Wisdom of the Better Few: Cold Start Recommendation via Representative based Rating Elicitation

Nathan Liu
Hong Kong University of Science and Technology
nliu@cse.ust.hk

Xiangrui Meng
Stanford University
mengxr@stanford.edu

Chao Liu
Microsoft Research
chaoliu@microsoft.com

Qiang Yang
Hong Kong University of Science and Technology
qyang@cse.ust.hk

ABSTRACT

Recommender systems have to deal with the cold start problem as new users and/or items are always present. Rating elicitation is a common approach for handling cold start. However, there still lacks a principled model for guiding how to select the most useful ratings. In this paper, we propose a principled approach to identify representative users and items using representative-based matrix factorization. Not only do we show that the selected representatives are superior to other competing methods in terms of achieving good balance between coverage and diversity, but we also demonstrate that ratings on the selected representatives are much more useful for making recommendations (about 10% better than competing methods). In addition to illustrating how representatives help solve the cold start problem, we also argue that the problem of finding representatives itself is an important problem that would deserve further investigations, for both its practical values and technical challenges.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval—*Information Filtering*

General Terms

Algorithms, Experimentation

Keywords

Collaborative Filtering, Recommender Systems, Cold Start

1. INTRODUCTION

In recent years, recommender systems have become indispensable tools for coping with information overload in a wide range of Web services such as E-commerce (e.g., Amazon), internet radio (e.g., Last.FM) and online news aggregators (e.g. Digg). Collaborative filtering (CF) is the de facto mechanism for recommendations: it generates recommendations for a user by utilizing the observed ratings of other users whose past ratings are correlated with the target user. Albeit effective, CF falls short when it comes to new users and/or new items. This is a well-known problem, commonly referred to as the *cold start* problem.

Due to its practical importance, cold start has attracted numerous research in recent years [15, 9, 6]. The immediate remedy to cold start is to leverage meta data for estimating the taste of new users and items based on similar users and items [15, 9, 6]. However, as meta data is not always available, as an alterative, the recommender system could explicitly ask the new user's ratings on some *seed* items. Ratings on seed items could help the recommender system calibrate the taste of new users. In the past, various heuristic methods on how to strategically selecting seed items have been proposed [19, 17], but a principled approach is still lacking for determining the valuable seed items and users.

In this paper, we propose a principled approach to determining the seeds by locating the *representatives* in a recommender system. Roughly speaking, representatives are those users whose linear combinations of tastes would accurately approximate other users' (the same for items). Specifically, we leverage the *maximal-volume* concept [8] for low-rank matrix approximations, a recent result from computational mathematics, and devise a representative-based matrix factorization (RBMF) model. This model directly optimizes a criterion that reflects the prediction accuracy for user-item ratings, which leads to superior results than existing methods for representative selection. In addition to providing a highly effective approach to rating elicitation, the RBMF model also provides a highly intuitive and interpretable view of the item or user population of a recommender system, as the factors correspond to actual users or items rather than some latent dimensions as in traditional matrix factorization models.

In summary, the major contributions of the paper are as follows:

- Proposing a principled approach to rating elicitation for new users and item via finding the set of most representative users and items based on observed ratings.
- Designing a novel matrix factorization model both for

RecSys'11, October 23–27, 2011, Chicago, Illinois, USA.

finding most representative users and items in a recommender systems and an efficient representative pursuit algorithm for solving the model.

The remainder of the paper is organized as follows. In section 2, we describe in detail the representative-based matrix factorization model for identifying representative users and items based on past ratings. Then in section 3, we show how to efficiently learn model parameters for new users and new items given the ratings on representative items and from representative users, respectively. In section 4, we report and discuss our experimental results. In section 5, we introduce related work. Finally, in section 6, we draw conclusions and point out several directions for future work.

2. REPRESENTATIVE-BASED MATRIX FACTORIZATIONS

2.1 Problem Definition

Collaborative filtering usually takes a collection of n users $\mathcal{U} = \{u_1, u_2, \dots, u_n\}$ and m items $\mathcal{I} = \{i_1, i_2, \dots, i_m\}$ as inputs. The observed ratings in a recommender system could be represented by a matrix $Y \in \mathbb{R}^{m \times n}$ where each entry y_{ij} corresponds to user j's rating on item i. Most existing matrix factorization models for CF are based on rank-k factorizations of the form $Y \approx UV$, where $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{k \times n}$ are often referred to as item and user latent factor matrices, both of which consist of free parameters that are to be learnt by minimizing certain forms of reconstruction error, for instance, the Frobenius distance $\|Y - UV\|_F$.

