Proceedings of the 2007 IEEE Symposium on
Computational Intelligence and Data Mining (CIDM 2007)

One-shot Collaborative Filtering

Shuhei Kuwata and Naonori Ueda
NTT Communication Science Laboratories, NTT Corporation
2–4 Hikaridai, Seika-cho, "Keihanna Science City" Kyoto, 619–0237 Japan
Telephone: +81 (774) 93–5143, Fax: +81 (774) 93–5155
Email: {kuwata,ueda}@cslab.kecl.ntt.co.jp

Abstract— **We propose a new one-shot collaborative filtering method. In contrast to the conventional methods, which predict unobserved ratings individually and independently, our method predicts all unobserved ratings simultaneously and with mutual dependence. With the proposed method, first for observed ratings, we compute empirical marginal distributions of the ratings over users and/or items. Then, for unrated data, these marginal distributions are represented as a function of unknown ratings, and the unknown ratings are predicted by minimizing the Kullback-Leibler (KL) divergence between both the rated and unrated rating distributions. We evaluate the prediction performance and the computational time of our method by using real movie rating data. We confirmed that the proposed method could provide prediction errors comparable to those provided by the conventional top-level methods, but could significantly reduce the computational time.**

I. Introduction

The recommender system is an information filtering technique that can help users by providing only relevant information. It automatically selects items according to each user's preference, and presents them for each user. Research on the recommender system started as an independent research area in mid-1990s, and a lot of work was undertaken in both the academic and industrial fields [1]. There have been a number of applications such as Amazon.com [2], GroupLens [3], and Ringo [4]. Recommender systems are usually categorized into the three groups described below.

- **Content-based filtering**
  This method recommends items that are similar to those that the user preferred in the past, and uses the preference data of only one user for whom we want to make a recommendation. Therefore it tends to recommend items in the same category.
- **Collaborative filtering (CF)**
  This method uses not only the preference data of the user for whom we want to make a recommendation, but also the preference data of the other users. Items are recommended that were preferred in the past by "like-minded" users, who have similar taste. Therefore it is likely that items will be recommended that the user has not previously seen or heard. This method has been already used in, for example, Amazon.com.
- **Hybrid approach**
  This method recommends items by combining both content-based filtering and collaborative filtering methods [5], [6], [7].

Further, CF methods are classified into two major approaches.

- **Nearest-Neighbors (NNs) based CF**
  This approach makes recommendations by finding K-*Nearest-Neighbors* (nearest-users who have similar taste or nearest-items that have similar trends) and by aggregating the preference data of the Nearest-Neighbors. Nearest-Neighbors are extracted from the preference database with a certain similarity measure, such as the Pearson correlation coefficient, vector similarity or adjusted cosine similarity [3], [8], [9], [10].
- **Probabilistic model based CF**
  This approach assumes that the preference data are generated from a probabilistic model. The probabilistic model is learned from the observed preference data with a statistical method and is used for predicting the future preference data [11], [12], [13].

As for the details, see the Ref.[1], [14].

The conventional CF methods mentioned above predict each unknown rating independently. In other words, such methods predict unobserved ratings by using only the observed ones. However, we think that the predicted values for the unrated data have more than a little effective information on the prediction for the other unrated data. We expect that the prediction performance will improve if we take account of the predicted values of the other unobserved ratings when predicting a particular unobserved rating. Based on this idea, in this paper, we propose a new collaborative filtering method in which all unrated ratings are dependently and simultaneously predicted. We call the proposed method *one-shot collaborative filtering*.

As Marlin [12] pointed out, all existing collaborative filtering methods assume that the preference data are *Missing At Random* (MAR). Our method is also based on this assumption. Under the MAR assumption, it is reasonable to think that the preference data distribution of the observed data is similar to that of the unobserved data. Therefore, in our method, we try to predict unobserved ratings by minimizing the distance between these two distributions. More specifically, we refer to preference data distribution as users and/or items marginal rating distributions. Then, for unrated data, these distributions are represented as a function of unknown ratings, and the unknown ratings are predicted by minimizing the *Kullback-Leibler (KL) divergence* [15] between both distributions. Since the prediction results in optimizing an objective function with

1-4244-0705-2/07/$20.00 ©2007 IEEE

respect to the unobserved ratings, our method predicts all unknown ratings simultaneously and dependently.

The contributions of this paper are as follows:

- We first present a new objective function for the one-shot collaborative filtering prediction and present a simple practical algorithm to minimize the objective function.
- We show that the proposed method can provide a prediction performance comparable to that of the conventional top-level methods.
- We show that the one-shot prediction can significantly reduce the computational time.

This paper organized as follows. In section **II**, we formulate the CF problem, which we deal with in this paper. In section **III**, we present the CF method based on a new objective function. In section **IV**, we report experimental results and evaluate the performance of the proposed method. Finally, section **V** provides some conclusion remarks.

II. FORMULATIONS

A. Definitions and Notations

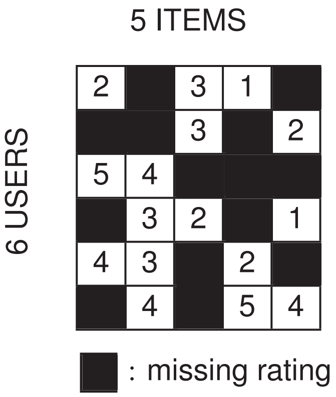
We assume that there are N users and M items, and use i and j to denote a user and an item index, respectively ($i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, M$). We will denote the user for which we want to make a recommendation as the *active user*, and use index a to distinguish this user from the others. Similarly, we will denote the item that we want to recommend as the *target item*, and use index t to distinguish it from the other items.

