Mach Learn (2014) 97:177–203
DOI 10.1007/s10994-014-5454-z

Collaborative filtering with information-rich and information-sparse entities

Kai Zhu · Rui Wu · Lei Ying · R. Srikant

Received: 2 March 2014 / Accepted: 29 May 2014 / Published online: 8 August 2014
© The Author(s) 2014

Abstract In this paper, we consider a popular model for collaborative filtering in recommender systems. In particular, we consider both the clustering model, where only users (or items) are clustered, and the co-clustering model, where both users and items are clustered, and further, we assume that some users rate many items (information-rich users) and some users rate only a few items (information-sparse users). When users (or items) are clustered, our algorithm can recover the rating matrix with $\omega(MK\log M)$ noisy entries while MK entries are necessary, where K is the number of clusters and M is the number of items. In the case of co-clustering, we prove that K^2 entries are necessary for recovering the rating matrix, and our algorithm achieves this lower bound within a logarithmic factor when K is sufficiently large. Extensive simulations on Netflix and MovieLens data show that our algorithm outperforms the alternating minimization and the popularity-among-friends algorithm. The performance difference increases even more when noise is added to the datasets.

Keywords Recommender system · Collaborative filtering · Matrix completion · Clustering model

Editors: Toon Calders, Rosa Meo, Floriana Esposito, and Eyke Hullermeier.

K. Zhu (✉) · L. Ying
School of Electrical, Computer and Energy Engineering, Arizona State University,
Tempe, AZ 85287, USA
e-mail: kzhu17@asu.edu

L. Ying
e-mail: lei.ying.2@asu.edu

R. Wu · R. Srikant
Department of Electrical and Computer Engineering, University of Illinois at Urbana-Champaign,
Urbana, IL 61801, USA

R. Wu
e-mail: ruiwu1@illinois.edu

R. Srikant
e-mail: rsrikant@illinois.edu

1 Introduction

Many websites today use recommender systems to recommend items of interests to their users. Well known examples include Amazon, Netflix and MovieLens, where each user is suggested items that he or she may like, using partial knowledge about all the users' likes and dislikes. In this paper, we focus on the so-called Netflix or MovieLens model in which there are a large number of users and a large number of movies (called items in this paper), and each user rates a subset of the items that they have watched. These ratings are typically from a discrete set; for example, each item could be given a rating of 1 through 5. If one views the user ratings as a matrix, with users as the rows and the items as the columns, then the resulting rating matrix is typically very sparse. The reason is that most users rate only a few items. The goal of a recommender system in such a model is to recommend items that a user may like, using the sparse set of available ratings. While the real goal is to just recommend a few items that each user would like, mathematically the problem is often posed as a *matrix completion* problem: fill in all the unknown entries of the matrix. The use of partial knowledge of about other users' preferences to make a prediction about a given user's preference is referred to as collaboration, and the process of making predictions is called filtering; therefore, recommender systems which use multiple users' behaviors to predict each user's behavior is said to use *collaborative filtering*. For example, GroupLens (Resnick et al. 1994; Konstan et al. 1997) is an early recommendation system that uses collaborative filtering.

With no assumptions, the matrix completion problem is practically impossible to solve. In reality, it is widely believed that the unknown matrix of all the ratings has a structure that can be exploited to solve the matrix completion problem. The two most common assumptions about the rating matrix are the following:

Low-rank assumption The assumption here is that the rating matrix has a small rank. Suppose that there are U users and M items, then the true rating matrix $\mathbf{B}$ is an $U \times M$ matrix. The low rank assumption means that the rank of the matrix $\mathbf{B}$ is assumed to be $K << \min\{U, M\}$. This assumption is typically justified by recalling a well-known result in linear algebra which states that every $U \times M$ matrix of rank K can be written in the form $\mathbf{A}\mathbf{D}^T$, where $\mathbf{A}$ is an $U \times K$ matrix and $\mathbf{D}$ is an $M \times K$ matrix. Thus, the $(i, j)^{\text{th}}$ entry of the rating matrix $\mathbf{B}$, can be written as $b_{ij} = \mathbf{a}_i \mathbf{d}_j^T$, where $\mathbf{a}_i$ is the i^{th} row of $\mathbf{A}$ and $\mathbf{d}_j$ is the j^{th} row of $\mathbf{D}$. Since $\mathbf{a}_i$ and $\mathbf{d}_j$ are vectors of length K, the low-rank assumption can be viewed as follows: each user and item can be characterized by K features each, and the rating of a item by a user is simply the inner product of these features. The low-rank assumption is popular because it is viewed as a mathematical abstraction of the real-life situation in which a typical user looks for only a features of a movie (such as the lead actors, director, language, genre, etc.) before he/she decides to watch it.

Cluster assumption The assumption here is that users and items are grouped into clusters, such that users in the same cluster will provide similar ratings to items in the same cluster. If there are K user and item clusters, and each user in a cluster provides the same rating to each item in a cluster, then the rating matrix can be summarized by a $K \times K$ block matrix. Thus, the cluster assumption would then be a stronger assumption than low-rank assumption since the rating matrix would only have K independent rows (and columns) and is thus also a rank-K matrix. However, if the stronger assumption leads to lower complexity algorithms with better predictive power, then such an assumption is well-justified. We emphasize that the true ratings are unknown, and so neither of the two assumptions can be actually verified in a real-life data set. The only way to justify an assumption is by studying the performance of algorithms resulting from the assumption,

by using some of the known ratings as training data to predict the remaining known ratings.

1.1 Prior work

We now briefly review some of the algorithms that have resulted from the above assumptions. One way to exploit the low rank assumption is to find a matrix whose rank is the smallest among all matrices which agree with the observed ratings at the known entries of the matrix. However, the resulting rank minimization problem is not a convex problem, and a popular heuristic is to replace the rank minimization objective with a nuclear norm minimization objective. In (Candès and Recht 2009; Candès and Tao 2010; Recht 2011; Gross 2011; Keshavan et al. 2010), it was shown that, under some conditions, both the nuclear norm minimization problem and the rank minimization problem yield the same result. Additionally, the nuclear norm minimization problem is a convex problem, and therefore, polynomial-time algorithms exist to solve such problems (Keshavan et al. 2010; Cai et al. 2010). In reality, each step of such algorithms requires one to perform complicated matrix operations (such as the Singular Value Decomposition or SVD) and many such steps are required for convergence, and hence such algorithms are too slow in practice. An alternative algorithm which runs much faster and is also quite popular in practice is the so-called alternating minimization (AM) algorithm: initially, an SVD of the known ratings portion of the rating matrix is performed to estimate the rank, and the $\mathbf{A}$ and $\mathbf{D}$ matrices mentioned earlier. Then, $\mathbf{A}$ is assumed to be fixed and a "better" $\mathbf{D}$ is obtained by minimizing the squared error between the entries of $\mathbf{AD}^T$ and the known entries of the rating matrix. Then, the resulting $\mathbf{D}$ is taken to be fixed, and a better $\mathbf{A}$ is selected using the same criterion. This process is repeated until some convergence criterion is satisfied. Thus, the AM algorithm simply alternates between improving the estimate of $\mathbf{A}$ and the estimate of $\mathbf{D}$, and hence its name. AM is one of two popular approaches for matrix factorization which is a major component of the winning algorithm of the Netflix challenge (Koren et al. 2009). This algorithm has been known for a while, but no performance guarantees were known till recently. Recently, it has been theoretically established in (Jain et al. 2013) that a variant of the AM algorithm obtains the true ratings under the same so-called *incoherence* assumption as in the case of nuclear norm minimization, and only requires slightly more entries to guarantee exact recovery of the rating matrix. Thus, for practical purposes, one can view the AM algorithm as the best algorithm known to date for the matrix completion problem under the low-rank assumption.

If the cluster assumption is made, then one can apply popular clustering methods such as K-means or spectral clustering to first cluster the users. Then, to predict the rating of a particular user, one can use a majority vote among users in his/her cluster. Such an approach has been used in (Tomozei and Massoulié 2014). In (Barman and Dabeer 2012), for each user, a subset of other users is selected to represent the user's cluster. The subset is chosen by computing a similarity score for each pair of users, and identifying the top k users who are most similar to the given user. Then the user's rating for a particular user is computed by a majority vote among his/her similarity subset. Note that neither of these algorithms uses the item clusters explicitly, and therefore, their performance could be far from optimal. Recalling our earlier comment that clustered models can be thought of as a special case of low-rank models, one can apply the low-rank matrix completion algorithms to the cluster models as well. Such an approach was taken in (Xu et al. 2013), where it was shown that the mathematics behind the low-rank matrix completion problem can be considerably simplified under the cluster assumption. The paper, however, does not exploit the presence of information-rich users (or items) in the system.

1.2 Our results

The focus of this paper is on the matrix completion problem under the cluster assumption. We consider both the clustering and co-clustering models, i.e., our results are applicable to the case when only users (or items) are assumed to be clustered and to the case when both items and users are assumed to be clustered. While the former case is considered for completeness and comparison to prior work, we believe that the latter assumption is even better based on experiments with real datasets which we report later in the paper. *The significant departure in our model compared to prior work is the assumption that there are information-rich and information-sparse users and items.* In particular, we assume that there exist some users who rate a significantly larger number of items than others, and we call the former *information-rich* and the latter *information-sparse*. We borrowed this terminology from (Banerjee et al. 2012) who use it in the context of their privacy and anonymity model. The presence of information-rich and information-sparse users in data sets has been well known. For example, in the MovieLens dataset, 38 out of 6,040 users rated more than 1,000 movies (the dataset has 3,952 movies in total), but more than 73 % of the users rated fewer than 200 movies. To the best of our knowledge, the presence of information-rich entities has not been exploited previously for matrix completion.

