly recommender systems, *Proceedings of the 7th ACM conference on Recommender systems, 2013*, 237–244.

[10] K. Yehuda and I. Haifa (2010), Collaborative filtering with temporal dynamics, *Commun. ACM*, 53(4): 89–97.

[11] J. Z. Kolter and M. Maloof (2003), Dynamic weighted majority: a new ensemble method for tracking concept drift, *ICDM 2003. Third IEEE International Conference on*, 123–130.

[12] D. Wang, P. Wu, P. Zhao, Y. Wu, C. Miao, and S. C. H. Hoi (2014), High-Dimensional Data Stream Classification via Sparse Online Learning, *Data Mining (ICDM), 2014 IEEE International Conference on*, 1007–1012.

[13] J. Abernethy, K. Canini, J. Langford, and A. Simma (2007), *Online collaborative filtering*, Univ. Calif. Berkeley Tech Rep, 2007.

[14] M. Ali, C. C. Campbell, and A. K. Tang (2011), Parallel Collaborative Filtering for Streaming Data, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.230.8613.

[15] J. Wilson, S. Chaudhury, B. Lall, and P. Kapadia (2014), Improving Collaborative Filtering based Recommenders using Topic Modelling, *Web Intelligence*, 340–346.

[16] T. Hofmann (2004), Latent semantic models for collaborative filtering, *ACM Trans. Inf. Syst.*, v 22(1): 89–115.

[17] R. Salakhutdinov, A. Mnih, and G. Hinton (2007), Restricted Boltzmann machines for collaborative filtering, *Proceedings of the 24th international conference on Machine learning*, 791–798.

[18] Li M; Wu C; Zhang L; You LN (2015), An Intuitionistic Fuzzy-Todim Method To Solve Distributor Evaluation And Selection Problem, *International Journal Of Simulation Modelling*, 14(3): 511-524.

[19] A. Mnih and R. Salakhutdinov (2007), Probabilistic matrix factorization, *Advances in neural information processing systems*, 1257–1264.

[20] G. Rainer, N. Nrik, H. Peter J., and S. Yannis (2011), Large-scale matrix factorization with distributed stochastic gradient descent, *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, 69–77.

[21] Saric T; Simunovic G; Simunovic K (2013), Use Of Neural Networks In Prediction And Simulation Of Steel Surface Roughness, *International Journal Of Simulation Modelling*, 12(4): 225-236.

[22] R. M. Bell, Y. Koren, and C. Volinsky (2007), The BellKor solution to the Netflix Prize, http://www.netflixprize.com/assets/ProgressPrize2007_KorBell.pdf, 1-15.

[23] J. Bennett and S. Lanning (2007), The Netflix Prize, *KDD Cup Workshop Conjunction KDD*, 2007.

[24] Z. Qiao, P. Zhang, J. He, Y. Cao, C. Zhou, and L. Guo (2014), Combining geographical information of users and content of items for accurate rating prediction, *Proceedings of the companion publication of the 23rd international conference on World wide web companion*, 361–362.

[25] W. Li and D. Yeung (2011), Social Relations Model for Collaborative Filtering, *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 803-808.

[26] Ocevcic Hrvoje; Nenadic Kresimir; Solic Kresimir (2014), Decision Support Based On The Risk Assessment Of Information Systems And Bayesian Learning, *Tehnicki Vjesnik-Technical Gazette*, 21(3): 539-544.

[27] Y. Koren, R. Bell, and C. Volinsky (2009), Matrix factorization techniques for recommender systems, *Computer*, 8: 30–37.

[28] M. Julien, B. Francis, P. Jean, and S. Guillermo (2010), Online Learning for Matrix Factorization and Sparse Coding, *J. Mach. Learn. Res.*, 11: 19–60.

[29] S. Shalev-Shwartz (2011), Online Learning and Online Convex Optimization, *Found. Trends Mach. Learn.*, 4(2): 107–194.

[30] R. Pálovics, A. A. Benczúr, L. Kocsis, T. Kiss, and E. Frigó (2014), Exploiting temporal influence in online recommendation, 273–280.

[31] G. Ling, H. Yang, I. King, and M. R. Lyu (2012), Online Learning for Collaborative Filtering, *IEEE World Congress on Computational Intelligence*, Brisbane, Australia, 1 – 8.