A user i provides a preference about an item j by assigning it a numerical rating $r_{i,j}$ from the ordinal scale $1, 2, \ldots, V$. Here, 1 (V) is the lowest (highest) score. We will use v to denote a rating value ($v \in \{1, 2, \ldots, V\}$) and $\hat{r}_{i,j}$ to denote the predicted value of $r_{i,j}$. R denotes an N by M rating matrix whose (i, j) element corresponds to $r_{i,j}$. Namely, $r_{i,j} = 0$ indicates that user i has not rated item j.

We assign the set of ratings that the user i has (not) given as $\mathcal{R}^i_{obs}$ ($\mathcal{R}^i_{mis}$), and the set of ratings that the users have (not) given for the item j as $\mathcal{R}^j_{obs}$ ($\mathcal{R}^j_{mis}$), respectively. Moreover, for convenience, we also use $\mathcal{R}_{obs}$ ($\mathcal{R}_{mis}$) to denote a set of ratings that has (not) been rated in R.

Here, we define the subset of $\mathcal{R}^i_{mis}$ which consists of the target ratings we want to predict as $\mathcal{R}^i_{tar}$. Similarly, $\mathcal{R}^j_{tar}$ and $\mathcal{R}_{tar}$ denotes the subset of $\mathcal{R}^j_{mis}$ and $\mathcal{R}_{mis}$, which consists of the target ratings, respectively.

Note that the number of ratings already given by users, $\#\{\mathcal{R}_{obs}\}$, is usually very small compared with the number of ratings not yet given by users, $\#\{\mathcal{R}_{mis}\}$. Here, $\#\{\}$ denotes the element count of the set. In addition, we assume that there is a little difference between the element counts of the set of target ratings and the observed ratings, for example $\#\{\mathcal{R}_{tar}\}$ and $\#\{\mathcal{R}_{obs}\}$. In the movie rating data that we used in the experiment, almost 95% of the elements of R are not given (*sparsity problem* [1]). This indicates that $\#\{\mathcal{R}_{obs}\} \ll \#\{\mathcal{R}_{mis}\}$ and we assume that $\#\{\mathcal{R}_{obs}\} \approx \#\{\mathcal{R}_{tar}\} \ll \#\{\mathcal{R}_{mis}\}$.



2	■	3	1	■
■	■	3	■	2
5	4	■	■	■
■	3	2	■	1
4	3	■	2	■
■	4	■	5	4

Fig. 1. Rating Matirx R ($N = 6, M = 5, V = 5$). Our method is based on the assumption that the rating data are missing at random.

B. Task of CF problem

The goal of the CF problem is to find and recommend items that an active user may favor. There are two types of goal: *prediction* and *recommendation*.

- Prediction is to predict the rating of an item for which the active user has not provided a rating.
- Recommendation involves presenting a list of top-l items that the active user will like the most.

If we predict all unobserved ratings, we can present a top-l list by ranking items based on the predicted rating value. Therefore, the task of the CF problem is reduced to the task of rating prediction problem. We specify our task in this paper as follows:

Given a rating matrix R, predict $\{r_{i,j} \in \mathcal{R}_{tar}\}$.

Although there are some CF methods [7], [16], [5] that take account of the following additional information,

- User demographic information: age, gender, etc.,
- Item profile information: genre, year of publication, etc.,

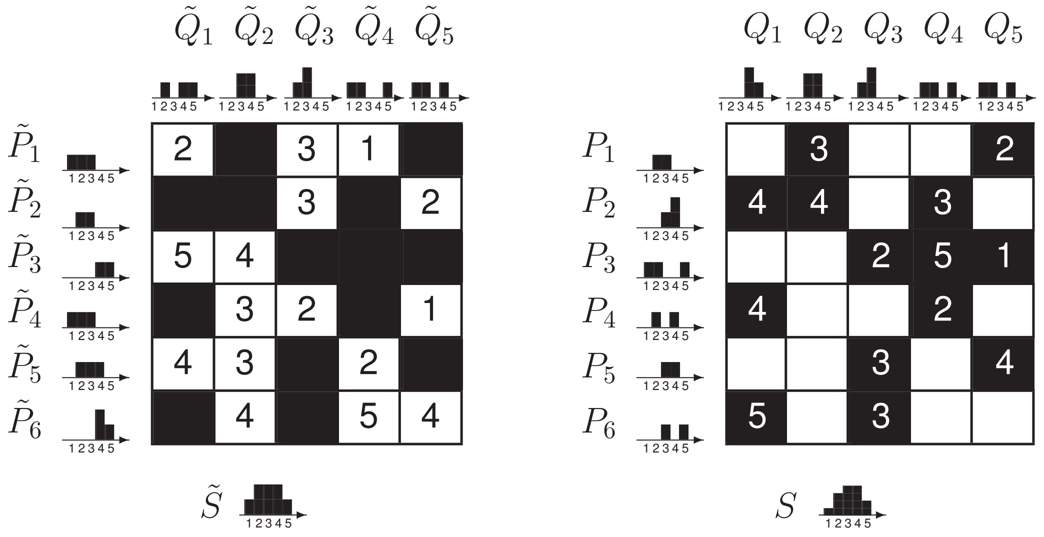
here, we restrict ourselves to CF methods that use only rating matrix R.

C. Performance Measures

As a measure of the prediction accuracy we employ *Normalized Mean Absolute Error (NMAE)* [12] defined by,

$$\begin{aligned} \text{NMAE} &= \frac{\text{MAE}}{\text{E[MAE]}}, \\ \text{MAE} &= \frac{1}{\#\{\mathcal{R}_{tar}\}} \sum_{r_{i,j} \in \mathcal{R}_{tar}} |r_{i,j} - \hat{r}_{i,j}|. \end{aligned}$$

E[MAE] is the expected value of the MAE when we predict rating values randomly. E[MAE] can be easily computed depending on the V value. The smaller NMAE is, the better the prediction performance becomes. An NMAE value of less (greater) than one means the method is performing better (worse) than random prediction. Since the highest rating value V differs from the data sets, normalizations enable us to compare across data sets.


