Online Multi-Task Collaborative Filtering for On-the-Fly Recommender Systems

Jialei Wang[*], Steven C.H. Hoi[*], Peilin Zhao[*], Zhi-Yong Liu[†]
[*]School of Computer Engineering, Nanyang Technological University, Singapore 639798
[†]State Key Lab of Management and Control for Complex System, Chinese Academy of Sciences, China
{jl.wang,chhoi,peilinzhao}@ntu.edu.sg, zhiyong.liu@ia.ac.cn

ABSTRACT

Traditional batch model-based Collaborative Filtering (CF) approaches typically assume a collection of users' rating data is given a priori for training the model. They suffer from a common yet critical drawback, i.e., the model has to be re-trained completely from scratch whenever new training data arrives, which is clearly non-scalable for large real recommender systems where users' rating data often arrives sequentially and frequently. In this paper, we investigate a novel efficient and scalable online collaborative filtering technique for on-the-fly recommender systems, which is able to effectively online update the recommendation model from a sequence of rating observations. Specifically, we propose a family of online multi-task collaborative filtering (OMTCF) algorithms, which tackle the online collaborative filtering task by exploiting the similar principle as online multitask learning. Encouraging empirical results on large-scale datasets showed that the proposed technique is significantly more effective than the state-of-the-art algorithms.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Filtering; I.2.6 [**Artificial Intelligence**]: Learning

Keywords

Recommender systems, Collaborative Filtering, Online learning, Multi-task Learning

1. INTRODUCTION

Collaborative filtering (CF) aims to make accurate predictions ("filtering") about the preferences of a user by learning/leveraging from a collection of preferences from many other users ("collaborative"). It is a core learning technique widely used in real-world recommender systems [30]. In literature, a variety of CF algorithms have been proposed, which can be generally grouped into two schools: (i) memory based methods [26, 20, 16, 11, 32, 13], and (ii) model based methods [27, 15, 31, 29, 23]. The model-based methods, typically powered by machine learning techniques, are more desired than the memory-based methods, especially when the amount of rating observations is relatively limited or sparse, which is common for real recommender systems.

Most existing model-based CF approaches assume a collection of users' rating data is given a priori to train the model in a batch learning fashion. Typically, the model has to be re-trained whenever there is new training data. Such approaches are impractical for real-world recommender systems where training data often arrive sequentially as new users are being added daily or even hourly, and new products/items are being offered dynamically. As a result, traditional batch learning methods are non-scalable due to their highly expensive re-training cost. This calls for an urgent need of efficient and scalable CF techniques for learning the model on-the-fly for real-world recommender systems.

To this end, we investigate Online Collaborative Filtering (OCF) techniques for on-the-fly recommender systems in this paper. The state-of-the-art OCF approach is based on the online low-rank matrix approximation algorithm [1, 17] by applying the simple online gradient descent (OGD) algorithms to solve the matrix factorization task, which thus avoids the highly expensive re-training cost of traditional batch matrix factorization algorithms. Despite their merit of high efficiency, their naive gradient descent approaches may not be very effective since they completely neglect the underlying structure of the collaborative filtering task.

To overcome the limitation of the existing OCF approaches, we propose a novel framework of Online Multi-Task Collaborative Filtering (OMTCF) by exploiting the close relationship between collaborative filtering and multi-task learning [22]. The key idea of OMTCF is to not only update the weight vectors of the user (task) related to the current observed data, but also the weight vectors of some other users (tasks) according to a user interaction matrix. As a result, OMTCF is able to learn the model more accurately than the existing OCF algorithms, with the only disadvantage of being slightly less efficient due to the cost of multi-task learning. In this paper, we propose a family of OMTCF algorithms to trade off between efficacy and efficiency, and extensively examine their performance for on-the-fly recommender systems on various large benchmark data sets.

The rest of the paper is organized as follows. Section 2 introduce the background and related work, Section 3 proposes the framework of Online Multi-Task Collaborative Filtering. Section 4 discusses our experimental results, and Section 5 sets out the conclusion of our work.

RecSys'13, October 12–16, 2013, Hong Kong, China.
Copyright 2013 ACM 978-1-4503-2409-0/13/10 ...$15.00.
http://dx.doi.org/10.1145/2507157.2507176 .

2. BACKGROUND AND RELATED WORK

This section briefly reviews the background of some major groups of related work, including batch collaborative filtering, online collaborative filtering, and multi-task learning.

One of the state-of-the-art methods for regular CF tasks is the latent factor or matrix factorization method [31, 23], which is one of the major techniques used by the Netflix prize winner's algorithms [14]. The key idea of latent factor model assumes that the similarity between users and items is simultaneously induced by some hidden lower-dimension structure in the data. For example, the rating that a user gives to a movie might depend on a few implicit factors such as the user's taste across various movie genres. Beyond this direction, some probabilistic and Bayesian matrix factorization for collaborative filtering methods have also been proposed [25, 19]. For a more comprehensive survey on regular CF techniques, please refer to the survey paper [30]. Although the batch algorithms for matrix factorization have shown great successes, they generally suffer from high time complexity and memory cost, which thus are non-scalable for building on-the-fly recommender systems.

Online collaborative filtering has received emerging attention recently. The work in [10] is perhaps one of early work, in which OCF is cast as an online ranking problem [8]. This algorithm is however not very practical because it assumes all the users' preferences can be known for each training instance, which is not always true. The related work most relevant to our study is the online collaborative filtering by stochastic gradient descent [1, 2] and another improved work in [17], which attempted to optimize the objective function of matrix factorization based formulation for collaborative filtering in online learning fashion. Their naive gradient descent strategy simply neglects the underlying structure of CF tasks which thus limits their learning efficacy. Our work is partially motivated to overcome their limitation. Besides, our work is also somewhat related to the online evolutionary collaborative filtering work in [18] which assigns larger weights to the new rating and incrementally updates the users' and items' similarities, but differs in two major aspects: (i) to be more precise from an online learning perspective, [18] is essentially a mini-batch or incremental algorithm that does not update the model upon a single observation; and (ii) [18] generally belongs to a simple memory-based approach, which is usually less effective as compared to the state-of-the-art matrix factorization approaches. Finally, our work is very different from the work [9] which tried to improve scalability of batch CF methods by exploring practical tricks, such as the MapReduce framework.

