Local Low-Rank Matrix Approximation with Preference Selection of Anchor Points

Menghao Zhang
Beijing University of Posts and Telecommunications
Beijing,China
Jack@bupt.edu.cn

Binbin Hu
Beijing University of Posts and Telecommunications
Beijing,China
hubinbin@bupt.edu.cn

Chuan Shi
Beijing University of Posts and Telecommunications
Beijing,China
shichuan@bupt.edu.cn

Bai Wang
Beijing University of Posts and Telecommunications
Beijing,China
wangbai@bupt.edu.cn

ABSTRACT

Matrix factorization is widely used in personalized recommender systems, text mining, and computer vision. A general assumption to construct matrix approximation is that the original matrix is of global low rank, while Joonseok Lee et al. proposed that many real matrices may be not globally low rank, and thus a locally low-rank matrix approximation method has been proposed [11]. However, this kind of matrix approximation method still leaves some important issues unsolved, for example, the randomly selecting anchor nodes. In this paper, we study the problem of the selection of anchor nodes to enhance locally low-rank matrix approximation. We propose a new model for local low-rank matrix approximation which selects anchor-points using a heuristic method. Our experiments indicate that the proposed method outperforms many state-of-the-art recommendation methods. Moreover, the proposed method can significantly improve algorithm efficiency, and it is easy to parallelize. These traits make it potential for large scale real-world recommender systems.

1. INTRODUCTION

Recommender systems have become an important research area since the appearance of the first paper on collaborative filtering in the mid-1990s [7, 16, 18]. There has been much work done both in the industry and academia over the last decade, and they focus on developing new approaches to recommender systems. Personalized recommendations based on user interest and purchasing behavior characteristics, provide the user information and goods they are interested in. With the continuous expansion of the e-commerce, and the rapid growth in the number and variety of goods, customers need to spend a lot of time to find what they want. The process of visiting a large number of irrelevant information and products will undoubtedly continue to lose customers and it will be certainly drowned in information overload problem. To solve these problems, personalized recommender systems came into being. Recommender systems have attracted much attention from multiple disciplines, and many techniques have been proposed to build recommender systems.

Obviously, the recommendation methods are the most critical part of recommender systems. Four fundamental approaches to recommendation can be mentioned: demographic filtering, collaborative and content-based recommendation, and simplified statistical approaches [8]. Collaborative filtering recommendation technology is one of the most successful technology of recommender systems. It is typically based on item ratings explicitly delivered by users. The method recommends products, which have been evaluated positively by another similar user or by a set of such users, whose ratings have the strongest correlation with the current user [6].

One of the most successful collaborative filtering method is Probabilistic Matrix Factorization (PMF) [17]. The goal of the matrix factorization is to decompose user - item rating matrix into user factor matrix and item factor matrix. This matrix factorization model scales linearly with the number of observations and, more importantly, performs well on the large, sparse, and very imbalanced datasets.

One basic assumption on the PMF is that the original matrix is a global low-rank matrix, which suggests that it is reasonable to assume that the matrix has low-rank. However, Joonseok Lee proposed Local Low-Rank Matrix Approximation (LLORMA) [11] with an assumption that the matrix is of locally low-rank rather than globally low-rank. The method randomly selects some anchor-points. Then, it estimates local low-rank matrix approximation for each neighborhood of the anchor-point. Finally, the local matrix models are linearly combined to predict new user-item rating. LLORMA regards the observed matrix as superposition of multiple matrices.

In LLORMA, a critical step is to randomly select anchor-points for each local matrix. This anchor-point selection method may lead to a bad performance and low efficien-

WWW 2017 Companion, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4914-7/17/04.
http://dx.doi.org/10.1145/3041021.3051148

cy because the anchor-points may be in the sparse region or in the overlapping area. In this paper, we study how to select anchor-points in local low-rank matrix factorization model, and propose a preferable anchor-point selecting method with Clustering for LLORMA (called CLLORMA). The method improves both the effectiveness and efficiency of local low-rank matrix factorization. The basic idea is to generate candidate anchor-points by a clustering method, and then select respective anchor-points based on area density and anchor-points distance criteria. First of all, we factorize the observed rating matrix to obtain the features of users (row) and items (column). Next, we cluster user and item factor matrices to obtain the candidate anchor-points and the distance simultaneously. Then we heuristically select the anchor-points from these candidate anchor-points. Finally, we combine all the local matrices established around the anchor-points to predict the original matrix. Our experiments show that the heuristic point selection method based on clustersąŕ density and distances between anchor-points is more accurate and efficient than randomly point selecting method in the local low-rank matrix factorization model.

The remainder of this paper is organized as follows. Section 2 introduces the related work, then the proposed clustering local low-rank matrix approximation (CLLORMA) model is detailed in Section 3. Experiments and analysis are shown in Section 4. Last, we conclude the paper in Section 5.

2. RELATED WORK

Recommender system is a solution for the information overload problem, helps users to find objects of interest through utilizing the user-item interaction information or users' and items' content information. Recommender systems have attracted so much attention over the years and many techniques have been proposed to provide recommendation service.