$\tilde{Q}_1$ $\tilde{Q}_2$ $\tilde{Q}_3$ $\tilde{Q}_4$ $\tilde{Q}_5$
$\tilde{P}_1$ $\tilde{P}_2$ $\tilde{P}_3$ $\tilde{P}_4$ $\tilde{P}_5$ $\tilde{P}_6$
1 2 3 4 5
2 3 1
3 2
5 4
3 2 1
4 3 2
4 5 4
$\tilde{S}$
Q_1 Q_2 Q_3 Q_4 Q_5
P_1 P_2 P_3 P_4 P_5 P_6
3 2
4 4 3
2 5 1
4 2
3 4
5 3
S

(a) Marginal rating distributions of *"observed"* ratings

(b) Marginal rating distributions of *"unobserved"* ratings

Fig. 2. Marginal rating distributions in matrix R with $V = 5$. (a) $\tilde{P}_i(i = 1, 2, \ldots, 6)$ is a distribution of observed ratings marginalized in rows, while $\tilde{Q}_j(j = 1, 2, \ldots, 5)$ is a distribution of observed ratings marginalized in columns. $\tilde{S}$ is a distribution of all observed ratings in matrix R. (b) just as (a), $P_i(i = 1, 2, \ldots, 6)$ and $Q_j(j = 1, 2, \ldots, 5)$ are the distributions of unobserved ratings marginalized in rows and columns, respectively. S is a distribution of all unobserved ratings in matrix R. The unknown ratings are predicted by minimizing the KL divergence between the rated and unrated rating distributions.

III. PROPOSED METHOD

A. Missing At Random

As mentioned above, all existing CF methods explicitly or implicitly assume that the preference data are missing at random (See Fig.1). We propose the CF method based on this MAR assumption. If the data are missing at random, then it seems reasonable to suppose that the preference data distribution of the observed data is similar to the preference data distribution of the unobserved data. Therefore, we can evaluate the plausibility of the predicted value for unobserved preference data by the similarity between the preference data distributions of the observed and unobserved data. As the preference data distributions we employ user's and/or item's marginal rating distributions.

B. Marginal Rating Distributions

Our method predicts unrated ratings based on the similarities between rating distributions. We consider the three following types of rating distributions (See Fig.2):

- Rating distributions marginalized in rows of R,
- Rating distributions marginalized in columns of R,
- Rating distributions of all ratings in R.

In the proposed method, first, we compute empirical marginal distributions of the observed ratings over users and/or items, $\tilde{P}_i(i = 1, 2, \ldots, N)$, $\tilde{Q}_j(j = 1, 2, \ldots, M)$ and $\tilde{S}$. Then, we try to obtain unobserved ratings such that empirical marginal rating distributions over users and/or items, $P_i(i = 1, 2, \ldots, N)$, $Q_j(j = 1, 2, \ldots, M)$ and S, are consistent with the corresponding marginal rating distributions of the observed ratings.

The empirical marginal distributions, $\tilde{P}_i$, $\tilde{Q}_j$ and $\tilde{S}$, are calculated as follows:

$$\tilde{P}_i(v) = \frac{\sum_{r_{i,j} \in \mathcal{R}^i_{obs}} \delta(r_{i,j} = v) + \eta}{\sum_l \sum_{r_{i,j} \in \mathcal{R}^i_{obs}} \delta(r_{i,j} = l) + \eta V}, \quad v = 1, \ldots, V. \tag{1}$$

$$\tilde{Q}_j(v) = \frac{\sum_{r_{i,j} \in \mathcal{R}^j_{obs}} \delta(r_{i,j} = v) + \eta}{\sum_l \sum_{r_{i,j} \in \mathcal{R}^j_{obs}} \delta(r_{i,j} = l) + \eta V}, \quad v = 1, \ldots, V. \tag{2}$$

$$\tilde{S}(v) = \frac{\sum_{r_{i,j} \in \mathcal{R}_{obs}} \delta(r_{i,j} = v) + \eta}{\sum_l \sum_{r_{i,j} \in \mathcal{R}_{obs}} \delta(r_{i,j} = l) + \eta V}, \quad v = 1, \ldots, V. \tag{3}$$

Here, $\tilde{P}_i(v)$ denotes the probability that the user i gives the rating value v. $\tilde{Q}_j(v)$ denotes the probability that the rating value v is provided for the item j. $\tilde{S}(v)$ denotes the probability that the rating value v occurs. We introduce the smoothing parameter η to assign a non-zero probability to the unseen rating values, and we set $\eta = 1$. $\delta()$ is an indicator function that takes the value 1 if the argument is true and 0 otherwise. It is obvious that $\sum_v \tilde{P}_i(v) = \sum_v \tilde{Q}_j(v) = \sum_v \tilde{S}(v) = 1$. The empirical marginal distributions of unobserved data, P_i, Q_j and S, are also calculated in the same manner as eqs.(1)-(3), respectively. Note that P_i is obtained by replacing $\mathcal{R}^i_{obs}$ in eq.(1) with $\mathcal{R}^i_{tar}$. Similarly, Q_j and S are obtained by replacing $\mathcal{R}^j_{obs}$ and $\mathcal{R}_{obs}$ with $\mathcal{R}^j_{tar}$ and $\mathcal{R}_{tar}$, respectively.