In this paper, we are interested in finding rank-k factorizations of the form $Y \approx CX$ or $Y \approx XR$, where X is a *loading* matrix consisting of free parameters while the other matrix $C \in \mathbb{R}^{m \times k}$ or $R \in \mathbb{R}^{k \times n}$ are *component* matrices formed by actual columns or rows of Y respectively. We refer to these two forms of factorizations as *representative-based matrix factorizations (RBMF)* of Y, which can be cast into the following optimization problems:

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|Y - CX\|_F^2 && (1) \\ \text{subject to} \quad & C \in \mathbb{R}^{m \times k} \text{ consists of exactly } k \text{ columns of } Y, \\ & X \in \mathbb{R}^{k \times n}, \end{aligned}$$

and

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}\|Y - XR\|_F^2 && (2) \\ \text{subject to} \quad & R \in \mathbb{R}^{k \times n} \text{ consists of exactly } k \text{ rows of } Y, \\ & X \in \mathbb{R}^{m \times k}. \end{aligned}$$

This kind of factorizations are highly interpretable. For example, given a user representative based factorization CX, any prediction $\hat{y}_{ij}$ is generated by $(CX)_{ij} = \sum_{t=1}^{k} c_{it} x_{tj}$, that is, taking the k representative user's ratings on item i and then computing a weighted average of those ratings based on the coefficients in x_j, the j-th column of X, the values in which can be naturally interpreted as user j's coordinates under the bases of the k representative users.

Note that any algorithm for user representative-based factorization $Y \approx CX$ can be easily turned into an algorithm for item representative-based factorization $Y \approx XR$ by taking Y^T as input. So in the remaining subsections, we would focus on the problem of finding user representative-based factorization: $Y \approx CX$.

The representative-based matrix factorizations could be naturally divided into two stages:

1. Finding representatives: forming the component matrix C by selecting k columns of Y,
2. Computing the loading matrix X given C to minimize the reconstruction error.

In the following two subsections, we describe our algorithms for performing these two steps respectively.

2.2 Finding Representatives

2.2.1 Overview

The representative set should consists of active users who well represent the whole population but with little taste overlap. We design the Representative Pursuit algorithm following this guideline. It consists of the following two major steps:

1. Dimension reduction: reducing the dimensionality of the column space of Y from m to k, while trying to preserve the relationships between users maximally,
2. Basis selection: selecting k representative users to form a well-conditioned set of bases in the reduced space.

These two steps define a framework for representative selection. For each step, we have choices in choosing suitable algorithms. Representative Pursuit (Algorithm 1) employs the rank-k SVD approximation for dimension reduction and the `maxvol` algorithm by Goreinov et al [7] for basis selection to implement the two steps.

Algorithm 1 Representative Pursuit

```
1: Compute the rank-$k$ SVD approximation to $Y$,
       $$Y_k = U \Sigma V^T, \qquad (3)$$
   where $U \in \mathbb{R}^{m \times k}$, $\Sigma \in \mathbb{R}^{k \times k}$ is a diagonal matrix with
   $Y$'s $k$ largest singular values as diagonals and $V \in \mathbb{R}^{n \times k}$.
2: Find a $k \times k$ submatrix $C$ of $V$ such that all the entries of
   $VC^{-1}$ are not greater than 1 in absolute value (Algorithm 2).
   Select the corresponding users as representative users.
```

2.2.2 Dimension Reduction

In this step, we want to map the column space of Y to a k-dimensional space while preserving the relationships among users as much as possible. In Algorithm 1, we choose the factor matrix V from the rank-k SVD approximation to Y as the dimension reduction result. The rank-k SVD approximation gives the best rank-k approximation to Y in 2-norm and Frobenius norm, where the missing values in Y are treated as zeros. In practice, other rank-k approximations may give better handling of missing values, e.g., Funk's stochastic gradient descent method [12]. In this work, we choose rank-k SVD for simplicity, which already generates very good results and saves us from tuning parameters. It is worth noting that other dimension reduction algorithms, such as multidimensional scaling and random projections, may also be incorporated here.

Given a rank-k approximation $Y_k = UV^T$, the factor matrices are not unique. For example, we have

$$Y_k = UV^T = \left(UG^{-T}\right)(VG)^T = U_1 V_1^T$$

hold for any non-singular matrix $G \in \mathbb{R}^{k\times k}$. It raises the question that how G affects the basis selection. We will see in the following subsection that, using the `maxvol` algorithm, the basis selection is invariant to G. This unique property frees us from defining a proper axis for the latent space.

2.2.3 Basis Selection

In this section, we consider the problem of selecting a well-conditioned set of bases in the latent space by exploring the maximal-volume concept [5]. In detail, it suggests selecting columns to maximize the parallelepiped spanned by them to form low-rank approximations. To maximize the volume, the columns selected should be large in magnitudes which indicates importance, and evenly distributed to prevent linear dependency, both of which fit our purpose well. To the best of our knowledge, the best approximation ratio of column-based matrix factorizations was achieved using random volume sampling by Deshpande and Vempala [5]. This result provides theoretical foundations for exploring the maximal-volume concept for representative selection. However, the problem of finding the maximal-volume k columns is provably NP-hard [4] and, moreover, we should not apply any approximation algorithm to a rating matrix directly due to its high portion of missing values. A more robust approach is to first reduce the dimensionality of the column space of Y and then search for the maximal-volume submatrix in the latent space for representative/basis selection.