According to the utilized information for recommendation, we can roughly classify contemporary recommendation methods into three types [19]: user-item interaction information based, social relation information [1, 3, 13, 15, 22] based and heterogeneous information based. With the prevalence of social media, social recommendation techniques [1, 3, 13, 14, 15, 22] continue to spring up, which mainly leverage rich social relations among users, such as following relations in Twitter. As the information on the web is increasingly complex, heterogeneous information network has become one of the hottest research topic. And series of recommendation techniques [4, 12, 19, 20, 23, 24, 25] based on heterogeneous information network are created.

However, for the efficiency and deployment, recommender systems are often built using Collaborative Filtering (CF) [5]. CF techniques only relies on user-item past interaction information, e.g., users' browsing history or product ratings. In order to establish recommendations, it does require to collect user-item data. There are two types of CF models: the neighborhood approach and latent factor models.

The neighborhood approach focus on gaining similarity information between items or between users based on user-item interaction matrix, for example, user-item rating matrix. The user-oriented approach computes similarities between users and evaluates the preference of a user to an item based on the ratings of similar users to the same item. Symmetrically, the item-oriented approach computes similarities between items and evaluates the preference of a user to an item based on the ratings of similar items rated by the same user.

Latent factor model, best-known as low-rank matrix factorization, has shown its effectiveness and efficiency in recommender systems since the Netflix Prize competition [2] commenced. These years, many researchers have been attracted to this discipline and proposed a series of low-rank matrix factorization based methods. Salakhutdinov et al. presented probabilistic algorithms that scale linearly with the number of observations and proposed the Probabilistic Matrix Factorization (PMF) [17]. Yehuda Koren [9] designed an SVD-like low-rank decomposition of the rating matrix and dubed this basic model SVD. In order to get more accurate rating prediction, Koren further integrated implicit feedback into SVD and proposed an updated version called SVD++.

Instead of assuming that user-item matrix has low-rank globally, Lee at al. [18] assumed that user-item matrix behaves as a low-rank matrix in the vicinity of certain row-column combinations, then proposed Local Low-Rank Matrix Approximation (LLORMA). Recently, the authors further combined LLORMA with a general empirical risk minimization for ranking losses and proposed a model called Local Collaborative Ranking (LCR) [10].

3. PRELIMINARY

3.1 Low Rank Matrix Factorization

Low rank matrix factorization method assumes that the observed matrix M is globally low rank, we recall here the notations from the previous section: the matrix $M \in R^{n_1 * n_2}$ (n_1 represents the number of users, n_2 represents the number of items) denotes the matrix of user-item ratings, and the observed training data is $\{(i, j, M_{i,j}) : (i, j) \in A\}$ where A is the set of user-item training ratings. A low-rank factorization of M is denoted by $M = UV^T$, where $U \in R^{n_1 * r}$,$V \in R^{n_2 * r}$, where r is the dimension number of latent factors and $r \ll \min(n_1, n_2)$. U and V represent usersąŕ and itemsąŕ distributions on latent semantic, respectively. The idea behind such model is that attitudes or preferences of a user are determined by a small number of unobserved factors. In a linear factor model, a userąŕs preferences are modeled by linearly combining item factor vectors using user-specific coefficients. For example, for n_1 users and n_2 movies, the $n_1 * n_2$ preference matrix M is given by the product of an $n_1 * r$ user factor matrix U and an $n_2 * r$ item factor matrix V. Training such a model amounts to finding the best approximation to the observed $n_1 * n_2$ target matrix M under the given loss function.

$$\underset{U,V}{argmin} \sum_{(i,j)\in A} \left([UV^T]_{i,j} - M_{i,j}\right)^2 + \lambda(\|U\|^2 + \|V\|^2) \quad (1)$$

3.2 Local Low-Rank Matrix Factorization

On the basis of that the observed matrix is of locally low rank, Joonseok Lee [11] proposed a local low-rank matrix ap-proximation method (LLORMA) which use a plurality of matrix to cover the large matrices. Figure 1 shows such an example representing how the local low-rank matrices describe the original matrix. The method assumes that the space of (row, column) pairs $\Phi = \{(u, i) : u = 1 \cdots m, i =$

$1 \cdots n\}$ is endowed with a distance function d that measures distances between pairs of users and items. The distance function leads to the notion of neighborhoods of user-item pairs, and local low-rank assumption states that M can be approximated by a set of low-rank matrices, where each local matrix is low-rank. Thus, M is approximated by a number of low-rank matrices, one for each neighborhood, and each of these matrices describes the original matrix for some subset of users and items. The LLORMA assumes that the matrix M is not low-rank but is locally low-rank. The intuition behind this assumption is that the entire rating matrix M is not low-rank but a submatrix restricted to certain types of similar users and items (for example, old users viewing documentary film) is low-rank.