The "dissimilarity" between both rated and unrated distributions is calculated by the KL divergence, which is a pseudo distance measure between two probability distributions,

$$\begin{aligned}
\mathrm{KL}(\tilde{P}_i(v)||P_i(v)) &= \sum_{v=1}^{V} \tilde{P}_i(v) \log \frac{\tilde{P}_i(v)}{P_i(v)}, \\
\mathrm{KL}(\tilde{Q}_j(v)||Q_j(v)) &= \sum_{v=1}^{V} \tilde{Q}_j(v) \log \frac{\tilde{Q}_j(v)}{Q_j(v)}, \\
\mathrm{KL}(\tilde{S}(v)||S(v)) &= \sum_{v=1}^{V} \tilde{S}(v) \log \frac{\tilde{S}(v)}{S(v)}.
\end{aligned}$$

Here, for example, $\mathrm{KL}(\tilde{P}_i(v)||P_i(v))$ denotes the KL divergence between $\tilde{P}_i(v)$ and $P_i(v)$. $\mathrm{KL}(\tilde{P}_i(v)||P_i(v))$ is nonnegative, and is zero if P_i is exactly the same as $\tilde{P}_i$. Therefore, we can use this measure to evaluate the plausibility of the predicted value for unrated data. We define the following equation as an objective function of a set of unrated data:

$$\begin{aligned}
&J\left(\{r_{i,j} \in \mathcal{R}_{tar}\}\right) = \\
&\frac{1}{N}\sum_{i=1}^{N} \mathrm{KL}(\tilde{P}_i||P_i) + \frac{1}{M}\sum_{j=1}^{M} \mathrm{KL}(\tilde{Q}_j||Q_j) + \mathrm{KL}(\tilde{S}||S). \qquad (4)
\end{aligned}$$

Here, $1/N$ and $1/M$ in eq.(4) are adopted for computing the consistency per distribution. We consider that if the set of ratings for the unobserved ratings $\{r_{i,j} \in \mathcal{R}_{tar}\}$ has a smaller value for the objective function J, this set has a better prediction accuracy.

As a result, the CF problem becomes an optimization problem that minimizes the objective function J (eq.(4)) with respect to unobserved ratings $\{r_{i,j} \in \mathcal{R}_{tar}\}$. Unlike the conventional CF methods, our method predicts each unobserved rating by considering the predicted values for the rest of the unobserved ratings, thus, the proposed method can predict all unknown ratings simultaneously (one-shot prediction).

C. Prediction Algorithm

We can think of several techniques that minimize the objective function J:

- We can apply the combinatorial optimization techniques, which find one or more best sets of ratings in a discrete rating space. However, we need to find the best set from an enormous amount of the number of combination of the unobserved ratings. Therefore this approach may be unsuitable for practical use.
- We can also apply constrained nonlinear optimization techniques, such as Newton's method, when we consider the rating values as being continuous. Here the constraint is that the rating value is in $[1, V]$. However, we found that it was very difficult to find the optimal set we wanted to obtain since there were many local minima.

Although there may be a number of other optimization algorithms, optimization is not the point in question. In addition, in this paper, we attach great importance to the *scalability* for practical use.

Therefore, we present a heuristic and easy-to-use approach for obtaining a set of predicted ratings $\{\hat{r}_{i,j} \in \mathcal{R}_{tar}\}$, where the objective function J in a discrete rating space has a comparatively low value. Roughly, we first assign a set of ratings that has a relatively small J value as the initial set of unobserved ratings. Then, the rating values of this initial set are replaced one by one by new rating values to reduce the value of the objective function J. This replacement continues as long as the value of the objective function J decreases. The prediction algorithm is presented below.

[Prediction Algorithm]

```
Inputs: $\{r_{i,j}|r_{i,j} \in \mathcal{R}_{obs}\}$
Outputs: $\{r_{i,j}|r_{i,j} \in \mathcal{R}_{tar}\}$

Initialize $\{r_{i,j}|r_{i,j} \in \mathcal{R}_{tar}\}$.
for all $r_{i,j} \in \mathcal{R}_{tar}$ do
    Compute,
        $\alpha^P_{i,j} \leftarrow \arg\min_{r_{i,j}} \mathrm{KL}(\tilde{P}_i||P_i)$,
        $\alpha^Q_{i,j} \leftarrow \arg\min_{r_{i,j}} \mathrm{KL}(\tilde{Q}_j||Q_j)$,
        $\beta^{P+Q}_{i,j} \leftarrow \min_{r_{i,j}} \mathrm{KL}(\tilde{P}_i||P_i) + \min_{r_{i,j}} \mathrm{KL}(\tilde{Q}_j||Q_j)$.
end for
Sort $\{r_{i,j}|r_{i,j} \in \mathcal{R}_{tar}\}$ in ascending order of $\beta^{P+Q}_{i,j}$.
while The value of the objective function $J$ decreases. do
    Update the rating in sorted order,
        $r_{i,j} \leftarrow \alpha^P_{i,j}$   only if   $\alpha^P_{i,j} = \alpha^Q_{i,j}$.   (5)
end while
```

Here, $\arg\min_x f(x)$ denotes the argument x that minimizes $f(x)$.

As the initial set $\{r_{i,j}|r_{i,j} \in \mathcal{R}_{obs}\}$, we use the average of both user average $\bar{r}_i$ and item average $\bar{r}_j$,

$$r_{i,j} = (\bar{r}_i + \bar{r}_j)/2. \qquad (6)$$

Here, user average $\bar{r}_i$ and item average $\bar{r}_j$ are computed by,

$$\begin{aligned}
\bar{r}_i &= \sum_{r_{i,j} \in \mathcal{R}^i_{obs}} r_{i,j}/\#\{\mathcal{R}^i_{obs}\}, \\
\bar{r}_j &= \sum_{r_{i,j} \in \mathcal{R}^j_{obs}} r_{i,j}/\#\{\mathcal{R}^j_{obs}\},
\end{aligned}$$

respectively, and we round the continuous value to nearest integer one for use. Since the prediction accuracy of eq.(6) is relatively good, we use it as a baseline for evaluation.