Our main contributions are as follows:

1. We present a clustered rating matrix model in which each cluster has information-rich and information-sparse entities, where an entity here could mean a user or an item. We note that an information-rich user may only rate a small fraction of the total number of items, but this fraction is distinctly greater than that of an information-sparse user. A similar comment applies to information-rich items as well.
2. We devise a similarity based scheme as in (Barman and Dabeer 2012) to exploit the presence of information-rich users to dramatically improve the performance of the algorithm in (Barman and Dabeer 2012). Two remarks are in order here: first, our algorithm uses a normalization, not found in (Barman and Dabeer 2012), which allows us compare users (and items) with widely different numbers of ratings. The second remark is that, by exploiting the presence of information-rich users, the performance of our algorithm achieves an easily provable lower-bound with a logarithmic factor when we only assume that users are clustered. In the case of co-clustering, the lower bound is achieved within a logarithmic factor if the number of clusters is sufficiently large.
3. As mentioned earlier, in practice, it is hard to verify what assumptions a true rating matrix will satisfy since the matrix is unknown except for a sparse subset. In particular, even if the cluster assumption holds, it is not clear whether each subset will contain an information-rich user and an information-sparse user. Therefore, using the theory developed here, we present a heuristic algorithm which exploits the cluster assumption, but does not require each cluster to have an information-rich user. This algorithm, which is our recommended algorithm, combines the theory presented in this paper with the algorithm in (Barman and Dabeer 2012).
4. We compare our algorithm with the AM algorithm, and the similarity score-based algorithm in (Barman and Dabeer 2012) known as the PAF (popularity-among-friends) algorithm by applying all three to both the MovieLens and Netflix datasets. As can be seen from the results presented later in the paper, our algorithm performs significantly better.
5. The proposed algorithm has low computational complexity. Consider the case where the number of items M is equal to the number of users. Further let α denote the fraction of entries in the ratings matrix that are known, and C denote the size of each user/item

cluster. Then the similarity score for each user pair requires $\alpha^2 M$ computations. Since there are $M(M-1)$ similarity scores to be computed for the users, and another $M(M-1)$ scores for items, the similarity score computation for co-clustering requires $O(\alpha^2 M^3)$ computations. Further, since sorting M items requires $O(M \log M)$ computations, identifying user and item clusters require a total of $O(\alpha M^3 + M^2 \log M)$ computations. Finally, majority voting in each $C \times C$ block requires αC^2 operations. So the overall complexity is $O(\alpha^2 M^3 + M^2 \log M + \alpha M^2 C^2)$. Here one of the three terms $O(\cdot)$ will dominate, depending on the assumptions regarding the order of α and C. Since α is typically small (please see the performance evaluation section for actual numbers), the computational complexity of our algorithm is quite small.

6 The proposed algorithm can be easily implemented in a distributed and parallel fashion. The pair-wise similarity scores can be computed in parallel, and the majority voting to determining the ratings can also be done simultaneously.

7. Another notable advantage of our algorithm (which is also the case for the algorithm in Barman and Dabeer 2012) compared to the algorithms in (Cai et al. 2010; Keshavan et al. 2010; Tomozei and Massoulié 2014) is that our algorithm can handle incremental additions to the rating matrix easily. For example, if a few new rows and/or columns are added to the matrix, then it is easy for our algorithm to compute additional similarity scores and obtain a similarity set of each user/item. This would be more computationally burdensome in the case of the any other algorithm surveyed in this paper, since one has to rerun the algorithm on the entire rating matrix.

2 Model

In this section, we present the basic model. A summary of notations can be found in 'Notation' section of Appendix 2. We consider a recommendation system consisting of U users and M items, and U and M are at the same order ($U = \Theta(M)$). Let $\mathbf{B}$ denote the $U \times M$ preference matrix, where b_{um} is the preference level of user u to item m. We assume that there are G different levels of preference, so $b_{um} \in \{1, \ldots, G\}$.

The goal of the recommendation system is to recover the preference matrix $\mathbf{B}$ from a sparse and noisy version of the matrix, named the rating matrix $\mathbf{R}$. We assume the users, or the items, or both are clustered. When the users are assumed to be clustered, they form K user-clusters, each with size U/K. We assume $K = O(M/\log M)$. In other words, the cluster size is at least of the order of $\log M$. Without loss of generality, we assume users are indexed in such a way that user u is in user-cluster $\lceil \frac{u}{U/K} \rceil = \lceil \frac{uK}{U} \rceil$. We further assume users in the same cluster have the same preference to every item, i.e., $b_{um} = b_{vm}, \forall m$, if $\lceil \frac{uK}{U} \rceil = \lceil \frac{vK}{U} \rceil$. Furthermore, we say that the preference matrix $\mathbf{B}$ is *fractionally separable for users* if the following condition holds.

Condition 1 (**Fractional Separability Condition for Users**) *There exists a constant* $0 < \mu < 1$ *such that*

$$\sum_{m=1}^{M} \mathbf{1}_{b_{um}=b_{vm}} \leq \mu M \quad \text{if} \quad \left\lceil \frac{uK}{U} \right\rceil \neq \left\lceil \frac{vK}{U} \right\rceil$$

$$\sum_{m=1}^{M} \mathbf{1}_{b_{um}=b_{vm}} = M \quad \text{if} \quad \left\lceil \frac{uK}{U} \right\rceil = \left\lceil \frac{vK}{U} \right\rceil$$

In other words, for any pair of users u and v who are not in the same cluster, they have the same preference on at most μ fraction of the items; and for any pair of users in the same cluster, they have the same preference on all items. □

When the items are assumed to be clustered, the items form K item-clusters, each with size M/K. Again, we assume items are indexed in such a way that item m is in cluster $\lceil \frac{m}{M/K} \rceil = \lceil \frac{mK}{M} \rceil$. We assume the items in the same cluster receive the same preference from the same user, i.e.,

$$b_{um} = b_{un} \quad \forall u \quad \text{if} \quad \left\lceil \frac{mK}{M} \right\rceil = \left\lceil \frac{nK}{M} \right\rceil.$$

We say the preference matrix $\mathbf{B}$ is *fractionally separable for items* if the following condition holds.

Condition 2 (**Fractional Separability Condition for Items**) *There exists a constant* $0 < \mu < 1$ *such that*

$$\sum_{u=1}^{U} \mathbf{1}_{b_{um}=b_{un}} \leq \mu U \quad \text{if} \quad \left\lceil \frac{mK}{M} \right\rceil \neq \left\lceil \frac{nK}{M} \right\rceil$$
$$\sum_{u=1}^{U} \mathbf{1}_{b_{um}=b_{un}} = U \quad \text{if} \quad \left\lceil \frac{mK}{M} \right\rceil = \left\lceil \frac{nK}{M} \right\rceil$$

In other words, for any pair of items m and n that are not in the same cluster, they receive the same preference from at most μ fraction of the users; and for any pair of items in the same cluster, they receive the same preference from all users. □

The observed rating matrix $\mathbf{R}$ is assumed to be sparse and noisy because most users only rate a few items, and a user may give inconsistent ratings to the same items when being asked multiple times (Amatriain et al. 2009). The process of generating $\mathbf{R}$ is illustrated below, where $\mathbf{B}$ is first passed through a noisy channel to create $\tilde{\mathbf{R}}$, and then $\tilde{\mathbf{R}}$ is passed through an erasure channel to create $\mathbf{R}$.

$$\mathbf{B} \xrightarrow{\text{a noisy channel}} \tilde{\mathbf{R}} \xrightarrow{\text{an erasure channel}} \mathbf{R}.$$

We assume the noisy channel satisfies a *biased rating* property under which a user is more likely to reveal the true preference.

Condition 3 (**Biased Rating**) *Given any* b_{um},

$$\Pr(\tilde{r}_{um} = g) = \begin{cases} p, & \text{if} \quad g = b_{um} \\ \frac{1-p}{G-1}, & \text{otherwise.} \end{cases},$$

where $p > \frac{1}{G}$. *So the probability that a user reveals the true preference is larger than the probability the user gives any other rating.* □

Recall that $\mathbf{R}$ contains only a few entries of $\tilde{\mathbf{R}}$. Let $r_{um} = \star$ if the entry is erased; and $r_{um} = \tilde{r}_{um}$ otherwise. We define two types of users: *information-rich users* who rate a large number of items and *information-sparse users* who rate only a few items. Specifically, the information-rich users and information-sparse users are defined as follows.

Condition 4 (**Heterogeneous Users**) *For an information-rich user u,*

$$\Pr(r_{um} = \star) = 1 - \beta \quad \textit{for all} \quad m;$$

and for an information-sparse user v,

$$\Pr(r_{vm} = \star) = 1 - \alpha \quad \textit{for all} \quad m.$$

In other words, an information-rich user rates βM items on average; and an information-sparse user rates αM items on average. We further assume the erasures are independent across users and items, the number of information-rich users in each user-cluster is at least 2 and at most η (a constant independent of M), $\alpha = o(\beta)$, and $\beta \leq \beta_{\max} < 1$. □

We further define two types of items: *information-rich items* that receive a large number of ratings and *information-sparse items* that receives only a few ratings. Specifically, the information-rich items and information-sparse items are defined in the following assumption.

Condition 5 (**Heterogeneous Items**) *For an information-rich item m,*

$$\Pr(r_{um} = \star) = 1 - \beta \quad \textit{for all} \quad u;$$

and for an information-sparse item n,

$$\Pr(r_{un} = \star) = 1 - \alpha \quad \textit{for all} \quad u.$$

In other words, an information-rich item receives βU ratings on average; and an information-sparse item receives αU ratings on average. We further assume the erasures are independent across users and items, the number of information-rich items in each item-cluster is at least 2 and at most η (a constant independent of M), $\alpha = o(\beta)$, and $\beta \leq \beta_{\max} < 1$. □

Remark In real datasets, the number of ratings per user is small. To model this, we let α and β be functions of M which go to zero as $M \to \infty$. We assumed that $\alpha(M) = o(\beta(M))$ to model the information richness and sparsity. Also when the system has both information-rich users and information-rich items, we assume $\tilde{r}_{um}$ is erased with probability β if either user m is an information-rich user or item m is an information-rich item; and $\tilde{r}_{um}$ is erased with probability α otherwise.

2.1 Remarks on the conditions

We present the conditions above in such a way that the notation in the analysis is simplified. Many of these conditions can be easily relaxed. We next comment on these extensions, for which only minor modifications of the proofs are needed.

1. Conditions 1 and 2: These two conditions have been stated in a very general form and are easily satisfied. For example, note that if the blocks of $\mathbf{B}$ are chosen in some i.i.d. fashion, the conditions would hold asymptotically for large matrices with high probability. Furthermore, the constant μ can be different in the two conditions.
2. Condition 3: The noisy channel can be any channel that guarantees

$$\Pr\left(\tilde{r}_{bm} = \tilde{r}_{vm} | b_{um} = b_{vm}\right) > \Pr\left(\tilde{r}_{bm} = \tilde{r}_{vm} | b_{um} \neq b_{vm}\right)$$

   i.e., when two users have the same preference for a item, they are more likely to give the same rating than when they have different preferences for the item.
3. Conditions 4 and 5: The upper bound η can be a function of M. The α and β in the two conditions can also be different.

4. Cluster sizes: The cluster sizes can be different but of the same order. We also remark that in (Tomozei and Massoulié 2014), K is assumed to be a constant, and in (Barman and Dabeer 2012), PAF requires $K = O\left(\sqrt{M}\right)$. Our co-clustering algorithm, which will be presented in Sect. 4, works for $K = O(M/\log M)$.

Finally, we note that we do not require all the conditions to hold for the results in the paper. We will next present the results when a subset of these conditions hold and the results when all conditions hold.