Since we already reduce the dimensionality of Y's column space to k. Finding k columns from V^T maximizing the parallelepiped spanned by them is equivalent to finding a $k \times k$ submatrix of V having the maximal determinant in absolute value because

$$\mathrm{Vol}(S) = |\det(S)| \tag{4}$$

for any square matrix S. Goreinov et al [7] presented a practical algorithm, named `maxvol`, for searching the maximal-volume $k \times k$ submatrix of a $n \times k$ matrix V based on a dominant property of the maximal-volume submatrix. In the remainder of this subsection, we first show the dominant property and demonstrate how it leads to the `maxvol` algorithm. Then we prove the invariant property and show how we incorporate activeness scores into the framework.

Without loss of generality, let's assume that the submatrix formed by the first k rows of V has the maximal volume. Then we must have

$$|VV(1:k,:)^{-1}|_\infty \le 1, \tag{5}$$

where MATLAB notations are adopted and $|\cdot|_\infty$ is the elementwise ∞-norm. We prove (5) by contradiction. Let

$$VV(1:k,:)^{-1} = \begin{pmatrix} I_k \\ Z \end{pmatrix}.$$

If $|Z|_\infty > 1$, again without loss of generality, let us assume that $|z_{11}| > 1$. Then we swap the first row and the $(k+1)$-th row of V, and denote the resulting matrix by $\bar V$. We have

$$\bar V(1:k,0)V(1:k,:)^{-1} = \begin{pmatrix} z_{11} & z_1^T \\ 0 & I_{k-1} \end{pmatrix}.$$

Therefore,

$$\begin{aligned} & |\det(\bar V(1:k,:))| \\ = \; & |\det(\bar V(1:k,:)V(1:k,:)^{-1}V(1:k,:))| \\ = \; & |\det(\bar V(1:k,:)V(1:k,:)^{-1})||\det(V(1:k,:))| \\ = \; & |z_{11}||\det(V(1:k,:))| \\ > \; & |\det(V(1:k,:))|, \end{aligned}$$

which contradicts the maximal-volume assumption on $V(1:k,:)$. Hence (5) holds, which says the maximal-volume submatrix is dominant in this sense. However, the reverse statement is not always true: If a $k \times k$ submatrix $V(I,:)$ with $|I| = k$ is dominant, i.e. $V(I,:)$ satisfies

$$|VV(I,:)^{-1}| \le 1,$$

$V(I,:)$ does not necessarily have the maximal volume among all the $k \times k$ submatrices. Although $V(I,:)$ may not be the maximal-volume one, Goreinov et al [7] showed that at the worst case we have

$$|\det(V(I,:))| \ge k^{-k/2}|\det(V(I^*,:))|,$$

where $V(I^*,:)$ is the maximal-volume $k \times k$ submatrix.

More importantly, the insight from the proof of (5) leads to a local search algorithm, `maxvol`, to find a local maximal-volume submatrix, which works well in practice.

Algorithm 2 `maxvol` [7]

```
1: Compute the LU factorization of $V$ with row pivoting,
       $V(p,:) = L_V U_V$.
2: Compute
       $B = VV(p(1:k),:)^{-1}$.
3: Let $(i,j) = \arg\max_{s,t} |b_{st}|$.
   If $|b_{ij}| > 1$, swap $p(i)$ and $p(j)$ and go to 2.
4: Output $p(1:k)$.
```

The LU factorization with partial pivoting provides a reasonable initial guess. The volume of $V(p(1:k),:)$ is strictly increasing after swapping $p(i)$ and $p(j)$ in step 3. Hence the algorithm will terminate in finite number of steps, less than 100 in practice. A careful implementation only takes one LU factorization with partial pivoting to initialize, and a rank one update for each iteration.

2.3 Solving Loading Matrix X

In this section, we consider the problem of computing X given C to minimize the objective function of (1), which is a least squares problem. The optimal solution is given by

$$X^* = (C^TC)^{-1}C^TY, \tag{6}$$

assuming C has full rank. To avoid breakdown due to rank deficiency, which is rare for the problem we are considering, we add a small positive λ to the diagonals of C^TC in (6) to increase numerical stability,

$$X^* = (C^TC + \lambda I)^{-1}C^TY, \tag{7}$$

where I is $k \times k$ identity matrix. Note that (7) is the optimal solution to the following regularized least squares problem,

$$\mathrm{minimize}_{X\in\mathbb{R}^{k\times n}} \; \frac{1}{2}\|Y - CX\|_F^2 + \frac{\lambda}{2}\|X\|_F^2.$$

3. COLD START RECOMMENDATION VIA REPRESENTATIVES

The representative-based matrix factorization (RBMF) model allows us to identify the set of most representative users and items based on past ratings and make recommendation to existing users. It also provides us an intuitive rating elicitation strategy for new users and items. In this section, we discuss how an existing RBMF model can be efficiently updated to cope with new users and new items.