In the prediction algorithm, before the rating update we sort all the target ratings by $\beta^{P+Q}_{i,j}$ to select those that substantially reduce the first and second terms of the objective function J when the rating value is updated by the new rating value $\alpha^P_{i,j}$ or $\alpha^Q_{i,j}$. Note that we decompose the objective function J into its individual terms when we compute the new rating, $\alpha^P_{i,j}$ and $\alpha^Q_{i,j}$. $\alpha^P_{i,j}$ ($\alpha^Q_{i,j}$) is the rating value that minimizes the objective function J with respect to $r_{i,j}$ in rows (columns) of R.

The rating update (eq.(5)) reduces the first and second terms of the objective function J. Here, we introduce the update condition in eq.(5) so that the rating values that minimize the first and second terms of the objective function J in both the rows and columns of R are consistent with each other. If we continue this rating update, the objective function J increases at a certain moment. This is because we do not consider the third term of the objective function J. Therefore, we stop the rating update when the value of the objective function J increases.

IV. EXPERIMENTS

A. Data Sets and Evaluation Protocols

To evaluate the proposed method, we constructed experimental data sets from the following movie rating data sets, which are widely used in CF research: *MovieLens data* (ML1,ML2), *EachMovie data* (EM). MoveLens data are distributed by GrouLens Research at the University of Minnesota, and EachMovie data were collected by the Compaq System Research Center from 1995 through 1997. With the EM, we extracted users who had rated more than 20 movies. The top of Table I summarizes the global statistics of these data sets.

TABLE I

THE CHARACTERISTICS OF MOVIELENS AND EACHMOVIE DATA SETS

	ML1	ML2	EM
# of Users	943	6,040	35,280
# of Items (Movies)	1,682	3,706	1,622
# of Ratings	100,000	1,000,209	2,314,777
Sparsity	93.7%	95.5%	96.0%
Point Scale	5	5	6
E[MAE]	1.6	1.6	1.944
# of Training Users	800	5,000	20,000
# of Test Users	143	1,040	15,280

We employed the experimental setup used in [12]. More specifically, for each data set we randomly divided the users into a set of *training users* and a set of *test users* as shown at the bottom of Table I. We created three independent the training $\&$ test data sets. Note that the number of training (test) users is the same for the three sets. Each of the three data sets, we tested by the same evaluation protocols as in [12]: *weak generalization* and *strong generalization*.

- Weak generalization (closed test) : The available known ratings of each training user are split into observed ratings and held out ratings. The evaluation is performed for each of the held out ratings. When predicting each of the held out ratings of an active user, we can employ all the observed ratings of all the training users including the active user. In this evaluation protocol, following [12], we did not employ test users.
- Strong generalization (open test): The available known ratings of each test user are split into an observed set and a held out set. As with weak generalization, each of the held out ratings is evaluated. But, unlike with weak generalization, when predicting each of the held out ratings of an active user, we can employ all the available known ratings of the training users and the observed ratings of the active user. Namely, we cannot employ the observed ratings of test users other than the active user. This protocol evaluates the prediction performance for the *new* user.

In both types of generalization, the splitting is done by the AllBut n protocol: For each user, n ratings are randomly selected from the available ratings for a held out item set. We set $n = 10\%, 20\%$ and utilize "AllBut 10%", "AllBut 20%", respectively. We created three independent held out item sets for each n and data set.

B. Conventional CF methods

We evaluated the prediction performance and the computational time of our method by comparison with the conventional representative CF methods: Nearest-Neighbors (NNs) based CF methods and probabilistic model based CF methods. In the following we briefly describe the conventional methods used in our experiments.

1) Nearest-Neighbors Approach: This approach tries to predict $r_{a,t}$ based on ratings already given by other users who have similar tastes to the active user. This like-minded user is called a "Nearest-Neighbor". While the traditional NNs based CF methods [3] measure the similarity between users (user-to-user similarity), recent methods measure item-to-item similarity [2], [9], [17]. Below, we refer to the user-to-user (item-to-item) similarity version as *user-base* (*item-base*).

There are a wide variety of similarity measures, including the Pearson correlation coefficient [3] and the adjusted cosine similarity [17]. Since the number of items M is usually fewer than the number of users N, a nearest item search needs much less time than a nearest user search. Therefore item-base has a practical advantage over user-base in terms of scalability.

- **kNNs methods (kNNs) [2], [3], [17]**

Once we find k NNs (users or items) for any similarity measure, the predictive rating of the target item for the active user is computed by the following weighted average of deviations from the average rating of the corresponding user (item) over k NNs.

$$\text{user-base}: \quad \hat{r}_{a,t} = \frac{\sum_{i=1}^{k} W_a^i (r_{i,t} - \bar{r}_i)}{\sum_{i=1}^{k} |W_a^i|} + \bar{r}_a, \qquad (7)$$

$$\text{item-base}: \quad \hat{r}_{a,t} = \frac{\sum_{j=1}^{k} W_t^j (r_{a,j} - \bar{r}_j)}{\sum_{j=1}^{k} |W_t^j|} + \bar{r}_t. \qquad (8)$$

Here, W_a^i is the similarity between active user a and user i. Similarly, W_t^j is the similarity between target item t and item j.

Clearly, the best choice of user-base or item-base, the best choice of similarity measure, and the best choice of k were mutually dependant and also sensitive to a given data set. To evaluate the best potential of NNs based CF, for each data set, we showed the best performance from the results obtained by changing user-base, item-base, similarity measure (Pearson and adjusted cosine), and $k = 100, 200, \ldots, 500$.