3 Main results

The focus of this paper is to derive the conditions under which the preference matrix $\mathbf{B}$ can be recovered from the observed rating matrix $\mathbf{R}$, and develop high-performance algorithms. We assume it is a large-scale system and say an event occurs *asymptotically* if it occurs with probability one when $M \to \infty$.

We let Φ denote a matrix completion algorithm, and $\Phi(\mathbf{R})$ denote the recovered matrix under algorithm Φ given observed rating matrix $\mathbf{R}$. Further, we define $X_{\mathbf{R}}$ to be the number of observed ratings in $\mathbf{R}$, i.e.,

$$X_{\mathbf{R}} = \sum_u \sum_m \mathbf{1}_{r_{um} \neq \star},$$

where $\mathbf{1}$ is the indicator function. *Our main results quantify the conditions required to asymptotically recover* $\mathbf{B}$ *from* $\mathbf{R}$ *in terms of the number of observed ratings* $X_{\mathbf{R}}$.

3.1 Clustering for recommendation

We first assume the users are clustered and satisfy the fractional separability condition (1).

Theorem 1 *Assume Conditions* (1), (3) *and* (4) *hold. If* $\alpha \leq \frac{K}{U}$, *then there exists a constant* $\bar{U}$ *such that for any matrix completion algorithm* Φ *and any* $U \geq \bar{U}$, *we can always find a rating matrix* $\mathbf{B}$ *such that*

$$\Pr(\Phi(\mathbf{R}) = \mathbf{B} | \mathbf{B}) \leq 1 - \frac{\delta}{3},$$

where $\delta = (1 - \beta_{\max})^{\eta} e^{-1.1}$.

Note that

$$E[X_{\mathbf{R}}] \leq MK$$

implies that $\alpha \leq \frac{K}{U}$. *So when the number of observed ratings is fewer than* MK, *no matrix completion algorithm can recover all* $\mathbf{B}$*'s accurately.* □

The proof of this theorem is presented in 'Proof of theorem 1' section of Appendix 2. The result is proved by showing that when $\alpha \leq \frac{K}{U}$ and U is sufficiently large, if

$$\Pr(\Phi(\mathbf{R}) = \mathbf{B} | \mathbf{B}) \geq 1 - \frac{e^{-1.1}}{3}$$

for some $\mathbf{B}$, then we can construct $\hat{\mathbf{B}}$ such that

$$\Pr(\Phi(\mathbf{R}) = \hat{\mathbf{B}} | \hat{\mathbf{B}}) \leq 1 - \frac{2e^{-1.1}}{3}.$$

Theorem 2 *Assume Conditions* (1), (3), *and* (4) *hold. If* $\alpha = \omega\left(\frac{K \log M}{M}\right)$ *and* $\alpha\beta = \omega\left(\frac{\log M}{M}\right)$, *then there exists a matrix completion algorithm* Φ *such that given any* $\epsilon > 0$, *there exists* M_ϵ *such that*

$$\Pr(\Phi(\mathbf{R}) = \mathbf{B}|\mathbf{B}) \geq 1 - \epsilon$$

holds for any rating matrix $\mathbf{B}$ *with at least* M_ϵ *items.*

Note that

$$E[X_{\mathbf{R}}] = \omega(MK \log M)$$

implies that $\alpha = \omega\left(\frac{K \log M}{M}\right)$. *So there exists a matrix completion algorithm that can recover* $\mathbf{B}$ *asymptotically when* $\alpha\beta = \omega\left(\frac{\log M}{M}\right)$ *and number of observed ratings is* $\omega(MK \log M)$. □

The proof of this theorem can be found in 'Proof of theorem 2' section of Appendix 2. Theorem 2 is established by presenting an algorithm which recovers the rating matrix asymptotically. This algorithm called User Clustering for Recommendation (UCR) is presented in Sect. 4.1. The algorithm is motivated by the PAF algorithm proposed in (Barman and Dabeer 2012). However, we made some key modifications to exploit the presence of information-rich users. The key steps of our clustering algorithm are summarized below.

(i) User u first compares her/his rating vector with other users, and selects an user who has the highest similarity to her/him (say user v). It can be proved that the selected user is an information-rich user in the same cluster.
(ii) Then the algorithm selects $U/K - 2$ users according to their normalized similarity to user v. It can be proved that these users are the users who are in the same cluster as user v (so in the same cluster as user u).
(iii) For each item m, the algorithm predicts b_{um} via a majority vote among the selected users, including users u and v. The predicted rating is asymptotically correct.

We note that theorems analogous to Theorems 1 and 2 can be established for item clustering. The corresponding algorithm in the case will be called Item Clustering for Recommendation (ICR).

3.2 Co-clustering for recommendation

We now assume both users and items are clustered and satisfy the fractionally separable conditions (1) and (2).

Theorem 3 *Assume Conditions* (1), (2), (3), (4), *and* (5) *hold. If* $\alpha \leq \frac{K^2}{MU}$ *and* $\beta \leq \frac{K}{\eta(M+U)-\eta^2 K}$, *then there exist a constant* $\bar{M}$ *such that for any matrix completion algorithm* Φ *and any* $M \geq \bar{M}$, *we can always find a rating matrix* $\mathbf{B}$ *such that*

$$\Pr(\Phi(\mathbf{R}) = \mathbf{B}|\mathbf{B}) \leq 1 - \frac{\delta}{3},$$

where $\delta = e^{-2.2}$.

Note that

$$E[X_{\mathbf{R}}] \leq K^2$$

Table 1 A summary of the main results

	Clustering	Co-clustering	Matrix completion with rank K
Necessary	$E[X_{\mathbf{R}}] = MK$	$E[X_{\mathbf{R}}] = K^2$	$E[X_{\mathbf{R}}] = \gamma MK \log M$ (Candès and Tao 2010)
Sufficient	$E[X_{\mathbf{R}}] = \omega(MK \log M)$, and $\alpha\beta M = \omega(\log M)$	$E[X_{\mathbf{R}}] = \omega(K^2 \log M)$, and $\alpha\beta M = \omega(\log M)$ or $\beta^2 K\omega(\log M)$	$E[X_{\mathbf{R}}] = \Omega(MK \log^2 M)$ (Recht 2011)

implies that $\alpha \leq \frac{K^2}{UM}$ *and* $\beta \leq \frac{K}{\eta(M+U)-\eta^2 K}$. *So when the number of observed ratings is fewer than* K^2, *no matrix completion algorithm can recover all* $\mathbf{B}'s$ *accurately.* □

The detailed proof is presented in 'Proof of theorem 3' section of Appendix 2.

Theorem 4 *Assume Conditions* (1), (2), (3), (4), *and* (5) *hold. Further, assume the following conditions hold:*

(i) $\alpha = \omega\left(\frac{K^2 \log M}{M^2}\right)$ *or* $\beta = \omega\left(\frac{K \log M}{M}\right)$; *and*

(ii) $\alpha\beta = \omega\left(\frac{\log M}{M}\right)$ *or* $\beta^2 = \omega\left(\frac{\log M}{K}\right)$.

Then there exists a matrix completion algorithm Φ *such that given any* $\epsilon > 0$, *there exists* M_ϵ *such that*

$$\Pr(\Phi(\mathbf{R}) = \mathbf{B}|\mathbf{B}) \geq 1 - \epsilon$$

holds for any rating matrix $\mathbf{B}$ *with at least* M_ϵ *items.*

Note that

$$E[X_{\mathbf{R}}] = \omega(K^2 \log M)$$

implies condition (1), *so* $\mathbf{B}$ *can be asymptotically recovered from* $\mathbf{R}$ *when the number of observed ratings is* $\omega(K^2 \log M)$ *and condition* (2) *holds.* □

Theorem 4 is proved by showing that there exists a co-clustering algorithm that clusters both users and items. Then the preference b_{um} is recovered by a majority vote among all observed r_{vn}s for all v in the same user-cluster as u and all n in the same item-cluster as m. We have relegated the proof of Theorem 4 to the appendix since the main ideas behind the proof are similar to the proof of Theorem 2. The detailed description of the co-clustering algorithm, named Co-Clustering for Recommendation (CoR), is presented in Sect. 4.2.

We summarize our main results, and compare them with corresponding results for matrix completion, in Table 1.

4 Algorithms

4.1 Clustering for recommendation

Before presenting the algorithms, we first introduce the notions of co-rating and similarity. Given users u and v, the co-rating of the two users is defined to be the number of items they both rate:

$$\varphi_{u,v} = \sum_{m=1}^{M} \mathbf{1}_{r_{vm}\neq\star, r_{um}\neq\star};$$

and the similarity of the two users is defined to be the number of items they rate the same minus the number of items they rate differently:

$$\sigma_{u,v} = \sum_{m=1}^{M} \mathbf{1}_{r_{um}=r_{vm}\neq\star} - \sum_{m=1}^{M} \mathbf{1}_{r_{um}\neq r_{vm}, r_{vm}\neq\star, r_{um}\neq\star} = 2\sum_{m=1}^{M} \mathbf{1}_{r_{um}=r_{vm}\neq\star} - \varphi_{u,v}.$$

We further define the normalized similarity of two users to be

$$\tilde{\sigma}_{u,v} = \frac{\sigma_{u,v}}{\varphi_{u,v}} = \frac{2\sum_{m=1}^{M} \mathbf{1}_{r_{um}=r_{vm}\neq\star}}{\varphi_{u,v}} - 1.$$

Similarly, we can define the co-rating, similarity and normalized similarity of two items:

$$\varphi_{m,n} = \sum_{u=1}^{U} \mathbf{1}_{r_{um}\neq\star, r_{un}\neq\star}$$

$$\sigma_{m,n} = 2\sum_{u=1}^{U} \mathbf{1}_{r_{um}=r_{un}\neq\star} - \varphi_{u,v}$$

$$\tilde{\sigma}_{m,n} = \frac{2\sum_{u=1}^{U} \mathbf{1}_{r_{um}=r_{un}\neq\star}}{\varphi_{m,n}} - 1.$$

When users are clustered, the following algorithm exploits the existence of both the cluster structure and information-rich entities to recover the matrix.

User Clustering for Recommendation (UCR)

(i) For user u, the algorithm selects a user v who has the highest similarity to user u, i.e., $v \in \arg\max_{w\neq u} \sigma_{u,w}$.

(ii) The algorithm then selects $\frac{U}{K} - 2$ users in a descending order according to their normalized similarity to user v. Define $\mathcal{F}_u$ to be the set of the selected $\frac{U}{K} - 2$ users, user v and user u.