Our work is also inspired in part by some related work in Multi-task learning (MTL) [3], which is a machine learning method where multiple tasks are jointly learnt such that each of them benefits from each other. A popular framework for MTL is multi-task feature learning [4], which assumes that all tasks share a common yet latent feature representation. Similarly, although MTL has many advantages, it also suffers some disadvantages, e.g., low efficiency. Recently, online multi-task learning [7, 24] has been proposed to extend traditional MTL in online learning setting. The online multi-task learner proceeds in rounds by observing a sequence of examples, each belonging to some task from a pre-defined set tasks. The basic idea of the online multi-task learning in [7] is that instead of only updating the weight vector of the task related to the current instance, it online updates weight vectors of multiple tasks based on a task interaction matrix. Furthermore, instead of using a fixed task interaction matrix, [24] proposed to update the task relationship matrix during the online learning process.

Although both online collaborative filtering and online multi-task learning have been actively studied in different research communities separately, to the best of our knowledge, no existing work has attempted to tackle online collaborative filtering by exploiting the idea of online multi-task learning for building on-the-fly recommender systems.

3. ONLINE MULTI-TASK COLLABORATIVE FILTERING

3.1 Problem Setting and Formulation

In this section, we present the proposed framework for Online Multi-Task Collaborative Filtering (OMTCF) for building on-the-fly recommender systems. We emphasize that an essential difference between online CF and traditional batch CF tasks is that batch CF assumes the entire collection of training data is given a prior before the learning task, while online CF learns the model on-the-fly from a sequence of training data. In the following, we will firstly introduce the problem setting and then present a family of OMTCF algorithms for addressing a variety of practical issues.

First of all, we introduce the problem of a regular collaborative filtering task. In a collaborative filtering task, some users from a total of n users rated some products (items) from a total of m products, and these ratings form an incomplete matrix $\mathrm{R} \in \mathbb{R}^{n\times m}$, where r_{ij} is the rating on the j-th item given by the i-th user. The goal of collaborative filtering is to predict the unknown ratings based on the known ones. For collaborative filtering, one of the well-known approach is the matrix factorization algorithm, which learns the latent structure by factorizing the rating matrix to a user matrix $\mathrm{U} \in \mathbb{R}^{n\times k}$ and an item matrix $\mathrm{V} \in \mathbb{R}^{m\times k}$ through the following optimization problem:

$$\underset{\mathrm{U}\in\mathbb{R}^{n\times k},\mathrm{V}\in\mathbb{R}^{m\times k}}{\arg\min} \|\mathrm{R} - \mathrm{U}\mathrm{V}^\top\|_F^2, \tag{1}$$

where $\|\mathrm{R}\|_F$ is the Frobenius norm of the matrix R. If the rating matrix R is fully observable, then this problem can be reduced to a Singular Value Decomposition (SVD) problem. However, the matrix for collaborative filtering is only partially observed, which results in an ill-posed problem for the above formulation. To tackle the challenge, we can apply the low-rank matrix approximation technique to re-formulate this problem. Specifically, let U_a be the a-th row of U, and V_b be the b-th row of V, then given a partial collection of observed pairs $\mathcal{C} = \{(a,b)\}$, let $|\mathcal{C}|$ denote the number of observed ratings, the collaborative filtering problem can be re-formulated as the regularized optimization task for low-rank matrix factorization:

$$\underset{\mathrm{U}\in\mathbb{R}^{n\times k},\mathrm{V}\in\mathbb{R}^{m\times k}}{\arg\min} \sum_{\forall(a,b)\in\mathcal{C}} \ell(U_a, V_b, r_{a,b}) + \lambda(\|\mathrm{U}\|_F^2 + \|\mathrm{V}\|_F^2) \tag{2}$$

where $\lambda > 0$ is a regularization parameter, and $\ell(U_a, V_b, r_{a,b})$ is a loss function that defines the loss between the true $r_{a,b}$ and the prediction $U_a V_b^\top$.

However, this new problem is a non-convex optimization which cannot be solved directly using a standard SVD algorithm. Since the optimization problem (2) is non-convex, [28] proposed to replace the Frobenius norm regularization by a trace norm regularization in order to relax it to a convex optimization problem, which can be solved by Semi-definite Programming (SDP) using the following property of trace norm regularization:

$$\|\mathrm{R}\|_{\Sigma} = \inf_{\mathrm{R}=\mathrm{U}\mathrm{V}^{\top}} \frac{1}{2}(\|\mathrm{U}\|_F^2 + \|\mathrm{V}\|_F^2). \tag{3}$$

However, the heavy cost of solving a sparse SDP task makes it infeasible for real datasets with millions of observations.

To overcome this limitation of regular collaborative filtering, [1] proposed the online collaborative filtering based on a simple gradient descent approach. In particular, given a single prediction problem of user a on item b, the algorithm first makes a prediction of rating $\hat{r}_{a,b} = U_a V_b^{\top}$; after the true rating $r_{a,b}$ is revealed, the algorithm suffers a loss $\ell(U_a, V_b, r_{a,b})$; finally, the OCF algorithm makes updates on their model U and V based on the gradient descent approach:

$$U_a = (1 - 2\tau\lambda)U_a - \tau\frac{\partial\ell(U_a, V_b, r_{a,b})}{\partial U_a}, \tag{4}$$

$$V_b = (1 - 2\tau\lambda)V_b - \tau\frac{\partial\ell(U_a, V_b, r_{a,b})}{\partial V_b}. \tag{5}$$

where τ is the learning rate parameter. In general, one can define different type of loss function for different purposes. For example, to optimize the Root Mean Square Error (RMSE), i.e., RMSE $= \sqrt{\frac{1}{|\mathcal{C}|}\sum_{(a,b)\in\mathcal{C}}(r_{a,b} - \hat{r}_{a,b})^2}$, we can define the loss by the square error function as:

$$\ell_1(U_a, V_b, r_{a,b}) = (r_{a,b} - U_a V_b^{\top})^2 \tag{6}$$

Similarly, if one aims to optimize the Mean Absolute Error (MAE), i.e., MAE $= \frac{1}{|\mathcal{C}|}\sum_{(a,b)\in\mathcal{C}}|r_{a,b} - \hat{r}_{a,b}|$, we can define the absolute loss function as:

$$\ell_2(U_a, V_b, r_{a,b}) = |r_{a,b} - U_a V_b^{\top}| \tag{7}$$

3.2 A Family of OMTCF Algorithms

The proposed OMTCF method is generally inspired by noticing the equivalence between multi-task learning and collaborative filtering formulations. To illustrate this clearly, we rewrite the formulation of the optimization task of collaborative filtering as follows:

$$\operatorname*{arg\,min}_{\mathrm{U}\in\mathbb{R}^{n\times k}, \mathrm{V}\in\mathbb{R}^{m\times k}} \sum_{i=1}^{n}\sum_{\forall(i,b)\in\mathcal{C}} \ell(U_i, V_b, r_{i,b}) + \lambda(\|\mathrm{U}\|_F^2 + \|\mathrm{V}\|_F^2) \tag{8}$$

By treating each user i as an individual task, each U_i as the task model, V as the features of items and $r_{i,b}$ as the prediction output of model i for item b, it is not difficult to see that the above formulation is essentially equivalent to the formulation of a multi-task learning that optimizes the models of multiple tasks simultaneously. The equivalence between collaborative filtering and multi-task learning has also been shown by some previous studies in literature [22, 21], which partially inspire our study in this paper.