- **Unified method with Similarity Fusion (SF) [8]**

This approach is a hybrid method employing user-base and item-base kNNs. Both the nearest-users and the nearest-items are searched for by using any similarity measure as with kNNs. In the prediction phase, each rating in these nearest sets is weighted by the corresponding similarity. Furthermore, the "nearest user-item ratings," which are the common ratings of both the nearest items and the nearest users except for the active user and target item ratings, are also weighted by the similarity computed from the corresponding user-to-user and item-to-item similarities. In the result, each rating $r_{i,j}$ in the nearest-users/items/user-item set $\mathcal{R}_{nns}$ is weighted by each similarity $W^{i,j}_{a,t}$:

$$\begin{aligned} \hat{r}_{a,t} &= \sum_{r_{i,j}\in\mathcal{R}_{nns}} W^{i,j}_{a,t} f_{a,t}(r_{i,j}), \qquad (9) \\ f_{a,t}(r_{i,j}) &= r_{i,j} - (\bar{r}_i - \bar{r}_a) - (\bar{r}_j - \bar{r}_t). \end{aligned}$$

As in eq.(7) and (8), the normalized rating $f_{a,t}(r_{i,j})$ is used instead of $r_{i,j}$ in eq.(9). In our experiments, according to [8], we used the cosine similarity for the nearest-user search and the adjusted cosine similarity for the nearest-item search. Note that we have to set the weight ratio of the user-base/item-base/user-item-base prediction in advance. This is described in detail in [8].

2) Probabilistic Model Approach: This approach assumes that the rating data are generated from a probabilistic model [11], [12], [13]. Therefore user index i, item index j and rating $r_{i,j}$ are treated as random variables. The unknown parameters included in the model are estimated with the statistical learning methods. All methods used in our experiments are based on a multinomial distribution, and therefore after model learning, the *median rating* defined by,

$$\left\{ v \,\middle|\, \Pr\{r_{a,t} < v\} \le \frac{1}{2}, \quad \text{and} \quad \frac{1}{2} \le \Pr\{r_{a,t} > v\} \right\},$$

is used to predict $r_{a,t}$. The expected value can also be used as the prediction value. But we employed the median rating because of its superior performance.

- **Multinomial Model (MULTI) [12]**

This model assigns one multinomial distribution for each item and the ratings of the same item are generated from the same multinomial distribution. The parameters of this model (the probabilities that the rating v generates for the item j) are computed by eq.(2). The joint distribution of user i's ratings is given by,

$$p(r_{i,1}, r_{i,2}, \ldots, r_{i,M}) = \prod_{j=1}^{M} p(r_{i,j}|j)^{\delta(r_{i,j}\neq 0)}.$$

$p(r_{i,j}|j)$ is the multinomial distribution of item j.

- **Mixture of Multinomials Model (MIXMULTI) [12]**

This model assumes that there are C latent classes of users $(z_1, z_2, \ldots, z_C)$ and that users in the same class have similar preferences. The user ratings are generated from the multinomial distribution assigned to the class to which the user belongs. The joint distribution is given by,

$$p(r_{i,1}, r_{i,2}, \ldots, r_{i,M}) = \sum_{c=1}^{C} p(z_c) \prod_{j=1}^{M} p(r_{i,j}|j, z_c)^{\delta(r_{i,j}\neq 0)}.$$

Here, $p(z_c)$ denotes the probability that class z_c occurs, and $p(r_{i,j}|j, z_c)$ denotes the multinomial distribution of item j in class z_c. Clearly, the prediction performance depends on C. In our experiments, we selected the best C from the results obtained by changing $C = 5, 10, 15$. Moreover, we stopped the iteration when the model learning had been repeated a maximum of 100 turns to avoid overfitting.

- **Aspect Model (AM) [12]**

This model assumes that there are C latent communities of users, and that the like-minded users are in the same communities. This model might be called *probabilistic Latent Semantic Analysis (pLSA)* [11], which was originally developped in the context of information retrieval [18]. It differs from the Mixture of Multinomials Model in that the user ratings are generated from multiple multinomial distributions. This model allows the users to belong to more than one class. The joint distribution is given by,

$$p(r_{i,1}, r_{i,2}, \ldots, r_{i,M}) = \prod_{j=1}^{M} \sum_{c=1}^{C} p(z_c|i) p(r_{i,j}|j, z_c)^{\delta(r_{i,j}\neq 0)}. \qquad (10)$$

Note that this model is not a correct generative model since we need to estimate all the parameters in this model again whenever a new user comes in. C was chosen in the same manner as in MIXMULTI. As in MIXMULTI, we stopped the iteration when the model learning had been repeated a maximum of 100 turns to avoid overfitting.

- **User Rating Profile Model (URP) [12], [13]**

This model is a correct generative model version of AM. In URP, $p(z_c|i)$ in eq.(10) is replaced with $p(z_c|\theta)p(\theta;\alpha)$. Here, $p(\theta;\alpha)$ is Dirichlet distribution with the hyper parameter α. The joint distribution is given by,

$$\begin{aligned} & p(r_{i,1}, r_{i,2}, \ldots, r_{i,M}) \\ &= \int_{\theta} p(\theta;\alpha) \prod_{j=1}^{M} \sum_{c=1}^{C} p(z_c|\theta) p(r_{i,j}|j, z_c)^{\delta(r_{i,j}\neq 0)} d\theta. \end{aligned}$$

The model parameters of MIXMULTI, AM and URP are learned by the Expectation-Maximization (EM) algorithm [12], [19]. C was chosen in the same manner as in MIXMULTI. Although this model can make a prediction for the new user without relearning the model parameters, the learning algorithm is more complex and requires more computational time than that of AM. In addition, we found that this model suffered greatly from the local optima problem. Therefore, following [12], we used the result provided by MIXMULTI as the initial prediction values. Moreover, to avoid overfitting, we stopped the learning iteration at 10 turns (much less than in AM).