(iii) For each item m, the preference b_{wm} for $w \in \mathcal{F}_u$ is determined by a majority vote among the users in $\mathcal{F}_u$, i.e.,

$$b_{wm} = \arg\max_{g} \sum_{v\in\mathcal{F}_u} \mathbf{1}_{r_{vm}=g}.$$

When items are clustered, a similar algorithm that clusters items can be used to recover the matrix. The algorithm is named Item Clustering for Recommendation (ICR), and is presented in 'Item Clustering for Recommendation' section of Appendix 1.

4.2 Co-clustering for recommendation

When both users and items are clustered, we propose the following co-clustering algorithm for recovering **B**. For given a (user, item) pair, the algorithm identifies the corresponding user-cluster and item-cluster, and then uses a majority vote within the $\frac{U}{K} \times \frac{M}{K}$ block to recover b_{um}.

Co-Clustering for Recommendation (CoR)

(i) Given a (user, item) pair (u, m), the algorithm calls steps (i) and (ii) of UCR to obtain a set of users $\mathcal{F}_u$; and calls steps (i) and (ii) of ICR to obtain a set of items $\mathcal{N}_m$.
(ii) For each item $n \in \mathcal{N}_m$ and each user $v \in \mathcal{F}_u$, b_{vn} is decided by a majority vote among r_{wl}'s for $w \in \mathcal{F}_u$ and $l \in \mathcal{N}_m$, i.e.,

$$b_{vn} = \arg\max_g \sum_{w \in \mathcal{F}_u} \sum_{l \in \mathcal{N}_m} \mathbf{1}_{r_{wl}=g}.$$

4.3 Hybrid algorithms

As we mentioned in the introduction, in practice, it is hard to verify whether the cluster assumptions hold for the preference matrix since the matrix is unknown except for a few noisy entries. Even if the cluster assumptions hold, it is hard to check whether every user-cluster (item-cluster) contains information-rich users (items). When a user-cluster does not contain an information-rich user, UCR is likely to pick an information-rich user from another cluster in step (i) and results in selecting users from a different cluster in step (ii). So when a user-cluster has no information-rich user, it is better to skip step (i) and select users according to their normalized similarity to user u; or just use PAF.

Since it is impossible to check whether a user-cluster has information-rich users or not, we propose the following hybrid algorithms, which combine three different approaches. We note that the hybrid algorithms are heuristics motivated by the theory developed earlier. Using the hybrid user-clustering for recommendation as an example, it combines the following three approaches: (i) first find an information-rich user and then use users' similarities to the selected information rich-user to find the corresponding user-cluster, (ii) directly use other users' similarities to user u to find the corresponding user-cluster, and (iii) directly use other users' *normalized* similarities to user u to find the corresponding user-cluster. After identifying the three possible clusters, the algorithm aggregates all the users in one cluster into a super-user, and computes the similarity between the super-user and user u. The super-user with the highest similarity is used to recover the ratings of user u. We further modify the definition of normalized similarity because this new normalized similarity works better than the original one in the experiments with real data sets:

$$\tilde{\sigma}_{u,v} = \frac{\sigma_{u,v}}{\sqrt{\sum_{m=1}^{M} \mathbf{1}_{r_{vm} \neq \star}}}.$$

We next present the detailed description of the Hybrid User-Clustering for Recommendation (HUCR) and the Hybrid Co-Clustering for Recommendation (HCoR). The Hybrid Item-Clustering for Recommendation (HICR) is similar to HUCR and is presented in 'Hybrid Item-Clustering for Recommendation' section of Appendix 1.

Hybrid User-Clustering for Recommendation (HUCR)

(i) For each user u, the algorithm calls steps (i) and (ii) of UCR to obtain a set of $T - 1$ users, not including user u. Denote the set by $\mathcal{F}_u^1$.
(ii) The algorithm selects $T - 1$ users in a descending order according to their modified normalized similarity to users u. Denote the set by $\mathcal{F}_u^2$.
(iii) The algorithm selects $T - 1$ users in a descending order according to their similarity to user u. Denote the set by $\mathcal{F}_u^3$.

(iv) The algorithm defines three super-users s_z ($z = 1, 2, 3$) such that the ratings of s_z is determined by a majority vote among the users in $\mathcal{F}_u^z$, i.e.,

$$r_{s_z,m} = \arg\max_g \sum_{v \in \mathcal{F}_u^z} \mathbf{1}_{r_{v,m}=g}.$$

(v) The algorithm selects the super-user who has the highest similarity to user u. Without the loss of generality, assume super-user s_1 is selected, then the ratings of user u is given by $b_{u,m} = r_{s_1,m}$.

Remark Here we could use the cluster size as T. But we use a new variable T here to emphasize the fact that the hybrid algorithms are designed for real datasets where we may not know the cluster size. In practice, we estimate T by noting that the similarity score undergoes a phase transition when too many similar users are selected.

Remark The algorithm uses similarity in step (v). In fact, the three super-users will be information-rich users, so there is no significant difference between using similarity and using normalized similarity.

Hybrid Co-Clustering for Recommendation (HCoR)

(i) For each (user, item) pair (u, m), the algorithm calls steps (i)-(iv) of HUCR to obtain the set of $T - 1$ users forming the super-user who has the highest similarity to user u, denoted by $\mathcal{F}_u$; and steps (i)-(iv) of HICR to obtain the set of $T - 1$ items forming the super-item that has the highest similarity to item m, denoted by $\mathcal{N}_m$.

(ii) The rating r_{mu} is determined as follows:

$$r_{mu} = \arg\max_g \sum_{v \in \mathcal{F}_u} \mathbf{1}_{r_{vm}=g} + \sum_{n \in \mathcal{N}_m} \mathbf{1}_{r_{un}=g} + \sqrt{\sum_{v \in \mathcal{F}_u, n \in \mathcal{N}_m} \mathbf{1}_{r_{vn}=g}}.$$

Remark We used modified majority voting in step (ii) of HCoR because of the following reason: After step (i), the algorithm identifies $T - 1$ users and $T - 1$ items as shown in Fig. 1. The ratings in region 1 (i.e., r_{vm} for $v \in \mathcal{F}_u$) are the ratings given to item m and the ratings in region 2 (i.e., r_{un} for $n \in \mathcal{N}_m$) are the ratings given by user u. The ratings in region 3 (i.e., r_{vn} for $v \in \mathcal{F}_u$ and $n \in \mathcal{N}_m$) are the ratings given by the users similar to user u to the items similar to item m. In our experiments with real datasets, we found that the ratings in region 1 and 2 are more important in predicting b_{um} than the ratings in region 3. Since we have $(T - 1) \times (T - 1)$ entries in region 3 but only $T - 1$ entries in region 1 or 2, so we use the square-root of the votes from region 3 to reduce its weight in the final decision.

5 Performance evaluation

In this section, we evaluate the performance of our hybrid clustering and co-clustering algorithms and compare them with PAF (Barman and Dabeer 2012) and AM (Shen et al. 2012). We tested the algorithms using both the MoiveLens dataset[1] and Netflix dataset.[2] Our main

[1] Available at http://grouplens.org/datasets.

[2] Available at http://www.netflixprize.com.

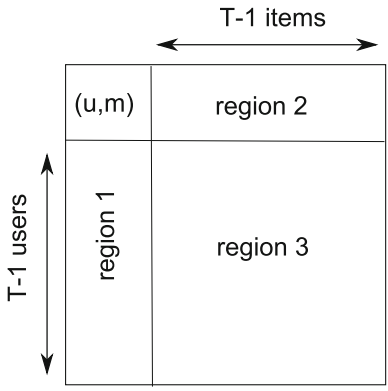


Fig. 1 The block ratings associated with (user, item) pair (u, m)

goal is to recommend to each user only those movies that are of most interest to that user. In other words, we only want to recommend movies to a user that we believe would have been rate highly by that user. Therefore, we quantize the ratings in both datasets so that movies which received a rating greater than 3.5 are reclassified as $+1$ and movies which received a rating of 3.5 or below are reclassified as -1. This binary quantization is also necessary to make a fair comparison with the results in (Barman and Dabeer 2012) since the algorithm there only works for binary ratings. For both datasets, we hide 70 % of the ratings. So 30 % of the ratings were used for training (or predicting); and 70 % of the ratings were used for testing the performance. The following three performance metrics are used in the comparison:

1. **Accuracy at the top** This terminology was introduced in (Boyd et al. 2012) which in our context means the accuracy with which we identify the movies of most interest to each user. In our model, instead of computing accuracy, we compute the error rate (the number of ratings that are not correctly recovered divided by the total number of ratings) when we recommend a few top items to each user, which they may like the most. But we continue to use the term "accuracy at the top" to be consistent with prior literature. Since the goal of a recommendation system is indeed to recommend a few items that a user may like, instead of recovering a user's preference to all items, we view this performance metric as the most important one among the three metrics we consider in this paper. In HCoR and PAF, the top-items were selected based on majority voting within each cluster, and in AM, the top-items were selected based on the recovered value. To make a fair comparison among the five algorithms, we restricted the algorithms to only recommend those items whose ratings were given in the dataset but hidden for testing.
2. **Accuracy for information-sparse users** In real datasets, a majority of users only rate a few items. For example, in the MovieLens dataset, more than 73.69 % users only rated fewer than 200 movies, and their ratings only consist of 34.32 % of the total ratings. The accuracy for information-sparse users measures the error rate for these information-sparse users who form the majority of the user population. Note the overall error rate is biased towards those information-rich users who rated a lot of movies (since they account for 65.68 % of the ratings in the MovieLens data, for example).
3. **Overall accuracy** The overall error rate when recovering the rating matrix. We include this metric for completeness.

Before presenting the detailed results, for the convenience of the reader, we summarize the key observations:

1. In all of the datasets that we have studied, our co-clustering algorithm HCoR consistently performs better than the previously-known PAF and AM algorithms. When noise is added to the datasets, the performance difference between our algorithm and the AM algorithm increases even more.
2. If the goal is to recommend a small number of movies to each user (i.e., concerning accuracy at the top), then even the user clustering algorithm HUCR performs better than PAF and AM, but worse than HCoR. Since HUCR has a lower computational complexity than HCoR, when the running time is a concern, we can use HUCR, instead of HCoR, to recommend a few movies to each user.
3. We have not shown other studies that we have conducted on the datasets due to space limitations, but we briefly mention them here. If the goal is to recommend a few users for each item (for example, a new item may be targeted to a few users), then item clustering performs better than PAF and AM, but HCoR still continues to perform the best. Also, simple clustering techniques such as spectral clustering, following by a majority vote within clusters, do not work as well as any of the other algorithms. When the running time is a concern, HICR can be used to recommend a few users for each item.

In the following subsections, we present substantial experimental studies that support our key observations 1 and 2.