Motivated by the above equivalence fact, our approach is to exploit the idea of online multi-task learning [7] to tackle the online collaborative filtering task. For online multi-task learning, one important step is to define a task interaction matrix $\mathrm{A} \in \mathbb{R}^{n\times n}$, which essentially indicates the similarity or interaction degree between tasks. Following the similar idea, we can define a user interaction matrix $\mathrm{A} \in \mathbb{R}^{n\times n}$ such as a large value $A_{i,j}$ indicates a close interaction should be imposed between user i and user j. In general, matrix A can be computed if prior knowledge about the users is available. If however such prior knowledge is not available, one can define some constant matrix. For example, one simple approach used by the previous study is to define A as follows:

$$\mathrm{A}_{fixed} = \begin{bmatrix} 1 & \frac{1}{2} & \dots & \frac{1}{2} \\ \frac{1}{2} & 1 & \dots & \frac{1}{2} \\ \dots & \dots & \dots & \dots \\ \frac{1}{2} & \frac{1}{2} & \dots & 1 \end{bmatrix}. \tag{9}$$

where the above matrix indicates that we should make a full update ($A_{ii} = 1$) of the user's model when the instance belongs to, and make a half update ($A_{ij} = \frac{1}{2}\forall i \neq j$) of other users' models otherwise.

Unlike the existing OCF approach where only one user is updated for every observed rating, the proposed online multi-task collaborative filtering (OMTCF) approach will update multiple users for each observed rating. More formally, given an incoming rating $r_{a,b}$ with respect to user a and item b, the algorithm will update the models of multiple users (i.e., multiple rows of U) according to the user interaction matrix A. Specifically, we have the following functions for updating the models of multiple users:

$$U_i = (1 - 2\tau\lambda)U_i - \tau A(a,i)\frac{\partial\ell(U_a, V_b, r_{a,b})}{\partial U_a}, \tag{10}$$

$$V_b = (1 - 2\tau\lambda)V_b - \tau\frac{\partial\ell(U_a, V_b, r_{a,b})}{\partial V_b}, \tag{11}$$

where $i = 1, \dots, n$, τ is the learning rate and $\ell(U_a, V_b, r_{a,b})$ is a loss function which is defined based on either Eq. (6) for RMSE or Eq. (7) for MAE. In the above, a larger value of $A(a,i)$ will result in a stronger update on U_i eventually. With respect to the two kinds of loss functions, we derive the detailed online updating rules by gradient descent associated with the interaction matrix in the following propositions.

PROPOSITION 1. *In an OMTCF task, given an observed rating pair (a, b), the updating rule with respect to the loss function ℓ_1 defined in Eq. (6) can be expressed as follows:*

$$U_i = (1 - 2\tau\lambda)U_i + 2\tau A_t(a,i)(V_b(r_{a,b} - U_a V_b^{\top})) \tag{12}$$

$$V_b = (1 - 2\tau\lambda)V_b + 2\tau U_a(r_{a,b} - U_a V_b^{\top}) \tag{13}$$

where $i = 1, \dots, n$.

PROPOSITION 2. *In an OMTCF task, given an observed rating pair (a, b), the updating rule with respect to the loss function ℓ_2 defined in Eq. (7) can be expressed as follows:*

$$U_i = (1 - 2\tau\lambda)U_i + 2\tau A_t(a,i)(V_b \cdot sign(r_{a,b} - U_a V_b^{\top})) \tag{14}$$

$$V_b = (1 - 2\tau\lambda)V_b + 2\tau U_a \cdot sign(r_{a,b} - U_a V_b^{\top}) \tag{15}$$

where $i = 1, \dots, n$.

In the proposed OMTCF algorithms, one key issue is how to determine an appropriate user interaction matrix A. The following will discuss this issue in detail.

3.3 Online Updating User Interaction Matrix

Instead of choosing a constant user interaction matrix A as shown in Eqn (9), another possible approach is to calculate a (sparse) matrix A from prior knowledge of user information when available:

$$A(i,j) = \text{similarity}(i,j) \tag{16}$$

where similarity(i,j) is a function of defining the similarity between two users according to the prior knowledge. For example, if we only have the age information about users, we can probably set the similarity value for the users who belongs to the same age group as 1, and 0 otherwise.

The above approaches are generally limited in that they either require the prior knowledge which is not always available or often fix the user interaction matrix as a constant matrix, which may somewhat restrict the power of the proposed algorithm for achieving faster convergence. To overcome these limitations, we also propose to learn the user interaction matrix A automatically during the online learning process. In particular, we can choose the user interaction matrix by using the covariance matrix of user matrix U that is updated sequentially as follows:

$$\mathrm{A} = cov(\mathrm{U}), \tag{17}$$

which is somewhat inspired by the Gaussian Process based multi-task learning [5], where the task relatedness is measured by the Gaussian Process covariance function.

Although the above suggested matrices are fairly reasonable, in a real-world scenario, they are only an approximation of the user similarity, which may not exactly reflect the true user interaction. In addition, online multi-task learning often converges much faster than online single task learning at the beginning of the online learning process, but the advantage of performing multi-task updates will become less significant as time goes by as the models become more and more accurate. To address these issues, we propose to define the following attenuation coefficient function over the user relationship matrix:

$$\theta_t(i,j) = (\frac{1}{t})^{\frac{1}{3}}\mathbb{I}_{i\neq j} + \mathbb{I}_{i=j} \tag{18}$$

where $\mathbb{I}_x$ is an indicator function that outputs 1 when x is true and 0 otherwise. This attenuation function weakens the effect of updating the multiple users as time goes. Thus, the final true user interaction matrix used in OMTCF will be the user interaction matrix multiplied by the attenuation coefficient as:

$$\mathrm{A}_t = \theta_t \odot \mathrm{A}, \tag{19}$$

where $\odot$ is the element-wise produce, i.e., $A_t(i,j) = \theta_t(i,j) * A(i,j)$. Finally, Algorithm 1 summarizes the framework of the proposed OMTCF algorithms. It is not difficult to see that the proposed algorithms are efficient and scalable with linear time and space complexity.