In most real world systems, new items and new users would be added to the system constantly. A recommender system should be able to adjust its model rapidly in order to be able to make recommendations regarding new users and new items as soon as possible. This require techniques for learning the parameters associated with new users and new items based on an increment of new data without the need to retrain the whole model entirely. This type of techniques have also been known as *folding in.* Using the RBMF model, folding in is effortless. In particular, we only need to obtain ratings from the k representative users for a new item in order to recommend it to other users. Similarly, we only need to ask a new user to rate k representative items to recommend other items to him.

Let's first consider the new item case, given a user representative based factorization $Y \approx CX$. Denote the index set of selected user representatives by J such that $C = Y(:, J)$. And let the new rating matrix be $\bar{Y}$, which contains $m+1$ rows and n columns. Our goal is to predict the newly inserted row $\bar{Y}(m+1, :)$ in order to effectively recommend the new item to users. Via representative users, it is s-traightforward to estimate $\bar{Y}(m+1, :)$ by eliciting ratings on the new item from representative users. To see this, let's consider how the newly inserted item would effect the user representative-based factorization of Y. With only one additional item inserted into the system, it is reasonable to assume that the representative set J remains the same as well as the relationships between users and representatives depicted by X. Hence a suboptimal user representative-based matrix factorization of the new rating matrix $\bar{Y}$ is given by

$$\bar{Y} \approx \bar{Y}(:, J)X = \begin{pmatrix} Y(:, J) \\ \bar{Y}(m+1, J) \end{pmatrix} X. \tag{8}$$

The only unknowns in the factorization above are the entries of $\bar{Y}(m+1, J)$, i.e., the ratings from representative users on the new item. Therefore, if $\bar{Y}(m+1, J)$ is available, or partially available, the newly inserted row $\bar{Y}(m+1, :)$ can be well approximated by

$$\bar{Y}(m+1, :) \approx \bar{Y}(m+1, J)X. \tag{9}$$

By similar arguments, we can easily fold in a new user into an item representative-based factorization model by asking the new user to rate representative items. The estimate of the newly inserted column of the rating matrix is given by

$$\bar{Y}(:, n+1) \approx X\bar{Y}(I, n+1), \tag{10}$$

where I is the index set of representative items.

Recall that we use activeness scores to promote active users or popular items to be selected as representatives in order to increase the success rate of obtaining ratings from representative users on new items, or ratings from new users on representative items. In practice, we have many other ways to increase the success rate, e.g., offering free trials of new items to representative users, or offering bonus points to new users for rating representative items.

We note that new ratings can also be folded into a general matrix factorization (MF) model, e.g., rank-k SVD, at the cost of solving a least squares problem. In the experiment section, we will try both MF model and RBMF model for folding in new items and new users and show that RBMF is not only costless compared to MF for folding in, but also generates comparable recommendations.

4. EXPERIMENTS

4.1 Data Sets

We used three different data sets for both movie and music recommendation in our experiments. The first data set is a randomly sampled subset of the Netflix data set[1] containing about 20 million ratings by 100,000 users on over 17,000 movies. Our second data set is the Movielens dataset[2] containing over 10 million ratings by over 60,000 users on about 10,000 movies. The third data set is obtained from the online radio service Last.fm[3] containing over 2 million records of play counts on around 5,000 artists by around 50,000 users.

4.2 Baseline Methods

In this section, we describe several simple strategies for finding a set of representative users and items, whose performances are compared with our representative-based matrix factorization (RBMF) model.

4.2.1 Random Strategy

A naive method to choose the representatives is simply to randomly select a subset of users or items. This strategy corresponds to the assumption that users or items are indifferent in terms of their predictive power on other users' and items' ratings and therefore there would be no gain in strategically choosing which users and items to elicit ratings from.

4.2.2 Most Ratings Strategy

Another simple method to select the representative is to choose the k users or k items which had the most observed ratings. This strategy is also easy to calculate. However, popularity in many cases are not equivalent to informativeness. For example, the most popular movies tend to be widely liked by almost any user and a rating on such movies would provide little information regarding the new user's unique preferences. Similarly, a very active user may be someone who frequently watches randomly selected movies and may not serve as a good user prototype.

4.2.3 K-Medoids Strategy

The previous two methods do not consider the correlation between the selected representatives and could potentially choose multiple highly similar users or items. To avoid such redundancy problem, we also consider another more complicated strategy based on k-medoids clustering [11]. k-medoids tries to group data objects into k clusters. Each cluster is represented by an actual data object, i.e. the representative. The other instances are clustered with the

[1] http://www.netflixprize.com/

[2] http://www.grouplens.org/node/73

[3] http://mtg.upf.edu/node/1671

representative to which it is most similar to. In our experiments, we used the CLARA algorithm [11] to generate the k-medoids clustering. It tries to maximize the sum of similarities between each object and its corresponding representative and uses an iterative process to repeatedly assign objects to clusters and choose new cluster representatives to improve the clustering.