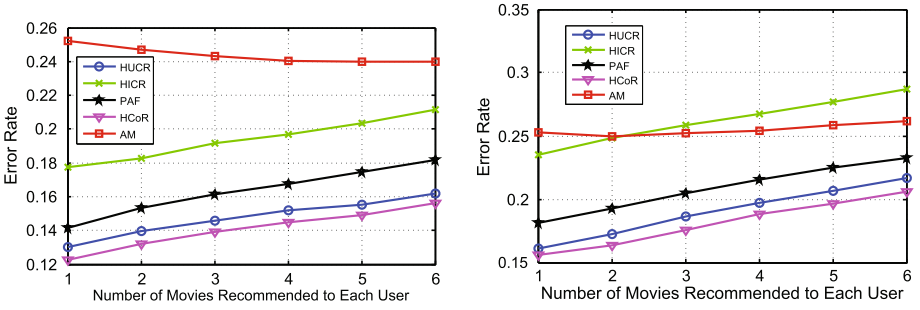
5.1 MovieLens dataset without noise

We conducted experiments on the MovieLens dataset, which has 3,952 movies, 6,040 users and 1,000,209 ratings. So users rate about 4 % of the movies in the MovieLens dataset.

We first evaluated the accuracy at the top. Figure 2a shows the error rate when we recommended x movies to each user for $x = 1, 2, \ldots, 6$. We can see that HCoR performs better than all other algorithms, and HUCR has a similar error rate as HCoR. In particular, when one movie is recommended to each user, HCoR has an error rate of 12.27 % while AM has an error rate of 25.22 % and PAF has an error rate of 14 %.

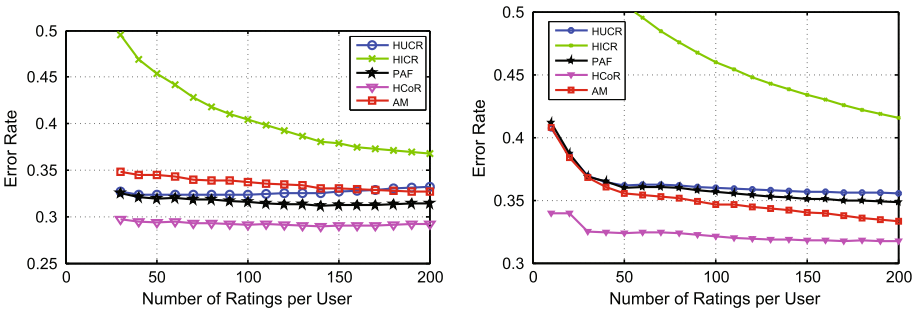
We then evaluated the accuracy for information-sparse users. Figure 3a shows the error rate for the users who rate between less than x movies, for $x = 30, 40, \ldots, 200$. We can see from the figure that HCoR has the lowest error rate. For example, for users who rated less than 30 movies, HCoR has an error rate of 29.72 % while AM has an error rate of 34.81 %.

For completeness, we also summarize the overall error rates in Table 2. HCoR has the lowest error rate.



(a) The MovieLens dataset **(b)** The Netflix dataset

Fig. 2 Accuracy at the *top* for two datasets. The figures show the *error rates* when we recommend x movies to each user. **a** The MovieLens dataset. **b** The Netflix dataset



(a) The MovieLens dataset **(b)** The Netflix dataset

Fig. 3 Accuracy for information-sparse users for two datasets. The figures show the *error rates* of users who rate different numbers of movies. **a** The MovieLens dataset. **b** The Netflix dataset

Table 2 The overall error rates

	HUCR (%)	HICR (%)	HCoR (%)	PAF (%)	AM (%)
MovieLens	34.69	32.87	29.2	32.07	30.62
NetFlix	36.49	34.72	31.11	35.51	31.11
MovieLens with noise	40.95	41.95	32.55	35.64	38.46
NetFlix with noise	42.47	43.51	34.28	39.06	38.88

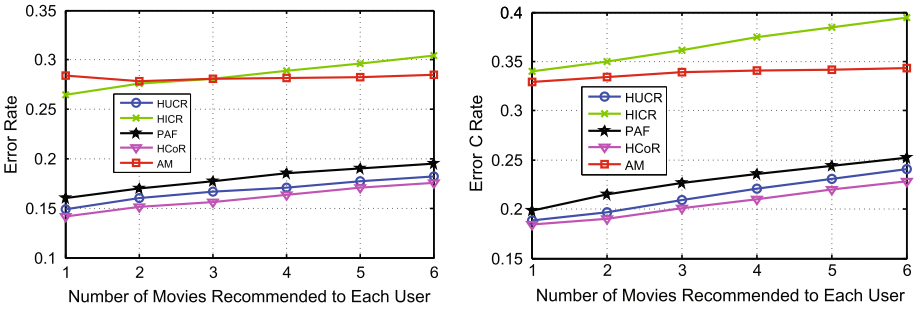
Remark It is possible that some ratings cannot be obtained under HUCR, HICR, HCoR and PAF, e.g., b_{um} cannot be obtained when none of selected users in step (ii) of HUCR rated movie m. When it occurs, we counted it as an error. So we made very conservative calculations in computing the error rates for HUCR, HICR, HCoR and PAF.

5.2 Netflix dataset without noise

We conducted experiments on the Netflix dataset, which has 17,770 movies, 480,189 users and 100,480,507 ratings. We used all movies but randomly selected 10,000 users, which gives 2,097,444 ratings for our experiment. The reason that we selected 10,000 users is that, otherwise, the dataset is too large to handle without the use of special purpose computers. In particular, it is not clear how one would implement that AM algorithm on the full dataset since the first step of that algorithm requires one to perform an SVD which is not possible using a computer with 8G RAM, 2.5 Ghz processor that we used.

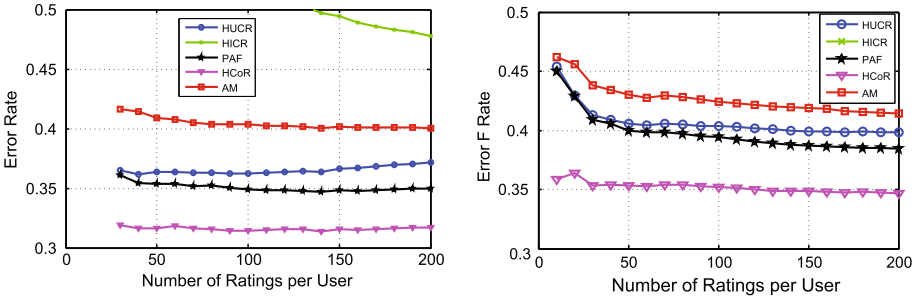
Figure 2b shows the accuracy at the top, i.e., the error rate when we recommended x movies to each user for $x = 1, 2, \ldots, 6$. We can see that HCoR performs better than all other algorithms, and HUCR has a similar error rate as HCoR. When one movie is recommended to each user, HCoR has an error rate of 15.58 % while AM has an error rate of 25.29 %.

We then evaluated the accuracy for information-sparse users. Figure 3b shows the error rate for the users who rate less than x items, for $x = 10, 20, 30, \ldots, 200$. Among the 10,000 randomly selected users, 70 % of them rated no more than 200 movies. We can see from the figure that HCoR has the lowest error rate. In particular, for the users who rate less than 10 items, HCoR has an error rate of 33.95 % while the error rate of AM is 40.81 %.



(a) The MovieLens data with noise **(b)** The Netflix data with noise

Fig. 4 Accuracy at the *top* for two datasets with noise. The figures show the *error rates* when we recommend x movies to each user. **a** The MovieLens data with noise. **b** The Netflix data with noise



(a) The MovieLens data with noise **(b)** The Netflix data with noise

Fig. 5 Accuracy for information-sparse users for two datasets with noise. The figures show the *error rates* of users who rate different numbers of movies. **a** The MovieLens data with noise. **b** The Netflix data with noise

Table 2 summarizes the overall error rates. AM and HCoR have the lowest overall error rate.

5.3 MovieLens dataset with noise

Our theoretical results suggested that our clustering and co-clustering algorithms are robust to noise. In this set of experiments, we independently flipped each un-hidden rating with probability 0.2, and then evaluated the performance of our clustering and co-clustering algorithms. The result for the accuracy at the top is shown in Fig. 4a. We can see that HCoR performs better than all other algorithms. When one movie is recommended to each user, HCoR has an error rate of 14.19 % while AM has an error rate of 28.41 %. Comparing to the noise-free case, the error rate of HCoR increases by 1.92 %, and the error rate of AM increases by 3.19 %.

The results for the accuracy for information-sparse users are presented in Fig. 5a. HCoR has the lowest error rate. For users who rated less than 30 movies, HCoR has an error rate of 31.87 % while AM has an error rate of 41.67 %. Comparing to the noise-free case, the error rate of HCoR increases only by 2.15 %, but the error rate of AM increases by 6.86 %.

For completeness, we also summarize the overall error rates in Table 2. HCoR has the lowest error rate, and AM has a significantly higher error rate in this case. Note that HCoR and AM have similar overall error rates in the noise-free case. From this set of experiments, we can see that HCoR is more robust to noise than AM.

5.4 Netflix dataset with noise

In this set of experiments, we flipped each un-hidden rating of the Netflix dataset with probability 0.2. Figure 4 shows the accuracy at the top with noisy entries. HCoR performs the best. When one movie is recommended to each user, HCoR has an error rate of 18.4 % while AM has an error rate of 32.89 %. So the error rate of AM increases by 7.6 % comparing to the noise-free case while the error rate of HCoR increases only by 2.82 %.

The results for the accuracy for information-sparse users is shown in Fig. 5b. HCoR has the lowest error rate. For the users who rate less than 10 items, HCoR has an error rate of 35.88 % while the error rate of AM is 46.71 %. Comparing to the noise-free case, the error rate of HCoR increases by 1.93 % while the error rate of AM increases by 5.9 %. HICR is not shown in the figure since the error rate is more than 50 %.

Table 2 summarizes the overall error rates. HCoR have the lowest overall error rate, which is 4.6 % lower than that of AM. Note that HCoR and AM have similar error rates in the noise-free case. From this set of experiments, we again see that HCoR is more robust to noise than AM.

6 Conclusion

In this paper, we considered both the clustering and co-clustering models for collaborative filtering in the presence of information-rich users and items. We developed similarity based algorithms to exploit the presence of information-rich entities. When users/items are clustered, our clustering algorithm can recover the rating matrix with $\omega(MK\log M)$ noisy entries; and when both users and items are clustered, our co-clustering can recover the rating matrix with $K^2\log M$ noisy entries when K is sufficiently large. We compared our co-clustering algorithm with PAF and AM by applying them to the MovieLens and Netflix data sets. In the experiments, our proposed algorithm HCoR has significantly lower error rates when recommending a few items to each user and when recommending items to the majority of users who only rated a few items. Due to space limitations, we only presented the proofs for the basic models in Appendix 2. The extensions mentioned in the remarks in Sect. 2 are straightforward. Furthermore, instead of assuming the cluster size is given in the clustering and co-clustering algorithms, the algorithms can estimate the erasure probability α using the number of observed ratings, i.e., $\alpha = 1 - \frac{\mathbf{X_R}}{UM}$, from which, the algorithms can further estimate the cluster-size. This estimation can be proved to be asymptotically correct.