Algorithm 1 OMTCF — Framework of Online Multi-Task Collaborative Filtering algorithms

```
**Input**: a sequence of rating pairs $\{(a_t, b_t), t = 1, \dots, M\}$
**Initialization**: Initialize a random matrix for user matrix U and item matrix V, and initialize user relationship matrix $\mathrm{A}_0$ as an identity matrix I
**for** $t = 1, 2, \dots, M$ **do**
  receive rating prediction request of user $a_t$ on item $b_t$
  make prediction $\hat{r}_{a_t,b_t} = U_{a_t}V_{b_t}^\top$
  the true rating $r_{a_t,b_t}$ is revealed
  the algorithm suffers a loss $\ell(U_a, V_b, r_{a,b})$
  update U and $V_{b_t}$ by Proposition 1(RMSE) or 2(MAE)
  update the user interaction matrix $\mathrm{A}_t$ as follows:
    $$\mathrm{A}_t = \theta_t \odot \begin{cases} \mathrm{A}_{fixed} & \text{(OMTCF-I)} \\ \mathrm{A}_{\text{similarity}} & \text{(OMTCF-II)} \\ cov(\mathrm{U}) & \text{(OMTCF-III)} \end{cases}$$
**end for**
```

3.4 Efficient Hybrid Algorithm for OMTCF

Although the proposed OMTCF algorithms achieve significantly faster convergence rate than the traditional OCF approach, this advantage is paid by higher computation cost of performing the multi-task updates. Specifically, because OMTCF needs to update the models of multiple users during the online learning step, the running time cost would be much higher than OCF. In this part, we would propose a hybrid algorithm to trade off between efficiency and efficacy.

The proposed algorithm follows the similar framework as the OMTCF-I algorithm. In the online learning process, we monitor the amount of prediction change (in either RMSE or MAE) over a past certain period (defined by a window size T) $\Delta(t,T)$ defined as follows:

$$\Delta(t,T) = \begin{cases} \mathrm{RMSE}(t-T) - \mathrm{RMSE}(t) & \text{(RMSE)} \\ \mathrm{MAE}(t-T) - \mathrm{MAE}(t) & \text{(MAE)} \end{cases}$$

We then introduce a threshold ζ to indicate whether the learning process has converged sufficiently. If the change $\Delta(t,T)$ is smaller than ζ, the algorithm will then switch to a single-task update, i.e.,

$$\mathrm{A}_t = \begin{cases} \theta_t \odot \mathrm{A}_{fixed} & \Delta(t,T) \geq \zeta \\ \mathrm{I} & \text{otherwise} \end{cases}$$

We denote this hybrid algorithm as "OMTCF-IV" for short.

3.5 OMTCF for Novel User/Item Extension

The above OMTCF algorithms follow a formal online learning setting, but they make an implicit assumption, i.e., both the number of users and the number of items are fixed and they are both known beforehand. In a real-world online application, this is not realistic. In this part, we aim to extend OMTCF algorithms to handle the situation where an incoming rating observation is related to a novel user or a novel item. The key idea of the proposed algorithm here is that we will try to expand the user/item matrix by adding one user/item vector whenever a new user/item appears. We assume the user interaction matrix is fixed as Eq.(9), which is the same as "OMTCF-I" (we could also consider other type of interaction matrix). We keep a multi-task cumulative updated user vector and update it at each iteration. This multi-task cumulative updated user vector will be applied to initialize the new user vector whenever a new user is added. This strategy is able to guarantee the algorithm achieves the same performance as the "OMTCF-I" without considering the difference caused by different random initializations. We denote this algorithm for handling novel sample extension as "OMTCF-V" for short. The details are summarized in Algorithm 2.

3.6 Practical Online Multi-task Collaborative Filtering for Large-Scale Datasets

As we can see, algorithm "OMTCF-V" can build the user and item matrix from scratch and online extended the user and item matrix based on the data observed, which could save the memory and make the algorithm more efficient. The algorithm "OMTCF-IV" can avoid unnecessary multiple updates when the online learning process has converged well. In fact, we can adopt the two improvements over original online multi-task collaborative filtering algorithm simultaneously into a single new algorithm, which is able to make online multi-task learning strategy for collaborative filtering more practical and efficient for large-scale datasets. We

Algorithm 2 OMTCF-V: The proposed OMTCF algorithm for handling novel sample extension

```
**Input**: a sequence of rating pairs $\{(a_t, b_t), t = 1, \ldots, M\}$, window size $T$ (e.g. $T$=1000), convergence threshold $\zeta$
**Initialization**: U = V = A = [] and randomize $U_{ini}$.
**for** $t = 1, 2, \ldots, M$ **do**
  receive rating prediction request of user $a_t$ on item $b_t$
  **if** user $a_t$ is new **then**
    initialize $U_{a_t} = U_{ini}$
    expand the user matrix U as $\mathrm{U} = [\mathrm{U}; U_{a_t}]$
    expand the user interaction matrix as:
    $\mathrm{A}_{t-1} \leftarrow [\mathrm{A}_{t-1}, \frac{1}{2}(\frac{1}{t})^{\frac{1}{3}}\mathbf{1}; \frac{1}{2}(\frac{1}{t})^{\frac{1}{3}}\mathbf{1}, 1]$, where $\mathbf{1}$ is a vector of all elements equal to 1
  **end if**
  **if** item $b_t$ is new **then**
    initialize $V_{b_t}$ as a random vector.
    expand the item matrix V as $\mathrm{V} = [\mathrm{V}; V_{b_t}]$
  **end if**
  make prediction $\hat{r}_{a_t,b_t} = U_{a_t} V_{b_t}^\top$
  the true rating $r_{a_t,b_t}$ is revealed
  the algorithm suffers a loss $\ell(U_a, V_b, r_{a,b})$
  update U and $V_{b_t}$ by Proposition 1(RMSE) or 2(MAE)
  calculate $U_{ini}$ as
  $\begin{cases} (1 - 2\tau\lambda)U_{ini} + \tau(\frac{1}{t})^{\frac{1}{3}}(V_b(r_{a,b} - U_a V_b^\top)) & \text{(RMSE)} \\ (1 - 2\tau\lambda)U_{ini} + \tau(\frac{1}{t})^{\frac{1}{3}}(V_b sign(r_{a,b} - U_a V_b^\top)) & \text{(MAE)} \end{cases}$
  update $\mathrm{A}_t$ by $\mathrm{A}_t = \theta_t \odot \mathrm{A}_{fixed}$
**end for**
```

name the new algorithm as "OMTCF-VI", which is a combination of OMTCF-IV and OMTCF-V. We omit the detailed algorithm due to space limitation.