4.3 Analysis of the Selected Representatives

In this set of experiments, we tried to evaluate the quality of a selected representative user or item sets with respect to the population based on the following quantitative measures:

- **Coverage**: the proportion of users (or items) which have rated the set of representative items (or have been rated by the representative users). This metric allows us to measure how widely are the user or item population being covered by the representative set as a whole.

- **Diversity**:the proportion of users (or items) which have rated less than 10% of the representative items (or have been rated by less than 10% of the representative users). This metric reflects how much unique information are carried by each representative.

The results of analysis are summarized in Table 1 and Table 2. We can see that the Most strategy tends to have the highest coverage but very low diversity. In contrast, the representatives chosen by RBMF tend to have much higher diversity while attaining similar coverage. This shows that RBMF based representative selection is highly effective for achieving a good balance between coverage and diversity.

Table 1: Analysis of User Representatives Selected by Different Algorithms

	Coverage	Diversity
most	0.9450	0.2273
kmedoids	0.9210	0.2613
rbmf	0.8657	0.4611

(a) Netflix

	Coverage	Diversity
most	0.9561	0.1072
kmedoids	0.9544	0.1298
rbmf	0.8918	0.3552

(b) Movielens

	Coverage	Diversity
most	0.3790	0.7474
kmedoids	0.2768	0.7525
rbmf	0.2868	0.7759

(c) Lastfm

4.4 Experiments on Rating Elicitation

In the following experiments, we try to compare the effectiveness of different representative selection strategies for the task of rating elicitation under both user and item cold start. To simulate *user cold start* setting, we split the users into a disjoint training set and a test set. The representative items are selected only based on the ratings by the users in the training set. Then for the users in the test set, we make predictions only based on their ratings on the representative items. The effectiveness of the representative items are then evaluated based on how well they help identify which of nonrepresentative items a users also likes. Similarly, for the *item cold start* setting, we split the items into a training and a test set. The representative users are determined based on the training set only. Then for the test set of items, we try to predict which other users would like the test items based only on the representative users' ratings.

Table 2: Analysis of Item Representatives Selected by Different Algorithms

	Coverage	Diversity
most	0.8784	0.1705
kmedoids	0.8737	0.2203
rbmf	0.8533	0.3164

(a) Netflix

	Coverage	Diversity
most	0.9271	0.1357
kmedoids	0.9471	0.1491
rbmf	0.9183	0.3270

(b) Movielens

	Coverage	Diversity
most	0.9399	0.2815
kmedoids	0.9435	0.3821
rbmf	0.9603	0.3135

(c) Last.FM

4.4.1 Evaluation Protocol

We evaluate different algorithm's ability to distinguish good recommendations from bad ones. Following most recently proposed evaluation schemes [12, 20], we consider both observed and missing ratings in our evaluation to account for the Not Missing At Random(NMAR) effect [13]. For Netflix and Movielens data sets, we consider an item as a good recommendation if the corresponding rating is higher than 4. Otherwise, we consider it as a bad recommendation if the rating is either lower than 4 or unknown. Similarly, for the Last.FM data set, we consider an artist to be a good recommendation to a user if he/she had played the artist more than 20 times and both unplayed artists or less played artists as bad recommendations.

For both settings, different methods are used either to rank items for each user in the user cold start setting or to rank users for each item in the item cold start setting. Since each result in a ranking is either good or bad, we adopted several binary relevance based information retrieval performance metrics including Precision at K (Pre@K), Mean Average Precision (MAP) and Area under Curve (AUC), which are described in the following.

Precision at K is a widely used performance metric which is defined as the fraction of good recommendations among the top k recommendations.

$$Pre@K = \frac{N_K^+}{N_K^+ + N_K^-} \tag{11}$$

where N_K^+ and N_K^- are the number of good and bad recommendations among the top k results in a ranking. In our experiment, the $Pre@K$ for the rankings for different cold start items and users are first computed independently and the average over all users and items in the test set are then taken as the final performance.

Rather than considering the precision at only the K-th position, the metric average precision (AP) considers all rank positions with good results and is defined as the average of the precisions at these positions:

$$AP = \frac{\sum_{i=1}^{N} Pre@i \times \delta^{+}(i)}{N^{+}} \qquad (12)$$

where $Pre@i$ is the precision at the i-th position and $\delta^{+}(i)$ is a binary indicator function that returns 1 if the result at the i-th position is good and 0 if otherwise. The mean average precision (MAP) is simply the mean of the average precisions over all users and items in the test set.

Area under Curve(AUC) is defined the area under the receiver operating characteristic (ROC) curve, which is mathematically equivalent to the proportion of correctly ordered pairs in the ranking. A pair of result is mis-ordered if the result at the higher rank position is a bad result while the result at the lower rank position is a good result:

$$AUC = 1 - \frac{N^{\mp}}{N^{+} \cdot N^{-}}$$

where $N^{\mp}$ denotes the number of mis-ordered result pairs. Similar to Pre@K, we compute the AUC for each user and item in the test and then take the average of these AUC scores as the overall performance.