Acknowledgments Research supported in part by AFOSR grant FA 9550-10-1-0573 and NSF grants ECCS-1202065 and ECCS-1255425. We thank Prof. Olgica Milenkovic for comments on an earlier version of the paper.

Appendix 1: Item based algorithms

Item clustering for recommendation (ICR)

Item Clustering for Recommendation (ICR)

(i) For item m, the algorithm selects an item n that has the highest similarity to item m.
(ii) The algorithm then selects $\frac{M}{K} - 2$ items in a descending order according to their normalized similarity to item n. Define $\mathcal{N}_m$ to be the set of the selected $\frac{M}{K} - 2$ items, item m and item n.
(iii) For each user u, the preference b_{ul} for $l \in \mathcal{N}_m$ is determined by a majority vote among the items in $\mathcal{F}_m$, i.e., $b_{ul} = \arg\max_g \sum_{n \in \mathcal{F}_m} \mathbf{1}_{r_{un}=g}$.

Hybrid item-clustering for recommendation (HICR)

Hybrid Item-Clustering for Recommendation (HICR)

(i) For each item m, the algorithm calls steps (i) and (ii) of ICR to obtain a set of $T - 1$ items, not including item m. Denote the set by $\mathcal{N}_m^1$.
(ii) The algorithm selects $T - 1$ items in a descending order according to their modified normalized similarity to item m. Denote the set by $\mathcal{N}_m^2$.
(iii) The algorithm selects $T - 1$ items in a descending order according to their similarity to item m. Denote the set by $\mathcal{N}_m^3$.
(iv) The algorithm defines three super-items s_z $(z = 1, 2, 3)$ such that the ratings of s_z is determined by a majority voting among the items in $\mathcal{N}_m^z$, i.e., $r_{u,s_z} = \arg\max_g \sum_{n \in \mathcal{N}_m^z} \mathbf{1}_{r_{u,n}=g}$.
(v) The algorithm selects the super-item that has the highest similarity to item m. Without the loss of generality, assume super-item s_1 is selected, then the ratings of item m is given by $b_{u,m} = r_{u,s_1}$.

Appendix 2: Proofs

Notation

- U : the number of users
- u, v, and w : the user index and $u, v, w \in \{1, \ldots, U\}$
- M : the number of items
- m, n, and l : the item index and $m, n \in \{1, \ldots, M\}$
- K : the number of clusters
- k : the cluster index
- G : the number of preference (rating) levels
- $\mathbf{B}$: the preference matrix
- $\mathbf{R}$: the observed rating matrix
- $\sigma_{u,v}$: the similarity between user u and user v

- $\varphi_{u,v}$: the number of items co-rated by users u and v
- $\sigma_{m,n}$: the similarity between item m and item n
- $\varphi_{m,n}$: the number of users who rate both items m and n
- $1-\alpha$: the erasure probability of an information-sparse user (item)
- $1-\beta$: the erasure probability of an information-rich user (item)

Given non-negative functions $f(M)$ and $g(M)$, we also use the following order notation throughout the paper.

- $f(M) = O(g(M))$ means there exist positive constants c and $\tilde{M}$ such that $f(M) \leq cg(M)$ for all $M \geq \tilde{M}$.
- $f(M) = \Omega(g(M))$ means there exist positive constants c and $\tilde{M}$ such that $f(M) \geq cg(M)$ for all $M \geq \tilde{M}$. Namely, $g(M) = O(f(M))$.
- $f(M) = \Theta(g(M))$ means that both $f(M) = \Omega(g(M))$ and $f(M) = O(g(M))$ hold.
- $f(M) = o(g(M))$ means that $\lim_{M\to\infty} f(M)/g(M) = 0$.
- $f(M) = \omega(g(M))$ means that $\lim_{M\to\infty} g(M)/f(M) = 0$. Namely, $g(M) = o(f(M))$.

Proof of Theorem 1

Recall that an information-rich user's rating is erased with probability $1-\beta$, an information-sparse user's rating is erased with probability $1-\alpha$, and the number of information-rich users in each cluster is upper bounded by a constant η.

Since $\lim_{U\to\infty}\left(1-\frac{K}{U}\right)^{\frac{U}{K}} = e^{-1}$ and $U/K = \Omega(\log U)$, there exists a sufficiently large $\bar{U}$ such that for any $U \geq \bar{U}$,

$$\left(1-\frac{K}{U}\right)^{\frac{U}{K}} \geq e^{-1.1}. \tag{1}$$

Now consider the case $U \geq \bar{U}$. If the theorem does not hold, then there exists a policy $\hat{\Phi}$ such that

$$\Pr(\hat{\Phi}(\mathbf{R}) = \mathbf{B}|\mathbf{B}) > 1-\frac{\delta}{3} \tag{2}$$

for all $\mathbf{B}$'s.

We define a set $\mathcal{R}$ such that $\mathbf{R} \in \mathcal{R}$ if in $\mathbf{R}$, all ratings of the first item given by the users of the first cluster are erased. Note that

$$\Pr(\mathbf{R} \in \mathcal{R}|\mathbf{B}) \geq (1-\beta)^{\eta}(1-\alpha)^{\frac{U}{K}-\eta} = \left(\frac{1-\beta}{1-\alpha}\right)^{\eta}(1-\alpha)^{\frac{U}{K}}.$$

Given $\alpha \leq \frac{K}{U}$ and $U \geq \bar{U}$, we have

$$\Pr(\mathbf{R} \in \mathcal{R}|\mathbf{B}) \geq (1-\beta_{\max})^{\eta}e^{-1.1} = \delta,$$

and

$$\Pr(\mathbf{R} \notin \mathcal{R}|\mathbf{B}) \leq 1-\delta. \tag{3}$$

Now given a preference matrix $\mathbf{B}$, we construct $\hat{\mathbf{B}}$ such that it agrees with $\mathbf{B}$ on all entries except on the ratings to the first item given by the users in the first cluster. In other words, $\hat{b}_{nu} \neq b_{nu}$ if $n = 1$ and $\lceil uK/U \rceil = 1$; and $\hat{b}_{nu} = b_{nu}$ otherwise. It is easy to verify that $\hat{\mathbf{B}}$ satisfies the fractionally separable condition for users (Condition (1)) since it changes only one moving rating for the users in the first cluster. Furthermore, for any $\mathbf{R} \in \mathcal{R}$, we have

$$\Pr(\mathbf{R}|\mathbf{B}) = \Pr\left(\mathbf{R}|\hat{\mathbf{B}}\right). \tag{4}$$

Now we consider the probability of recovering $\hat{\mathbf{B}}$ under $\hat{\Phi}$, and have

$$\begin{aligned}
&\Pr\left(\hat{\Phi}(\mathbf{R}) = \hat{\mathbf{B}} \middle| \hat{\mathbf{B}}\right) \\
&= \Pr\left((\hat{\Phi}(\mathbf{R}) = \hat{\mathbf{B}}) \cap (\mathbf{R} \in \mathcal{R}) \middle| \hat{\mathbf{B}}\right) + \Pr\left((\hat{\Phi}(\mathbf{R}) = \hat{\mathbf{B}}) \cap (\mathbf{R} \notin \mathcal{R}) \middle| \hat{\mathbf{B}}\right) \\
&\leq \Pr\left((\hat{\Phi}(\mathbf{R}) \neq \mathbf{B}) \cap (\mathbf{R} \in \mathcal{R}) \middle| \hat{\mathbf{B}}\right) + \Pr\left(\mathbf{R} \notin \mathcal{R} | \hat{\mathbf{B}}\right) \\
&=_{(a)} \Pr\left((\hat{\Phi}(\mathbf{R}) \neq \mathbf{B}) \cap (\mathbf{R} \in \mathcal{R}) \middle| \mathbf{B}\right) + \Pr\left(\mathbf{R} \notin \mathcal{R} | \hat{\mathbf{B}}\right) \\
&\leq \Pr\left(\hat{\Phi}(\mathbf{R}) \neq \mathbf{B}) \middle| \mathbf{B}\right) + \Pr\left(\mathbf{R} \notin \mathcal{R} | \hat{\mathbf{B}}\right) \\
&\leq_{(b)} \frac{\delta}{3} + 1 - \delta \\
&= 1 - \frac{2\delta}{3}.
\end{aligned}$$

where equality (a) holds due to Eq. (4), and inequality (b) yields from inequalities (2) and (3). The inequality above contradicts (2), so the theorem holds.

Proof of Theorem 2

We first calculate the expectation of the similarity σ_{uv} in the following cases:

– *Case 1* u and v are two different information-rich users in the same cluster. In this case, we have

$$\begin{aligned}
E[\sigma_{uv}] &= 2M\beta^2\left(p^2 + (G-1)\left(\frac{1-p}{G-1}\right)^2\right) - M\beta^2 \\
&= 2M\beta^2\left(p^2 + \frac{(1-p)^2}{G-1}\right) - M\beta^2,
\end{aligned}$$

where β^2 is the probability the two users' ratings to item m are not erased, p^2 is the probability that the observed ratings of the two users are their true preference, and $(G-1)\left(\frac{1-p}{G-1}\right)^2$ is the probability that the observed ratings of the two users are the same but not their true preference. We define

$$z_1 = \left(p^2 + \frac{(1-p)^2}{G-1}\right),$$

so $E[\sigma_{uv}] = M\beta^2(2z_1 - 1)$ in this case.

– *Case 2* u and v are in the same cluster, u is an information-rich user, and v is an information-sparse user. In this case, we have

$$\begin{aligned}
E[\sigma_{uv}] &= 2M\alpha\beta\left(p^2 + (G-1)\left(\frac{1-p}{G-1}\right)^2\right) - M\alpha\beta \\
&= 2M\alpha\beta\left(p^2 + \frac{(1-p)^2}{G-1}\right) - M\alpha\beta \\
&= M\alpha\beta(2z_1 - 1),
\end{aligned}$$

where $\alpha\beta$ is the probability the two users' ratings to item m are not erased.