Remarks on Parallelization. One important advantage of online CF over batch CF is that online algorithms only need to process one rating at a time, making the updating efficient and scalable. Although our OMTCF algorithms are generally more computationally expensive than the simple OCF algorithm in [1], this disadvantage can be somewhat compensated by exploring the advantages of parallel computing techniques, which can be easily deployed by the proposed OMTCF algorithms. In particular, in contrast to the OCF algorithm, our OMTCF algorithms update multiple users simultaneously and the updates of these users do not affect each other at each learning round, making them to be easily parallelized by a trivial implementation.

4. EXPERIMENTAL RESULTS

In this section, we evaluate the empirical performance of the proposed OMTCF algorithms for online collaborative filtering tasks. It is important to note that we do not compare the proposed algorithms with other existing batch collaborative filtering algorithms since the performance evaluation protocol and settings between online and batch learning algorithms are very different, making them unfair and meaningless to be compared directly.

4.1 Compared Algorithms

We compare the proposed OMTCF algorithms with the emerging online collaborative filtering algorithm. Specifically, the compared algorithms in our experiments include:

- "OCF": the Online Collaborative Filtering algorithm by online gradient descent method described in [1];
- "DA-OCF" the Dual-Averaging method for online collaborative filtering proposed in [17];
- "OMTCF-I": the proposed OMTCF in Algorithm 1 where the user-interaction matrix is set as Eq.(9) as suggested in [7];
- "OMTCF-II": the proposed OMTCF algorithm with one sparse user interaction matrix is computed using prior user information in Eq.(16);
- "OMTCF-III": the proposed OMTCF algorithm with the task covariance matrix in Eq.(17);
- "OMTCF-IV": the proposed Hybrid OMTCF algorithm;
- "OMTCF-V": the proposed OMTCF for new sample extension algorithm, as shown in Algorithm 2.
- "OMTCF-VI": our practical OMTCF algorithm.

4.2 Experimental Testbed and Setup

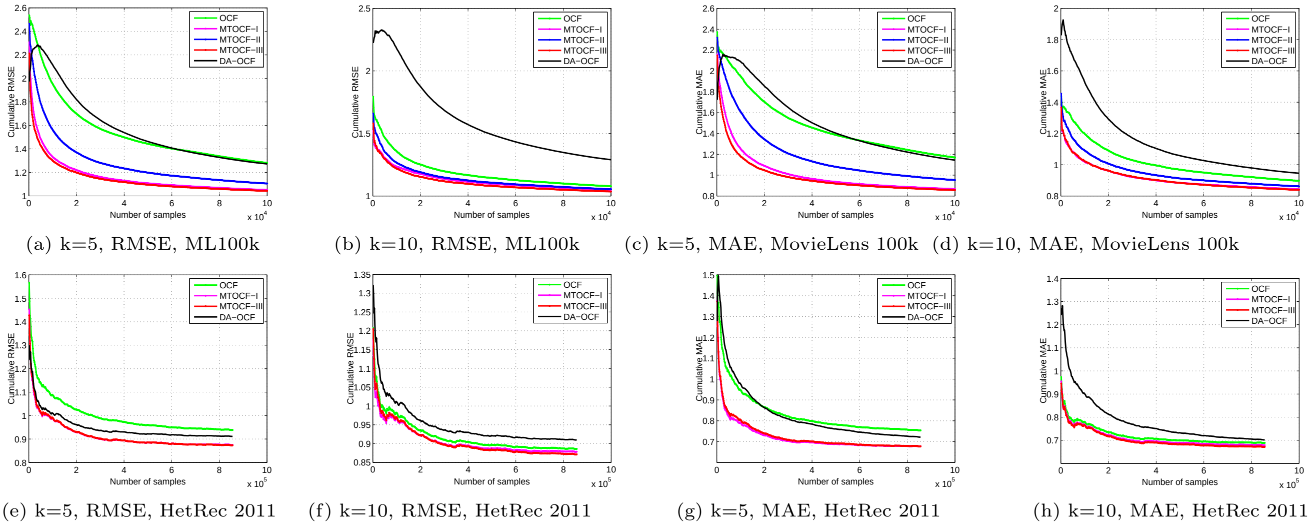
To extensively examine the empirical performance, we conduct the experiments on a variety of publicly available datasets widely used for benchmark evaluation of CF in literature. The first two relatively smaller datasets include: (i) the "MovieLens 100k" dataset collected by the Grouplens Research Project from the MovieLens web site [1], where users can freely give their ratings on various movies and receive movie recommendations; it consists of 100,000 ratings from 943 users on 1682 movies, and also provides additional user info, such as their genders, and ages, etc; (ii) the "HetRec 2011 MovieLens" dataset, which links the movies of MovieLens with their corresponding web pages at IMDb and Rotten Tomatoes movie review systems; this dataset consists of 855,598 ratings from 2113 users on 10197 movies; it however does not contain any user info. We also test on other four large-scale data sets which will be introduced later.

To make fair comparisons, all the algorithms adopt the same experimental setup. Firstly, for the proposed "OMTCF-II" algorithm, we set the similarity between a pair of users as 1 when they are from the same gender and in the same age period, while the similarities of others pairs as 0. Note that two users are considered as in the same age period if the difference of their ages is less than 5 years. Considering the fact that only the "MovieLens 100k" dataset consists of the user information, the algorithm "OMTCF-II" is only tested on this dataset. Secondly, for the proposed "OMTCF-III" algorithm, we use the gaussian kernel for the users vectors to compute the user similarity, where the kernel width σ is set as 1. To make a fair comparison, the learning rate γ of all algorithms is set to 0.005, and the regularization parameter is set to 3^{-6}, the λ parameter in DA-OCF algorithm was set to 0.006, which was suggested to achieve the best performance according to [17]. The rank parameter k of matrix U and V is set to two cases: 5 and 10, respectively. After the parameters are chosen, all the experiments were conducted over 20 runs of different random permutations for each dataset. All the experimental results were reported by averaging over these 20 runs. For performance metric, we evaluate the performance of online collaborative filtering algorithms by measuring their scores of online Root Mean Square Error (RMSE) and online Mean Absolute Error (MAE) on the test set.