TABLE II

THE PREDICTION PERFORMANCE OF EACH CF METHOD: A SMALLER VALUE MEANS BETTER PREDICTION PERFORMANCE.

(a) Weak Generalization

		OSCF	kNNs	SF	MULTI	MIXMULTI	AM	URP	BASE LINE
ML1	AllBut10%	**0.456**	0.468	0.477	0.488	0.489	0.470	0.487	0.498
	AllBut20%	**0.457**	0.470	0.479	0.490	0.492	0.486	0.492	0.499
ML2	AllBut10%	0.443	0.440	0.459	0.464	0.445	**0.416**	0.464	0.481
	AllBut20%	0.446	0.441	0.459	0.465	0.444	**0.419**	0.468	0.482
EM	AllBut10%	0.445	0.419	0.451	0.463	0.409	**0.403**	0.465	0.469
	AllBut20%	0.447	0.421	∗	0.463	0.411	**0.405**	0.465	0.469

(b) Strong Generalization

		OSCF	kNNs	SF	MULTI	MIXMULTI	AM	URP	BASE LINE
ML1	AllBut10%	**0.450**	0.456	0.463	0.477	0.464	-	0.474	0.488
	AllBut20%	**0.458**	0.461	0.469	0.480	0.476	-	0.486	0.493
ML2	AllBut10%	**0.446**	0.449	0.460	0.467	0.448	-	0.472	0.483
	AllBut20%	0.447	0.452	0.461	0.468	**0.446**	-	0.470	0.484
EM	AllBut10%	0.445	0.428	0.456	0.462	**0.407**	-	0.465	0.469
	AllBut20%	0.446	0.430	0.451	0.462	**0.406**	-	0.464	0.469

TABLE III

THE COMPUTATIONAL TIME OF EACH CF METHOD: A SMALLER VALUE MEANS BETTER SCALABILITY.

(a) Weak Generalization

		OSCF	kNNs	SF	MULTI	MIXMULTI	AM	URP	BASE LINE
ML1	AllBut10%	0.67 s	1.24 m	5.10 m	**0.22 s**	11.0 s	5.74 m	4.42 m	**0.22 s**
	AllBut20%	1.00 s	2.19 m	8.67 m	**0.21 s**	8.33 s	6.40 m	4.39 m	0.22 s
ML2	AllBut10%	11.9 s	2.36 h	22.4 h	**4.22 s**	10.3 m	1.65 h	1.20 h	4.33 s
	AllBut20%	15.0 s	3.82 h	40.4 h	**4.11 s**	12.8 m	1.67 h	4.00 h	4.22 s
EM	AllBut10%	41.3 s	12.2 h	220 h	**13.9 s**	3.98 h	5.35 h	3.54 h	14.1 s
	AllBut20%	50.6 s	21.5 h	∗	**13.8 s**	4.71 h	5.48 h	3.18 h	14.2 s

(b) Strong Generalization

		OSCF	kNNs	SF	MULTI	MIXMULTI	AM	URP	BASE LINE
ML1	AllBut10%	0.44 s	15.8 s	1.01 m	**0.18 s**	17.0 s	-	11.3 m	0.22 s
	AllBut20%	0.56 s	31.3 s	1.95 m	**0.18 s**	17.7 s	-	10.2 m	0.11 s
ML2	AllBut10%	8.44 s	11.1 m	4.57 h	**3.78 s**	11.4 m	-	1.48 h	4.08 s
	AllBut20%	9.11 s	18.9 m	7.42 h	**3.89 s**	15.1 m	-	1.66 h	3.96 s
EM	AllBut10%	29.7 s	1.21 h	59.9 h	**12.8 s**	4.73 h	-	5.13 h	13.8 s
	AllBut20%	31.8 s	1.62 h	145 h	**12.8 s**	4.20 h	-	4.20 h	13.4 s

C. RESULTS & DISCUSSION

Tables II and III show the results obtained with a Dual Xeon 3.60GHz CPU and 2GB memory machine. The prediction performances are shown in Table II and the computational times are shown in Table III. We refer to our method as *One-Shot Collaborative Filtering* (OSCF). BASE LINE is a prediction method using only eq.(6). Since AM needs to learn the model parameteres whenever the new user comes in, it is not applicable for strong genelarization case. Therefore the corresponding entries are indicated by “-”. In Table III, s, m and h denote seconds, minutes and hours, respectively. The figures in these tables are the average values over three test data sets. Note that the symbol “∗” in Tables II and III indicates that since the method could not predict ratings in reasonable time (less than two weeks), we gave up the computation.

1) Prediction Performance Results: As shown in Table II, all the methods were able to obtain smaller NMAE values than BASE LINE for both weak and strong generalizations. Moreover the prediction performances by all the methods for large data (EM) were better than those for small data (ML1). The tendency was particularly dominant with the probabilistic models (MIXMULTI and AM). This is intuitively reasonable since larger data have more information than smaller data, and larger data consist of some similar clusters. Note that the best C varied in the model and data sets.

From the results, it would be hard to say which method was the best for all data sets, but we can mention the following interesting results. In both cases ((a) & (b)), the proposed method (OSCF) obtained the best performances for small data (ML1). This indicates that, as we expected, dependent prediction worked effectively, particularly when the numbers of users and items were small.

In contrast, for large data (EM) OSCF was worse than the other methods excluding SF. In particular, the probabilistic models (MIXMULTI and AM) significantly outperformed the other approaches for EM data. This means that the introduction of the latent classes or communities into the model was certainly successful for large data sets.