– *Case 3* u and v are in different clusters, and both are information-rich users. In this case, under the biased rating condition (3), we can obtain

$$E[\sigma_{uv}] \leq 2\mu M\beta^2 z_1 + 2(1-\mu)M\beta^2\left(2p\frac{1-p}{G-1} + (G-2)\left(\frac{1-p}{G-1}\right)^2\right) - M\beta^2$$
$$= 2\mu M\beta^2 z_1 + 2(1-\mu)M\beta^2\left(\frac{1-p^2}{G-1} - \left(\frac{1-p}{G-1}\right)^2\right) - M\beta^2.$$

We define

$$z_2 = \left(\frac{1-p^2}{G-1} - \left(\frac{1-p}{G-1}\right)^2\right),$$

so

$$M\beta^2(2z_2 - 1) \leq E[\sigma_{uv}] \leq M\beta^2(2\mu z_1 + 2(1-\mu)z_2 - 1)$$

in this case.

– *Case 4* u and v are in different clusters, u is an information-rich user, and v is an information-sparse user. In this case, we have

$$M\alpha\beta(2z_2 - 1) \leq E[\sigma_{uv}] \leq M\alpha\beta(2\mu z_1 + 2(1-\mu)z_2 - 1).$$

– *Case 5* u and v are in the same cluster, and are both information-sparse users. In this case, we have

$$E[\sigma_{uv}] = M\alpha^2(2z_1 - 1).$$

– *Case 6* u and v are in different clusters, and are both information-sparse users. In this case, we have

$$E[\sigma_{uv}] \leq M\alpha^2(2\mu z_1 + 2(1-\mu)z_2 - 1).$$

Note that $z_1 - z_2 = \left(p - \frac{1-p}{G-1}\right)^2$, so $z_1 > z_2$ when $p > \frac{1}{G}$. Now we define $\mathcal{P}_j$ to be the set of (u, v) pairs considered in case j above.

Recall that we assume $\alpha\beta M = \omega(\log U)$ and $\frac{\alpha}{\beta} = o(1)$. Given any $\epsilon > 0$, we define event $\mathcal{E}_j$ for $j \in \{1, 2, 3, 4\}$ to be

$$\mathcal{E}_j = \left\{(1-\epsilon)E[\sigma_{uv}] \leq \sigma_{uv} \leq (1+\epsilon)E[\sigma_{uv}] \quad \forall\, (u, v) \in \mathcal{P}_j\right\},$$

and $\mathcal{E}_j$ for $j = 5, 6$ to be

$$\mathcal{E}_5 = \{\sigma_{uv} \leq 0.1M\alpha\beta(2z_1 - 1) \quad \forall\, (u, v) \in \mathcal{P}_5\}$$
$$\mathcal{E}_6 = \{\sigma_{uv} \leq 0.1M\alpha\beta\,(2\mu z_1 + 2(1-\mu)z_2 - 1) \quad \forall\, (u, v) \in \mathcal{P}_6\},$$

Using the Chernoff bound (Mitzenmacher and Upfal 2005, Thereorem 4.4 and Theorem 4.5), we now prove that when M is sufficiently large,

$$\Pr\left(\mathcal{E}_j\right) \geq 1 - \frac{1}{M} \tag{5}$$

for any j. We establish this result by considering the following cases:

– First consider Case 2 in which users u and v are in the same cluster. Recall that

$$\sigma_{u,v} = 2\sum_{m=1}^{M} \mathbf{1}_{r_{um}=r_{vm}\neq\star} - \sum_{m=1}^{M} \mathbf{1}_{r_{vm}\neq\star, r_{um}\neq\star}.$$

In this case, $\mathbf{1}_{r_{vm}\neq\star, r_{um}\neq\star}$'s are identically and independently distributed (i.i.d.) Bernoulli random variables (across m), and $\mathbf{1}_{r_{vm}=r_{um}\neq\star}$'s are i.i.d. Bernoulli random variables as well. Applying the Chernoff bound to each of them, we obtain

$$\Pr\left(\left|2\sum_{m=1}^{M} \mathbf{1}_{r_{um}=r_{vm}\neq\star} - 2M\alpha\beta z_1\right| \leq \frac{\epsilon}{2} M\alpha\beta(2z_1-1)\right) \geq 1 - 2\exp\left(-\frac{\epsilon^2(2z_1-1)^2}{24z_1} M\alpha\beta\right)$$

$$\Pr\left(\left|\sum_{m=1}^{M} \mathbf{1}_{r_{vm}\neq\star, r_{um}\neq\star} - M\alpha\beta\right| \leq \frac{\epsilon}{2} M\alpha\beta(2z_1-1)\right) \geq 1 - 2\exp\left(-\frac{\epsilon^2(2z_1-1)^2}{12} M\alpha\beta\right).$$

Note that $2z_1 > 1$. Combining the two inequalities above, we further obtain

$$\Pr\left(|\sigma_{uv} - E[\sigma_{uv}]| \leq \epsilon E[\sigma_{uv}]\right) \geq 1 - 4\exp\left(-\frac{\epsilon^2(2z_1-1)^2}{24z_1} M\alpha\beta\right).$$

Now based on the fact that $|\mathcal{P}_j| \leq U^2$ for any j, we have

$$\begin{aligned}\Pr(\mathcal{E}_2) &= \Pr\left(|\sigma_{uv} - E[\sigma_{uv}]| \leq \epsilon E[\sigma_{uv}],\ \forall\ (u,v) \in \mathcal{P}_2\right)\\ &\geq 1 - 4U^2\exp\left(-\frac{\epsilon^2(2z_1-1)^2}{24z_1} M\alpha\beta\right)\\ &= 1 - 4\exp\left(-\frac{\epsilon^2(2z_1-1)^2}{24z_1} M\alpha\beta + 2\log U\right).\end{aligned}$$

Since $\alpha\beta M = \omega(\log M)$ and $U = \Theta(M)$, when M is sufficiently large, we obtain

$$\Pr(\mathcal{E}_2) \geq 1 - \frac{1}{M}.$$

The proof for Case 1 is similar.

– Next consider Cases 3 and 4, where the two users are in different clusters. Use Case 4 as an example. We assume users u and v have the same preference on items $1, \ldots, \mu_1 M$, and different preference on items $\mu_1 M + 1, \ldots, M$, where $\mu_1 < \mu$. Then for $m = 1, \ldots, \mu_1 M$, $\mathbf{1}_{r_{vm}\neq\star, r_{um}\neq\star}$'s are i.i.d. Bernoulli random variables and $\mathbf{1}_{r_{vm}=r_{um}\neq\star}$'s are i.i.d. Bernoulli random variables. Similar results hold when $m = \mu_1 M + 1, \ldots, M$. We can then prove inequality (5) by applying the Chernoff bound to the two cases separately.
– For Case 5, we define a new user w who is in the same cluster with user v and associated with an erasure probability $1 - 0.05\beta$. Since $\alpha = o(\beta)$, we have for any $A > 0$,

$$\Pr(\sigma_{wv} \geq A) \geq \Pr(\sigma_{uv} \geq A).$$

Then $\Pr(\mathcal{E}_5) \geq 1 - \frac{1}{M}$ can be proved by using the Chernoff bound to lower bound the probability that $\sigma_{wv} \leq 0.1M\alpha\beta(2z_1 - 1)$. The proof for Case 6 is similar.

We further consider co-rating of two users u and v ($\varphi_{u,v}$) in the following two scenarios:

– Scenario 1: u and v are both information-rich users. In this scenario, we have

$$E[\varphi_{uv}] = M\beta^2. \tag{6}$$

– Scenario 2: u is an information-rich user and v is an information-sparse user. In this scenario, we have

$$E[\varphi_{uv}] = M\alpha\beta. \tag{7}$$

We now define $\mathcal{Q}_1$ to be the set of (u, v) pairs in scenario 1, and $\mathcal{Q}_2$ to be the set of (u, v) pairs in scenario 2. We define

$$\mathcal{F}_j = \left\{(1-\epsilon)E[\varphi_{uv}] \le \varphi_{uv} \le (1+\epsilon)E[\varphi_{uv}] \quad \forall\, (u,v) \in \mathcal{Q}_j\right\}$$

for $j = 1, 2$. Based on the Chernoff bound, we have that when M is sufficiently large for any j,

$$\Pr\left(\mathcal{F}_j\right) \ge 1 - \frac{1}{M}.$$

Without the loss of generality, we assume $2\mu z_1 + 2(1-\mu)z_2 > 1$.[3] We choose $\epsilon \in (0, 1)$ such that

$$\frac{(1-\epsilon)^2}{(1+\epsilon)^2}\frac{2z_1 - 1}{2\mu z_1 + 2(1-\mu)z_2 - 1} > 1.$$

Such an ϵ exists because $z_1 > z_2$. We further assume $\mathcal{E}_j (j = 1, 2, 3, 4, 5, 6)$ and $\mathcal{F}_j (j = 1, 2)$ all occur. Now consider step (i) of the algorithm, if u is an information-rich user, then the similarity between u and v is

$$\sigma_{uv} \begin{cases} \ge (1-\epsilon)M\beta^2(2z_1 - 1), & \text{case 1;} \\ \le (1+\epsilon)M\beta^2(2\mu z_1 + 2(1-\mu)z_2 - 1), & \text{case 3;} \\ \le (1+\epsilon)M\alpha\beta(2z_1 - 1), & \text{case 2;} \\ \le (1+\epsilon)M\alpha\beta(2\mu z_1 + 2(1-\mu)z_2 - 1), & \text{case 4.} \end{cases}$$

Since σ_{uv} is the largest when v is an information-rich user in the same cluster, an information-rich user in the same cluster is picked in step (i) of the algorithm.

If u is an information-sparse user, we have

$$\sigma_{uv} \begin{cases} \ge (1-\epsilon)M\alpha\beta(2z_1 - 1), & \text{case 2;} \\ \le (1+\epsilon)M\alpha\beta(2\mu z_1 + 2(1-\mu)z_2 - 1), & \text{case 3;} \\ \le 0.1M\alpha\beta(2z_2 - 1), & \text{case 5;} \\ \le 0.1M\alpha\beta(2\mu z_1 + 2(1-\mu)z_2 - 1), & \text{case 6.} \end{cases}$$

Again σ_{uv} is the largest when v is an information-rich user in the same cluster, so an information-rich user in the same cluster is picked in step (i) of the algorithm.

Now given v is an information-rich user, based on Eqs. (6) and (7), the normalized similarity $\tilde{\sigma}_{vw}$ satisfies

$$\tilde{\sigma}_{vw} \begin{cases} \ge (1-\epsilon)^2(2z_1 - 1), & \text{case 1;} \\ \ge (1-\epsilon)^2(2z_1 - 1), & \text{case 2;} \\ \le (1+\epsilon)^2(2\mu z_1 + 2(1-\mu)z_2 - 1), & \text{case 3;} \\ \le (1+\epsilon)^2(2\mu z_1 + 2(1-\mu)z_2 - 1), & \text{case 4.} \end{cases}$$

So the normalized similarity when w is in the same cluster as v is larger than the similarity when w is not in the same cluster. Therefore in step (i) of the algorithm, all users in the same cluster as v are selected. v and u are in the same cluster, so at the end of step (ii), all users are in user u's cluster are selected.