4.4.2 Results

We empirically compared different representative selection strategies in terms of the effectiveness of the elicited ratings for making recommendations on new users and new items. For evaluating item representatives, we split the users into a training set and test set and then apply different methods to determine a set of representative items based on the complete rating matrix of the training users. Similarly, for evaluating user representatives, we construct a training and a test set of items and use different methods to identify the most representative users based on the rating matrix of the training items. In both cases, we use 80% of the users or items for training and the remaining 20% for test.

Once the representative users/items are determined based on the training data, we could then simulate rating elicitation on new users/items by extracting the set of ratings from the representative users and items from the test set. We then apply the folding in techniques described in section 3 to learn the parameters for the simulated new users or new items based on their ratings on the representative items or from representative users. In our experiments, we first built both a representative-based matrix factorization (RBMF) model and a traditional matrix factorization model (MF) [12] from the training data. We then try to fold in the new users and items based on the ratings from the representatives for both models following the procedure in section 3. Finally, we use the model after folding in to make recommendations. Comparing the effectiveness of different representative selection strategies would allow us to study whether they perform consistently for different models.

In our experiments, we also tried to vary the number of representative users from 20 to 100 and the number of representative items from 5 to 20 to simulate increasing rating elicitation cost budget. This allows us to compare the effectiveness of different strategies under different budgets.

The complete results are shown in Table 3 and Table 4 respectively, which reveals the following findings:

1. The naive random strategy generally performed significantly worse than the other strategies. This confirmed our belief that it is necessary to carefully choose which users and items to elicit ratings under cold start.
2. The RBMF based representative selection outperformed the other methods by a significant margin under almost all settings except for the user representative task on the Last.FM data set, in which it attained similar performance as k-medoids.
3. We can see that the relative effectiveness of different methods were quite consistent irrespective of whether RBMF or MF was used for folding in. Moreover, the relative performances are also rather consistent across different values of k, the number of representatives.

5. RELATED WORKS

Recently, Amatriain et. al. [1] investigated how to accurately predict a users' movie preferences only based on only a small set of experts' ratings, where the experts are movie critics who regularly publish online reviews. While this model relies on a set of predefined experts, the representative users in our model are automatically chosen from the existing user population of the system. Moreover, as the representative users or items are within the target system itself rather than some independent third parties.

While there are very few works to study the problem of finding representative users/items in recommender system, the social network analysis community have long recognized that a small number of influential users could often be instrumental in affecting other users' adoption of products. This had let to various *viral marketing* strategies [2, 18, 16], which first targets the influencers in a social network and then relies social influence to escalate the impact. Moreover, the reaction to new products by influencers can also serve as strong evidence for predicting the community's adoption rates [3]. These existing methods heavily relied on the underlying network connections between users in order to discover influencers and design information propagation models. In contrast, our model only requires user feedback and is applicable to recommender systems without explicit social networks.

Matrix factorizations using original columns or rows (CX), or using both (CUR), have received increasing interests in the fields of scientific computing and theoretical computer science. Deshpande and Vempala [5] proved that, in theory, given a matrix $A \in \mathbb{R}^{m \times n}$ and an integer $k \ll m, n$, there exists a submatrix C consisting of k columns of A such that $\|A - C(C^{+}A)\|_F \leq \sqrt{k+1}\|A - A_k\|_F$, where A_k denotes the best rank-k approximation to the matrix A in Frobenius norm. Practical algorithms generally fall into two categories: 1) using Rank Revealing QR (RRQR) factorizations [14], 2) exploring the maximal-volume concept [8].

In the data mining community, there exists very few works on algorithms and applications of CX or CUR type factorizations and the existing works have mostly emphasized the sparsity and interpretability of these models. Sun et al applied CX factorization on large graphs and showed that the sparse nature of the matrix C could lead much more compact yet highly accurate decomposition of large sparse matrices [21]. Hyvonen et. al. [10] considered non-negative CX and CUR problems. They proposed two algorithms, both of which solve non-negative least squares problems at each iteration. We don't consider the non-negative option since it is easy to interpret negative factors for matrix factorizations in recommender systems.

Table 3: Comparing Different Strategies for Selecting User Representatives for Item Cold Start

		Netflix			Movielens			Last.fm		
	K	P@10	MAP	AUC	P@10	MAP	AUC	P@10	MAP	AUC
random	20	0.080	0.032	0.523	0.118	0.049	0.553	0.036	0.016	0.528
most		0.272	0.053	0.693	0.405	0.118	0.814	0.032	0.015	0.533
kmedoids		0.129	0.057	0.701	0.138	0.051	0.554	0.053	**0.025**	0.539
rbmf		**0.330**	**0.102**	**0.714**	**0.482**	**0.144**	**0.833**	**0.057**	**0.025**	**0.540**
random	40	0.123	0.043	0.549	0.244	0.053	0.567	0.051	0.021	0.548
most		0.281	0.089	**0.746**	0.415	0.124	0.817	0.049	0.020	0.555
kmedoids		0.124	0.049	0.544	0.179	0.063	0.576	**0.078**	**0.034**	**0.562**
rbmf		**0.356**	**0.111**	0.728	**0.515**	**0.162**	**0.825**	0.060	0.026	0.557
random	100	0.164	0.057	0.576	0.239	0.076	0.619	0.084	0.032	0.582
most		0.296	0.094	0.747	0.014	0.044	0.736	0.079	0.029	0.593
kmedoids		0.176	0.067	0.575	0.217	0.066	0.651	**0.136**	**0.051**	**0.601**
rbmf		**0.374**	**0.122**	**0.765**	**0.536**	**0.173**	**0.839**	0.122	0.046	0.596