4.3 Evaluation on Two Smaller Datasets

Table 1 summarizes the average performance of the compared algorithms over the two datasets. From the experimental results, several observations can be drawn as follows.

[1] http://movielens.umn.edu



(a) k=5, RMSE, ML100k (b) k=10, RMSE, ML100k (c) k=5, MAE, MovieLens 100k (d) k=10, MAE, MovieLens 100k

(e) k=5, RMSE, HetRec 2011 (f) k=10, RMSE, HetRec 2011 (g) k=5, MAE, HetRec 2011 (h) k=10, MAE, HetRec 2011

Figure 1: Online cumulative predictive performance of different online collaborative filtering algorithms for on-the-fly recommendation (best viewed in color).

Table 1: Evaluation of online CF algorithms, where k is the rank parameter for matrix U and V. The bold element indicates the best performance for each setting. Note that OMTCF-II is not applicable to the second dataset as user info is unavailable.

MovieLens 100k	k=5		k=10	
	RMSE	MAE	RMSE	MAE
OCF	1.2808	1.1685	1.0781	0.8970
DA-OCF	1.2722	1.1438	1.2898	0.9369
OMTCF-I	1.0509	0.8665	1.0506	0.8434
OMTCF-II	1.1057	0.9527	1.0543	0.8614
OMTCF-III	**1.0429**	**0.8564**	**1.0359**	**0.8393**
HetRec 2011 MovieLens	k=5		k=10	
	RMSE	MAE	RMSE	MAE
OCF	0.9386	0.7540	0.8859	0.6887
DA-OCF	0.9114	0.7273	0.9098	0.7018
OMTCF-I	0.8750	**0.6777**	0.8784	0.6786
OMTCF-III	**0.8734**	**0.6777**	**0.8715**	**0.6716**

First, compared with the previous OCF and DA-OCF approach, we observe that all the proposed OMTCF algorithms achieve significantly better performance of smaller RMSE/MSE values for all the cases. This shows that the proposed learning strategy is more effective than the existing naive gradient descent approach in improving the online prediction performance of collaborative filtering tasks.

Second, we found that, when the dimensionality is relatively lower (i.e., k=5), the gap between the OMTCF and OCF tends to be more significant. This shows that OMTCF is more suitable to learn an effective low-rank matrix factorization for collaborative filtering tasks. DA-OCF seems dose not benefit much as the dimensionality increases.

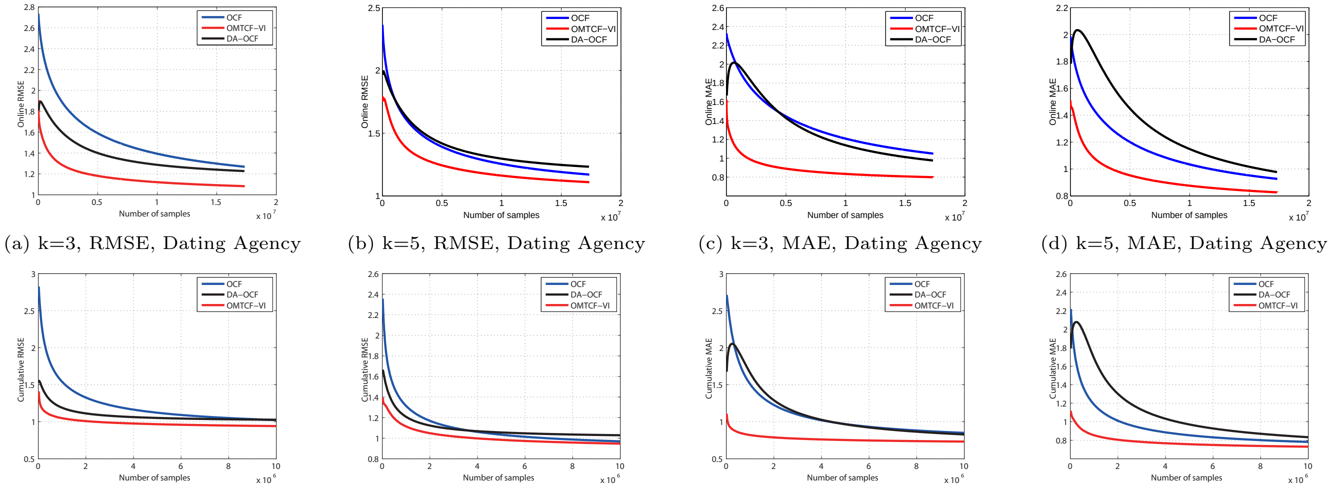
Third, by comparing the performance between several variants of OMTCF algorithms using different user interaction matrices, we found that OMTCF-III performs the best among all the proposed algorithms. This validates the efficacy of learning the user interaction matrix in online collaborative filtering learning tasks. Further, OMTCF-II tends to perform worse than the other OMTCF algorithms, which indicates that it should be careful to exploit the user information for user interaction matrix, especially when such user information is noisy or incomplete in real applications. For such situations, a simple user interaction matrix, e.g., defined in Eq.(9), might be even better than some complicated user interaction matrix which may be corrupted with noise.

Besides, in order to inspect the details of online collaborative filtering performance, we also show the online performance convergence of all the compared algorithms in the entire online learning process in Figure 1. Similar to the above observations, the experimental results in the figure again verify the efficacy of the proposed OMTCF algorithms, which consistently outperform the previous OCF algorithm. Specifically, we observe that the OMTCF algorithms in general converge much faster than OCF in the beginning of the learning process, which is very important and beneficial to many recommender systems as a fast convergence performance makes the online recommendation solution being able to effectively adapt the fast changes in a real online recommendation environment.

4.4 Efficiency and New Sample Extension

Although the OMTCF algorithms are able to significantly improve the online RMSE/MAE performance, they generally suffer high time complexity. The time efficiency of an online collaborative filtering algorithm is also very important for large-scale applications. To overcome this limitation of the previous OMTCF algorithms, we also evaluate the efficient hybrid algorithm, which aims to achieve a trade off between efficacy and efficiency. In particular, it will reduce to a single-task updating algorithm when the online RMSE or MAE performance is no longer improved significantly. In addition to improving the efficiency issue of OMTCF, we also propose OMTCF-V to tackle the new sample extension issue, which is also evaluated in the following.

Specifically, we compare all the OMTCF algorithms with OCF in terms of both predictive performance (RMSE/MAE) and time efficiency. For OMTCF-I,II,III algorithms, the parameters are set as the same as the previous experiment. For OMTCF-IV, the parameters are set the same as OMTCF-I, except that ζ is set to 5×10^{-2} for RMSE and set to 3×10^{-2} for MAE, the rank parameter k is set to 5 for all algorithms.