[3] The other cases can be proved following similar steps.

Now consider the ratings of item m given by user-cluster k and define

$$M_{m,k,g} = \sum_{u:\lceil uK/U\rceil=k} \mathbf{1}_{r_{um}=g}$$

to be the number of users in cluster k who give g to item m. With a slight abuse of notation, let b_{km} to be the true preference of users in cluster k to item m, so we have

$$E\left[M_{m,k,g}\right] \begin{cases} \geq \left(2\beta + \left(\frac{U}{K} - 2\right)\alpha\right) p, & g = b_{km} \\ \leq \left(\eta\beta + \left(\frac{U}{K} - \eta\right)\alpha\right) \frac{1-p}{G-1}, & g \neq b_{km} \end{cases}$$

Define $\mathcal{G}$ to be the event that a majority voting within a user-cluster gives the true preference of an item for all items and user-clusters, i.e.,

$$\mathcal{G} = \{b_{km} = \arg\max_g M_{m,k,g} \quad \forall\, m, k\}.$$

Now when $\frac{\alpha U}{K} = \omega(\log M)$, using the Chernoff bound, it is easy to verify that

$$\Pr(\mathcal{G}) \geq 1 - \frac{1}{M}.$$

Now when $\mathcal{E}_j (j = 1, 2, 3, 4, 5)$ and $\mathcal{F}_j (j = 1, 2)$ occur, the users are clustered correctly by the algorithm; and when $\mathcal{G}$ occurs, a majority voting within the cluster produces the true preference. Therefore, the theorem holds.

Proof of Theorem 3

Given $\alpha \leq \frac{K^2}{UM}, \beta \leq \frac{K}{\eta(M+U)-\eta^2 K}$ and a constant η, there exists $\bar{M}$ such that for any $M \geq \bar{M}$,

$$(1-\alpha)^{\left(\frac{U}{K}-\eta\right)\left(\frac{M}{K}-\eta\right)} \geq e^{-1.1},$$

and

$$(1-\beta)^{\frac{\eta M}{K}+\frac{\eta U}{K}-\eta^2} \geq e^{-1.1}.$$

Now consider the case $M \geq \bar{M}$ and $K = \Theta(\log M)$. If the theorem does not hold, then there exists a policy $\hat{\Phi}$ such that

$$\Pr(\hat{\Phi}(\mathbf{R}) = \mathbf{B}|\mathbf{B}) > 1 - \frac{\delta}{3} \tag{8}$$

for all $\mathbf{B}$'s.

We define a set $\mathcal{R}$ such that $\mathbf{R} \in \mathcal{R}$ if in $\mathbf{R}$, all ratings of the items in the first item-cluster given by the users of the first cluster are erased. Note that when $M \geq \bar{M}$, we have

$$\begin{aligned} \Pr(\mathbf{R} \in \mathcal{R}|\mathbf{B}) &\geq (1-\beta)^{\frac{\eta M}{K}+\frac{\eta U}{K}-\eta^2}(1-\alpha)^{\left(\frac{U}{K}-\eta\right)\left(\frac{M}{K}-\eta\right)} \\ &\geq e^{-2.2} = \delta. \end{aligned} \tag{9}$$

Now given a preference matrix $\mathbf{B}$, we construct $\hat{\mathbf{B}}$ such that it agrees with $\mathbf{B}$ on all entries except on the rating to the first item-cluster given by the first user-cluster. In other words, $\hat{b}_{nu} \neq b_{nu}$ if $\lceil nK/M\rceil = 1$ and $\lceil uK/U\rceil = 1$; and $\hat{b}_{nu} = b_{nu}$ otherwise. It is easy to verify that $\hat{\mathbf{B}}$ satisfies the fractionally separable conditions both for users and items (Conditions (1) and (2)) as long as $\hat{\mathbf{B}}$ satisfies the two fractionally separable conditions because the

construction of $\hat{\mathbf{B}}$ changes only the rating of the first (item-cluster, user-cluster) pair and $K = \Theta(\log M)$. Furthermore, for any $\mathbf{R} \in \mathcal{R}$, we have

$$\Pr(\mathbf{R}|\mathbf{B}) = \Pr\left(\mathbf{R}|\hat{\mathbf{B}}\right). \tag{10}$$

Following the same argument in the proof of Theorem 1, we have

$$\begin{aligned}\Pr\left(\hat{\Phi}(\mathbf{R}) = \hat{\mathbf{B}} \middle| \hat{\mathbf{B}}\right) &\leq \Pr\left(\hat{\Phi}(\mathbf{R}) \neq \mathbf{B}) \middle| \mathbf{B}\right) + \Pr\left(\mathbf{R} \notin \mathcal{R} | \hat{\mathbf{B}}\right) \\ &\leq \frac{\delta}{3} + 1 - \delta \\ &= 1 - \frac{2\delta}{3},\end{aligned}$$

which contradicts (8). So the theorem holds.

Note that $E[X_{\mathbf{R}}] \geq \alpha U M$ and $E[X_{\mathbf{R}}] \geq \beta(\eta K(M+U) - \eta^2 K^2)$ always hold. So $E[X_{\mathbf{R}}] \leq K^2$ implies $\alpha \leq \frac{K^2}{UM}$ and $\beta \leq \frac{K}{\eta(M+U)-\eta^2 K}$.

Proof of Theorem 4

Following similar argument as the proof of Theorem 2, we can prove that when $\alpha\beta M = \omega(\log M)$ or $\beta^2 K = \omega(\log M)$ all user-clusters and item-clusters are correctly identified with probability at least $1 - \frac{1}{M}$.

Now consider the ratings of item-cluster k_m given by user-cluster k_u and define

$$M_{k_u,k_m,g} = \sum_{u:\lceil uK/U\rceil = k_u} \sum_{m:\lceil mK/M\rceil = k_m} \mathbf{1}_{r_{um}=g}$$

to be the number of g ratings given by users in cluster k_u to to items in cluster k_m. With a slight abuse of notation, let b_{k_u,k_m} to be the true preference. Further let η_{k_u} denote the number of information-rich users in cluster k_u and η_{k_m} denote the information-rich items in cluster k_m. When $g = b_{k_u,k_m}$, we have

$$E\left[M_{k_u,k_m,g}\right] = \left(\left(\eta_{k_u}\frac{M}{K} + \eta_{k_m}\frac{U}{K} - \eta_{k_u}\eta_{k_m}\right)\beta + \left(\frac{U}{K} - \eta_{k_u}\right)\left(\frac{M}{K} - \eta_{k_m}\right)\alpha\right)p;$$

and otherwise,

$$\begin{aligned}E\left[M_{k_u,k_m,g}\right] = \Bigg(&\left(\eta_{k_u}\frac{M}{K} + \eta_{k_m}\frac{U}{K} - \eta_{k_u}\eta_{k_m}\right)\beta \\ &+ \left(\frac{U}{K} - \eta_{k_u}\right)\left(\frac{M}{K} - \eta_{k_m}\right)\alpha\Bigg)\frac{1-p}{G-1}.\end{aligned}$$

Define $\mathcal{G}$ to be the event that a majority voting within an item and user-cluster gives the true preference of item-cluster for all items and user-clusters, i.e.,

$$\mathcal{G} = \{b_{k_u,k_m} = \arg\max_g M_{k_u,k_m,g} \quad \forall\, k_u, k_m\}.$$

Now when $\frac{\alpha U M}{K^2} = \omega(\log M)$ or $\frac{\beta M}{K} = \omega(\log M)$, using the Chernoff bound, it is easy to verify that

$$\Pr(\mathcal{G}) \geq 1 - \frac{1}{M}.$$

Further, $E[X_{\mathbf{R}}] = \omega(K^2 \log M)$ implies that $\frac{\alpha U M}{K^2} = \omega(\log M)$ or $\frac{\beta M}{K} = \omega(\log M)$, so the theorem holds.

References

Amatriain, X., Pujol, J. M., & Oliver, N. (2009). I like it.. i like it not: Evaluating user ratings noise in recommender systems. *User modeling, adaptation, and personalization* (pp. 247–258). Berlin: Springer.

Banerjee, S., Hegde, N., & Massoulié, L. (2012). The price of privacy in untrusted recommendation engines. In *The 50th Annual Allerton Conference on Communication, Control, and Computing*, pp 920–927.

Barman, K., & Dabeer, O. (2012). Analysis of a collaborative filter based on popularity amongst neighbors. *IEEE Transactions on Information Theory*, *58*(12), 7110–7134.

Boyd, S., Cortes, C., Mohri, M., & Radovanovic, A. (2012). Accuracy at the top. In F. Pereira, C. J. C. Burges, L. Bottou & K. Q. Weinberger (Eds.), *Advances in neural information processing systems* (pp. 962–970). Lake Tahoe, Neveda: Curran Associates, Inc.

Cai, J. F., Candès, E. J., & Shen, Z. (2010). A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, *20*(4), 1956–1982.

Candès, E. J., & Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, *9*(6), 717–772.

Candès, E. J., & Tao, T. (2010). The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, *56*(5), 2053–2080.

Gross, D. (2011). Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, *57*(3), 1548–1566.

Jain, P., Netrapalli, P., & Sanghavi, S. (2013). Low-rank matrix completion using alternating minimization. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, pp 665–674.

Keshavan, R. H., Montanari, A., & Oh, S. (2010). Matrix completion from noisy entries. *The Journal of Machine Learning Research*, *99*, 2057–2078.

Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., & Riedl, J. (1997). GroupLens: Applying collaborative filtering to usenet news. ACM *Communications*, *40*(3), 77–87.

Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer*, *42*(8), 30–37.

Mitzenmacher, M., & Upfal, E. (2005). *Probability and computing: Randomized algorithms and probabilistic analysis*. Cambridge: Cambridge University Press.

Recht, B. (2011). A simpler approach to matrix completion. *The Journal of Machine Learning Research*, *12*, 3413–3430.

Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM Conference on Computer Supported Cooperative Work*, New York, NY, pp. 175–186.

Shen, Y., Wen, Z., & Zhang, Y. (2012). Augmented lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optimization Methods and Software*, *29*, 1–25.

Tomozei, D. C., & Massoulié, L. (2014). Distributed user profiling via spectral methods. *Stochastic Systems*, *4*, 1–43. doi:10.1214/11-SSY036.

Xu, J., Wu, R., Zhu, K., Hajek, B., Srikant, R., & Ying, L. (2013). Exact block-constant rating matrix recovery from a few noisy observations. arXiv:1310.0512.