(a) Results of Folding in the RBMF model

		Netflix			Movielens			Last.fm		
	K	P@10	MAP	AUC	P@10	MAP	AUC	P@10	MAP	AUC
random	20	0.200	0.056	0.637	0.224	0.059	0.65	0.024	0.015	0.528
most		0.083	0.049	0.737	0.125	0.075	0.798	0.028	0.016	0.530
kmedoids		0.176	0.053	0.631	0.261	0.072	0.664	0.042	**0.023**	**0.537**
rbmf		**0.308**	**0.082**	**0.726**	**0.444**	**0.120**	**0.805**	**0.045**	0.022	**0.537**
random	40	0.211	0.057	0.628	0.236	0.064	0.664	0.040	0.020	0.544
most		0.034	0.047	0.729	0.075	0.071	0.803	0.038	0.019	0.547
kmedoids		0.197	0.060	0.641	0.275	0.077	0.672	0.052	0.026	0.553
rbmf		**0.299**	**0.080**	**0.734**	**0.425**	**0.117**	**0.813**	**0.055**	**0.027**	**0.555**
random	100	0.224	0.066	0.654	0.347	0.099	0.713	0.055	0.027	0.581
most		0.014	0.044	0.736	0.024	0.064	0.811	0.051	0.025	0.584
kmedoids		0.217	0.066	0.651	0.316	0.094	0.707	**0.067**	**0.032**	0.590
rbmf		**0.255**	**0.079**	**0.744**	**0.428**	**0.122**	**0.836**	0.065	**0.032**	**0.592**

(b) Results of Folding in the MF model

Table 4: Comparing Different Strategies for Selecting Item Representatives for User Cold Start

		Netflix			Movielens			Last.fm		
	K	P@10	MAP	AUC	P@10	MAP	AUC	P@10	MAP	AUC
random	5	0.017	0.017	0.516	0.012	0.016	0.537	0.012	0.011	0.506
most		0.459	**0.227**	**0.819**	0.432	0.215	0.821	0.193	0.091	**0.696**
kmedoids		0.172	0.089	0.594	0.191	0.092	0.620	0.198	0.095	0.638
rbmf		**0.465**	0.226	0.815	**0.501**	**0.262**	**0.831**	**0.211**	**0.099**	0.692
random	10	0.158	0.085	0.592	0.037	0.026	0.545	0.023	0.016	0.506
most		0.488	0.236	**0.833**	0.500	0.254	0.851	0.224	0.105	0.749
kmedoids		0.167	0.087	0.589	0.256	0.120	0.660	0.235	0.112	0.739
rbmf		**0.519**	**0.248**	0.820	**0.559**	**0.299**	**0.862**	**0.252**	**0.117**	**0.762**
random	20	0.215	0.110	0.632	0.134	0.070	0.591	0.096	0.048	0.575
most		0.503	0.239	**0.850**	0.530	0.277	0.874	0.263	0.124	0.788
kmedoids		0.252	0.126	0.644	0.380	0.199	0.724	0.287	0.139	0.781
rbmf		**0.563**	**0.273**	0.848	**0.637**	**0.343**	**0.888**	**0.322**	**0.152**	**0.814**

(a) Results of Folding in the RBMF Model

		Netflix			Movielens			Last.fm		
	K	P@10	MAP	AUC	P@10	MAP	AUC	P@10	MAP	AUC
random	5	0.278	0.112	0.746	0.261	0.111	0.727	0.174	0.074	0.725
most		0.369	0.166	0.748	0.400	0.198	0.773	0.191	0.090	0.750
kmedoids		0.300	0.122	0.746	0.309	0.133	0.732	0.213	0.099	0.760
rbmf		**0.459**	**0.200**	**0.783**	**0.495**	**0.251**	**0.804**	**0.219**	**0.101**	**0.763**
random	10	0.278	0.114	0.748	0.291	0.128	0.731	0.187	0.077	0.725
most		0.316	0.155	0.748	0.351	0.202	0.789	0.179	0.089	0.764
kmedoids		0.304	0.126	0.751	0.304	0.130	0.736	**0.243**	0.101	0.764
rbmf		**0.512**	**0.228**	**0.807**	**0.520**	**0.266**	**0.829**	0.234	**0.111**	**0.773**
random	20	0.305	0.123	0.754	0.354	0.143	0.742	0.201	0.085	0.736
most		0.254	0.153	0.766	0.288	0.197	0.805	0.167	0.090	0.789
kmedoids		0.340	0.133	0.753	0.424	0.206	0.778	**0.279**	0.123	0.771
rbmf		**0.520**	**0.246**	**0.831**	**0.586**	**0.309**	**0.854**	0.261	**0.125**	**0.811**