(a) k=3, RMSE, Dating Agency (b) k=5, RMSE, Dating Agency (c) k=3, MAE, Dating Agency (d) k=5, MAE, Dating Agency

(e) k=3, RMSE, MovieLens 10M (f) k=5, RMSE, MovieLens 10M (g) k=3, MAE, MovieLens 10M (h) k=5, MAE, MovieLens 10M

Figure 2: Online cumulative predictive performance of three online algorithms for on-the-fly recommendation on large-scale datasets (best viewed in color). We can only show two largest datasets due to space limitation.

Table 2: Evaluation of Efficacy and Time Cost

MovieLens 100k	RMSE		MAE	
	RMSE	TIME(s)	MAE	TIME(s)
OCF	1.2779	0.76	1.1890	0.68
DA-OCF	1.2722	2.01	1.1438	2.06
OMTCF-I	1.0534	102.41	0.8644	104.08
OMTCF-II	1.1103	23.81	0.9514	24.03
OMTCF-III	1.0443	5883.83	0.8514	5732.84
OMTCF-IV	1.0607	2.19	0.8967	2.99
OMTCF-V	1.0532	88.61	0.8648	91.66
OMTCF-VI	1.0521	1.34	0.8625	1.96
HetRec 2011 MovieLens	RMSE		MAE	
	RMSE	TIME(s)	MAE	TIME(s)
OCF	0.9408	7.79	0.7560	5.99
DA-OCF	0.9114	17.08	0.7273	16.93
OMTCF-I	0.8757	1964.89	0.6786	1992.89
OMTCF-III	0.8719	2.3×10^6	0.6785	2.3×10^6
OMTCF-IV	0.8794	22.27	0.6818	33.85
OMTCF-V	0.8767	1126.62	0.6779	1159.84
OMTCF-VI	0.8757	12.78	0.6713	16.57

The parameters for OMTCF-V is the same as OMTCF-IV. After choosing the parameters, all the experiments were conducted over 20 runs of different random permutations for each dataset. All the experimental results were reported by averaging over these 20 runs. The performance evaluations are summarized in Table 2.

Several observations can be drawn from the results. First, we found that OMTCF-III, though achieving the best RMSE and MAE performance, runs significantly slower than the other algorithms, indicating the importance of improving efficiency of the proposed algorithms. Further, we can see that OMTCF-IV achieves significantly smaller values of both RMSE and MAE than OCF, which demonstrates the strategy of the proposed OMTCF-IV algorithm is significantly more effective than the state-of-the-art OCF algorithm. In addition, the OMTCF-IV algorithm is slightly worse than the OMTCF-I algorithm in terms of RMSE and MAE values, but the difference is very small. In addition, according to the time evaluation results, OMTCF-IV runs slightly slower than the OCF algorithm, but significantly faster than OMTCF-I. From these promising results, we can conclude that the proposed OMTCF-IV algorithm empirically achieves fairly comparable RMSE/MAE performance as OMTCF-I, and shares quite similar empirical time complexity as OCF, which indicates the OMTCF-IV algorithm in general achieves a good trade-off between efficacy and efficiency.

Finally, we examine the efficacy of the proposed OMTCF-V algorithm for handling new sample extension issue. In particular, by inspecting the performance of the OMTCF-V algorithm, we observe that OMTCF-V achieves almost identical values of RMSE and MAE as compared with the OMTCF-I algorithm (the marginal differences should be caused by different randomization for the user and item vectors), but save much time cost than the OMTCF-I algorithm, which demonstrates that the proposed OMTCF-V algorithm is not only capable of handling the new sample extension issue perfectly, but also runs more efficiently than the previous OMTCF algorithms. Among all the OMTCF algorithms, OMTCF-VI is the most efficient one and has comparable RMSE/MAE performance with other algorithms, making it more applicable for large-scale applications.

4.5 Evaluation on Large-Scale Datasets

We now evaluate the practical OMTCF-VI algorithm on four large-scale datasets: (i) **Dating Agency [6]:** The Dating Agency dataset[2], which contains 17,359,346 anonymous ratings of 168,791 profiles made by 135,359 LibimSeTi users; (ii) **Jester Joke:** this dataset from online Joke recommendation system[3] contains 1,761,439 ratings of 150 jokes from 63,974 users; (iii)**Movielens 1M:** this Movie rating dataset[4] contains 1,000,209 ratings of 3,900 movies by 6,040 users; and (iv) **Movielens 10M:** this Movie rating dataset[5] contains 10,000,054 ratings of 10,681 movies by 71,567 users.

Table 3 and Figure 2 summarize the empirical results of our evaluation. We can observe that OMTCF-VI performs significantly better than the OCF algorithm in terms of both RMSE and MAE metrics. In terms of the time efficiency, the online algorithms (OCF and OMTCF-VI) are generally fairly efficient, which typically took several hundreds sec-

[2] `http://www.occamslab.com/petricek/data/`
[3] `http://goldberg.berkeley.edu/jester-data/`
[4] `http://movielens.umn.edu`
[5] `http://movielens.umn.edu`

onds to run on a dataset with 10,000,000 ratings. Finally, among all the online algorithms, the proposed OMTCF-VI algorithm is on average just 2-3 times slower than the OCF algorithm, and slightly slower than or sometimes comparable to the DA-OCF algorithm.

Table 3: Evaluation on Large-scale datasets.