Note that the results of MIXMULTI, AM and URP are the best results obtained by manually selecting C, and model initialization of URP are accomplished by using the MIXMULTI result, as mentioned before. That is, these models were advantageous for comparing the best performances of the models with the proposed method.

The OSCF results for EM were slightly better than those for ML1 and ML2. However, it is clear that OSCF was the most stable method. Although MULTI performed relatively stable for all data, it was worse than OSCF in all cases.

SF was worse than kNN. The main reason is that the SF parameters were sensitive to the data set and we have to adjust the SF parameter values according to the data set.

2) Computational Time Results: As shown in Table III, the larger the data scale became, the more computational time we needed. However, the increase in the computational time with respect to data scale varied greatly according to the method. In both cases ((a) & (b)), OSCF could predict all unobserved data in significantly less time than the other methods excluding MULTI and BASE LINE. Although the computational times of MULTI and BASE LINE were less than that of OSCF, the differences were negligible compared with that of the other approaches. In contrast, the computational time of kNNs, SF, MIXMULTI, AM and URP increased rapidly as the data scale increased.

The main reason for OSCF being able to provide a prediction in a very short time is that the OSCF prediction algorithm dose not depend on the matrix scale (N or M) but on the number of the target ratings $\#\{\mathcal{R}_{tar}\}$, while the others depend on the matrix scale (N or M or NM).

Here, SF requires a long computational time since this method has to search for both the nearest-users and the nearest-items. If we searched for the both nearest-neighbors in parallel, we would obtain the SF prediction result faster.

It is clear that if we continue the iteration for the parameter estimation in MIXMULTI, AM and URP more than 100 turns, it will require more computational time. The scale of the data set in the real world is incomparably larger than that of this experiment. Therefore, simply because the prediction performance is the best, it does not follow that the method is the best.

V. CONCLUSIONS

We proposed a CF method that predicts all target ratings simultaneously. Our method predicts one unobserved rating taking the predicted values for the other unrated data into consideration, while the conventional methods predict the unobserved rating individually. Experimental results showed that the proposed method provides a comparable prediction performance to the conventional CF methods in a significantly shorter time. As our method needs only the MAR assumption, we can use it for a wide variety of missing data.

REFERENCES

[1] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Trans. Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, June 2005.

[2] G. Linden, B. Smith, and J. York, "Amazon.com Recommendations: Item-to-Item Collaborative Filtering," *IEEE Internet Computing*, vol. 7, no. 1, pp. 76–80, January/February 2003.

[3] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An Open Architecture for Collaborative Filtering of Netnews," in *Proceedings of ACM CSCW1994*, Chapel Hill, NC, US, October 1994, pp. 175–186.

[4] U. Shardanand and P. Maes, "Social Information Filtering: Algorithms for Automating "Word of Mouth"," in *Proceedings of ACM CHI1995*, Denver, Colorado, US, May 1995, pp. 210–217.

[5] L. Si and R. Jin, "Unified Filtering by Combining Collaborative Filtering and Content-Based Filtering via Mixture Model and Exponential Model," in *Proceedings of ACM CIKM2004*, Washington D.C., US, November 2004, pp. 156–157.

[6] K. Yu, V. Tresp, and S. Yu, "A Nonparametric Hierarchical Bayesian Framework for Information Filtering," in *Proceedingss of ACM SIGIR2004*, Sheffield, UK, July 2004.

[7] Z. Xu, V. Tresp, K. Yu, and H.-P. Kriegel, "Infinite Hidden Relational Models," in *Proceedings of the 22nd International Conference on Uncertainty in Artificial Intelligence (UAI2006)*, Cambridge, MA, US, July 2006.

[8] J. Wang, A. P. de Vries, and M. J. Reinders, "Unifying User-based and Item-based Collaborative Filtering Approaches by Similarity Fusion," in *Proceedings of ACM SIGIR2006*, Seattle, Washington, US, August 2006.

[9] M. Deshpande and G. Karypis, "Item-Based Top-N Recommendation Algorithms," *ACM Trans. Information Systems*, vol. 22, no. 1, pp. 143–177, January 2004.

[10] 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 *Proceedings of ACM SIGIR2005*, Salvador, Brazil, August 2005.

[11] T. Hofmann, "Latent Semantic Models for Collaborative Filtering," *ACM Trans. Information Systems*, vol. 22, no. 1, pp. 89–115, January 2004.

[12] B. Marlin, "Collaborative Filtering: A Machine Learning Perspective," Master's thesis, University of Toronto, 2004.

[13] ——, "Modeling User Rating Profiles For Collaborative Filtering," in *Proceedings of the 17th Annual Conference on Neural Information Processing Systems (NIPS-2003)*, British Columbia, Canada, December 2001.

[14] J. S. Breese, D. Heckerman, and C. Kadie, "Empirical Analysis of Predictive Algorithms for Collaborative Filtering," in *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI)*, Madison, Wisconsin, US, July 1998, pp. 43–52.

[15] S. Kullback and R. Leibler, "On information and sufficiency," *Annals of Mathematical Statistics*, vol. 22, pp. 79–86, 1951.

[16] L. Getoor and M. Sahami, "Using Probabilistic Relational Models for Collaborative Filtering," in *Proceedings of Workshop on Web Usage Analysis and User Profiling (WEBKDD-99)*, San Diego, CA, US, August 1999.

[17] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," in *Proceedings of ACM WWW2001*, Honolulu, Hawaii, US, May 2001, pp. 285–295.

[18] T. Hofmann, "Unspervised Learning by Probabilistic Latent Semantic Analysis," *Machine Learning*, vol. 42, pp. 177–196, 2004.

[19] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society Series B*, vol. 34, pp. 1–38, 1977.