(b) Results of Folding in the MF model

6. CONCLUSION AND FUTURE WORK

In this paper, we proposed a principled approach to finding the most representative users and items in a recommender system via the representative-based matrix factorization (RBMF) model. We show that the representative users and items can be used as targets from which to elicit ratings during cold start. We conducted extensive experiments on both movie and music rating data sets and demonstrated that the representative-based rating elicitation significantly outperforms existing heuristic strategies for both user and item cold start. We also designed several quantitative measures including coverage and diversity for characterizing the representative set and showed that the representatives chosen by the RBMF model is able to attain both high coverage and high diversity.

The problem of finding representative users and items in a recommender systems present many new and interesting challenges for data mining. One potential to improve our current model for representative selection is to incorporate user/item meta-data in addition to ratings. Another issue is to design refined objective functions based on the coverage and diversity metrics we identified. Moreover, we will also try to explore the use of representative finding in other novel applications (e.g., behavioral targeting) to further validate its usefulness.

7. ACKNOWLEDGEMENT

We would like to thank the support from grant Hong Kong RGC project 621010.

8. REFERENCES

[1] X. Amatriain, N. Lathia, J. M. Pujol, H. Kwak, and N. Oliver. The wisdom of the few: a collaborative filtering approach based on expert opinions from the web. In *SIGIR '09: Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval.*

[2] H. Bao and E. Y. Chang. Adheat: an influence-based diffusion model for propagating hints to match ads. In *WWW '10: Proceedings of the 19th international conference on World wide web*, 2010.

[3] R. Bhatt, V. Chaoji, and P. Rajesh. Predicting product adoption in large-scale social networks. In *CIKM '10: Proceedings of the 19th international conference on World wide web*, 2010.

[4] A. Civril and M. Magdon-Ismail. Finding maximum Volume sub-matrices of a matrix. *RPI Comp Sci Dept TR*, pages 07–08, 2007.

[5] A. Deshpande and S. Vempala. Adaptive sampling and fast low-rank matrix approximation. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 292–303, 2006.

[6] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme. Learning attribute-to-feature mapping for cold-start recommendations. In *ICDM'10: Proceedings of the 10th IEEE international conference on data mining*, 2010.

[7] S. Goreinov, L. Oaeledets, D. Savostyanov, E. Tyrtyshnikov, and N. Zamarashkin. How to find a good submatrix. *Matrix Methods: Theory, Algorithms and Applications*, page 247, 2010.

[8] S. Goreinov and E. Tyrtyshnikov. The maximal-volume concept in approximation by low-rank matrices. In *Structured matrices in mathematics, computer science, and engineering: proceedings of an AMS-IMS-SIAM joint summer research conference, University of Colorado, Boulder, June 27-July 1, 1999*, volume 280, page 47. Amer Mathematical Society, 2001.

[9] A. Gunawardana and C. Meek. Tied boltzmann machines for cold start recommendations. In *RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems*, 2008.

[10] S. Hyvonen, P. Miettinen, and E. Terzi. Interpretable nonnegative matrix decompositions. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2008.

[11] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis.* Wiley-Interscience, 2005.

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

[13] B. M. Marlin and R. S. Zemel. Collaborative prediction and ranking with non-random missing data. In *RecSys '09: Proceedings of the third ACM conference on Recommender systems*, 2009.

[14] C. Pan. On the existence and computation of rank-revealing LU factorizations* 1. *Linear Algebra and its Applications*, 316(1-3):199–222, 2000.

[15] S.-T. Park, D. Pennock, O. Madani, N. Good, and D. DeCoste. Naïve filterbots for robust cold-start recommendations. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2006.

[16] F. J. Provost, B. Dalessandro, R. Hook, X. Zhang, and A. Murray. Audience selection for on-line brand advertising: privacy-friendly social network targeting. In *SIGKDD 2009.*

[17] A. M. Rashid, I. Albert, D. Cosley, S. K. Lam, S. M. McNee, J. A. Konstan, and J. Riedl. Getting to know you: learning new user preferences in recommender systems. In *IUI '02: Proceedings of the 7th international conference on Intelligent user interfaces.*

[18] M. Richardson and P. Domingos. Mining knowledge-sharing sites for viral marketing. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2002.

[19] S. Shearin and H. Lieberman. Intelligent profiling by example. In *IUI '01: Proceedings of the 6th international conference on Intelligent user interfaces.*

[20] H. Steck. Training and testing of recommender systems on data missing not at random. In *KDD '10: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*

[21] J. Sun, Y. Xie, H. Zhang, and C. Faloutsos. Less is more: Compact matrix decomposition for large sparse graphs. In *Proceedings of the 2007 SIAM International Conference on Data Mining (SDM)*, 2007.