Dating Agency	RMSE, k = 3		RMSE, k = 5	
	RMSE	TIME(s)	RMSE	TIME(s)
OCF	1.2684	183.85	1.1700	189.86
DA-OCF	1.2269	342.15	1.2324	340.18
OMTCF-VI	**1.0823**	583.21	**1.1094**	563.54
Dating Agency	MAE, k = 3		MAE, k = 5	
	MAE	TIME(s)	MAE	TIME(s)
OCF	1.0494	189.69	0.9265	192.68
DA-OCF	0.9757	393.43	0.9767	405.70
OMTCF-VI	**0.7986**	520.34	**0.8263**	626.79
Jester Joke	RMSE, k = 3		RMSE, k = 5	
	RMSE	TIME(s)	RMSE	TIME(s)
OCF	1.1243	18.56	1.1287	18.50
DA-OCF	1.1305	34.77	1.1279	34.82
OMTCF-VI	**1.1195**	30.10	**1.1261**	29.33
Jester Joke	MAE, k = 3		MAE, k = 5	
	MAE	TIME(s)	MAE	TIME(s)
OCF	0.8866	18.64	0.8896	18.81
DA-OCF	0.8977	34.54	0.8962	35.16
OMTCF-VI	**0.8530**	30.68	**0.8708**	30.88
MovieLens 1M	RMSE, k = 3		RMSE, k = 5	
	RMSE	TIME(s)	RMSE	TIME(s)
OCF	1.1070	10.29	1.0441	10.57
DA-OCF	1.0636	19.76	1.0644	19.98
OMTCF-VI	**0.9766**	19.90	**0.9738**	18.07
MovieLens 1M	MAE, k = 3		MAE, k = 5	
	MAE	TIME(s)	MAE	TIME(s)
OCF	0.9773	10.70	0.8770	10.53
DA-OCF	0.8800	19.50	0.8833	19.54
OMTCF-VI	**0.7737**	22.19	**0.7719**	22.12
MovieLens 10M	RMSE, k = 3		RMSE, k = 5	
	RMSE	TIME(s)	RMSE	TIME(s)
OCF	1.0226	104.17	0.9700	105.35
DA-OCF	1.0262	194.67	1.0285	196.29
OMTCF-VI	**0.9515**	331.39	**0.9565**	333.38
MovieLens 10M	MAE, k = 3		MAE, k = 5	
	MAE	TIME(s)	MAE	TIME(s)
OCF	0.8494	107.24	0.7822	109.09
DA-OCF	0.8284	192.78	0.8328	195.55
OMTCF-VI	**0.7370**	380.13	**0.7425**	347.20

5. CONCLUSIONS

This paper proposed a novel framework of Online Multi-Task Collaborative Filtering (OMTCF) for on-the-fly recommender systems, in which a family of novel OMTCF algorithms were proposed. We conducted an extensive set of experiments by comparing several variants of the proposed algorithms with the existing online algorithms. Our promising empirical results showed that (i) online algorithms are very efficient and highly scalable which can run on large datasets with 10-million ratings in several minutes using a regular machine; and (ii) the proposed online technique is considerably more effective than the existing online algorithms. Future work will address the theoretical analysis of the proposed algorithms and explore new algorithms. For example, our current algorithms treat users as tasks and update the models of multiple users simultaneously. One can also treat items as tasks and update the models of multiple items simultaneously. Our technique may also be extended to tackle some open challenges in CF, e.g., tracking temporal dynamics [12], for which online algorithms potentially could be a natural and perhaps better solution.

Acknowledgements

This work was supported by MOE tier-1 grant (RG33/11).

References

[1] J. Abernethy, K. Canini, J. Langford, and A. Simma. Online collaborative filtering. *UC Berkeley, Tech. Rep.*, 2011.

[2] M. Ali, C. C. Johnson, and A. K. Tang. Parallel collaborative filtering for streaming data. *UT Austin, Tech. Rep.*, 2011.

[3] R. K. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

[4] A. Argyriou, T. Evgeniou, and M. Pontil. Multi-task feature learning. In *NIPS*, pages 41–48, 2006.

[5] E. V. Bonilla, K. M. A. Chai, and C. K. I. Williams. Multi-task gaussian process prediction. In *NIPS*, 2007.

[6] L. Brozovsky and V. Petricek. Recommender system for online dating service. *arXiv preprint cs/0703042*, 2007.

[7] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Linear algorithms for online multitask classification. *Journal of Machine Learning Research*, 11:2901–2934, 2010.

[8] K. Crammer and Y. Singer. Pranking with ranking. In *NIPS*, pages 641–647, 2001.

[9] A. Das, M. Datar, A. Garg, and S. Rajaram. Google news personalization: scalable online collaborative filtering. In *WWW*, pages 271–280, 2007.

[10] E. F. Harrington. Online ranking/collaborative filtering using the perceptron algorithm. In *ICML*, pages 250–257, 2003.

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

[12] Y. Koren. Collaborative filtering with temporal dynamics. In *KDD*, pages 447–456, 2009.

[13] Y. Koren. Factor in the neighbors: Scalable and accurate collaborative filtering. *ACM Trans. KDD*, 4(1):1:1–1:24, 2010.

[14] Y. Koren, R. M. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 42(8):30–37, 2009.

[15] D. Lemire and A. Maclachlan. Slope one predictors for online rating-based collaborative filtering. In *SDM*, 2005.

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

[17] G. Ling, H. Yang, I. King, and M. R. Lyu. Online learning for collaborative filtering. In *IJCNN*, pages 1–8, 2012.

[18] N. N. Liu, M. Zhao, E. W. Xiang, and Q. Yang. Online evolutionary collaborative filtering. In *RecSys*, pages 95–102, 2010.

[19] L. W. Mackey, D. Weiss, and M. I. Jordan. Mixed membership matrix factorization. In *ICML*, pages 711–718, 2010.

[20] P. Melville, R. J. Mooney, and R. Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *AAAI/IAAI*, pages 187–192, 2002.

[21] X. Ning and G. Karypis. Multi-task learning for recommender system. *JMLR - Proceedings Track*, 13:269–284, 2010.

[22] N. D. Phuong and T. M. Phuong. Collaborative filtering by multi-task learning. In *RIVF*, pages 227–232, 2008.

[23] J. D. M. Rennie and N. Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *ICML'05*, pages 713–719, Bonn, Germany, 2005.

[24] A. Saha, P. Rai, H. D. III, and S. Venkatasubramanian. Online learning of multiple tasks and their relationships. In *AISTATS*, 2011.

[25] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. In *NIPS*, 2007.

[26] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295, 2001.

[27] L. Si and R. Jin. Flexible mixture model for collaborative filtering. In *ICML*, pages 704–711, 2003.

[28] N. Srebro, J. D. M. Rennie, and T. Jaakkola. Maximum-margin matrix factorization. In *NIPS*, 2004.

[29] D. H. Stern, R. Herbrich, and T. Graepel. Matchbox: large scale online bayesian recommendations. In *WWW'09*, pages 111–120, Madrid, Spain, 2009.

[30] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. Artificial Intellegence*, 2009, 2009.

[31] M. Weimer, A. Karatzoglou, Q. V. Le, and A. J. Smola. Cofi rank - maximum margin matrix factorization for collaborative ranking. In *NIPS*, 2007.

[32] G.-R. Xue, C. Lin, Q. Yang, W. Xi, H.-J. Zeng, Y. Yu, and Z. Chen. Scalable collaborative filtering using cluster-based smoothing. In *SIGIR*, pages 114–121, Salvador, Brazil, 2005.