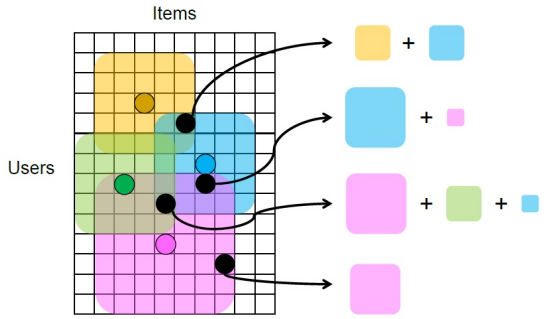


Figure 1: Local low-rank matrix factorization

The method randomly selects a serious of user-item (row, column) pairs as anchor-points , and then sample elements to construct a local matrix with in a distance d to the certain anchor point. It estimates a low-rank approximation for each neighborhood by minimizing the squared reconstruction error, weighted by the proximity of the reconstruction site to the anchor point. Formally, each local model is learned by

$$\arg\min_{U,V} \sum_{(i,j)\in A}^{K} K((i',j'),(i,j))([UV^T]_{i,j} - M_{i,j}])^2 \qquad (2)$$

where $K((i',j'),(i,j))$ is a two-dimensional smoothing kernel that measures the proximity of the reconstruction site (i,j) to the anchor point (i',j') . This kernel function may be defined in several ways. The smoothing kernels are inversely related to distance function. The optimization problem above is essentially a weighted version of the global matrix factorization problem, but it needs to be repeated q times - once for each anchor point. Unfortunately, it is computationally impractical to solve a new weighted matrix factorization problem above for all user-item prediction pairs. Thus, instead of treating each test user-item pair as an anchor point and solving the corresponding model (2), the anchor points are selected before the test user-item pairs are observed. The q anchor points lead to q local models that are then linearly combined to provide the estimate of the test user-item rating. The specific linear combination rule is given by locally constant regression or non-parametric Nadaraya-Watson regression. So we finally get the objective function

$$\hat{M}_{i,j} = \sum_{t=1}^{q} \frac{K_h((i_t,j_t),(i,j))}{\sum_{s=1}^{q} K_h((i_s,j_s),(i,j))} [U_t V_t^T]_{i,j} \qquad (3)$$

where (i_t, j_t) is the anchor point of local model t. In other words, (3) is a convex combination of each local model's prediction, ensuring that points (i,j) closer to the queried point contribute more than those far from it. More details on locally constant regression and other forms of non-parametric regression may be found in any book on non-parametric statistical modeling, for example [21]. This method can solve the questions accurately, however it also has shortcomings. In the following, we propose a new method to solve the matrix approximation method.

4. THE CLLORMA METHOD

In this section, we first introduce the disadvantages of low rank matrix factorization and the basic idea of our method, and then we present our method in detail.

4.1 Disadvantages Of Current Methods

In the previous section, we introduced local low-rank matrix approximation. The model is characterized by multiple low-rank matrices. The local low-rank matrix approximation method is based on low-rank so as to extract user features better. However, it randomly selects anchor-points at the first step which may lead to an uneven local matrix combination. With intuitive views, the original matrix is covered by a certain amount of local low-rank matrices, some areas are covered by majority of local matrices and some areas are not covered. And thus, randomly selecting anchor-points may lead to three shortcomings. First, some rating intensive areas may not be covered while rating sparse areas may be covered excessively. Second, randomly selecting anchor-points method may make the local matrices distribute unevenly. In Figure 2, for example, region A (dense area) is covered by one matrix while region B (sparse area) is covered by a number of matrices. Last but not least, respective points may not be selected as anchor-points.

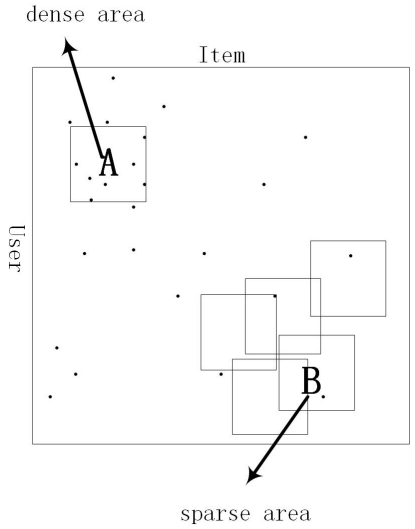


Figure 2: Choose anchor-points randomly on the rating matrix.

4.2 Basic Idea

In order to use the matrix information thoroughly, we propose a representative and uniform anchor-point selection

method to consummate the local low-rank matrix factorization method which regards the original matrix as a weighted sum of local low-rank matrices. We can solve the disadvantages mentioned in the previous section by considering the rating density around anchor-points and distance between anchor-points. Specifically, from the original rating matrix, we have the user and item rating information, and then we cluster the user and item using rating information by K-Means method. Next, considering rating density of different clustering species and the distance between these clustering center-points, we select the anchor-points from these center-points according to the clustering result. At last, we set up local low-rank matrices around these anchor-points, and then combine these local matrices together to describe the original matrix. Next, we will present these steps.

4.3 Cluster Latent Factor Matrix Of User And Item

In order to make the anchor-points distribute evenly and to be the most representative point, we decide to use clustering method to divide the original matrix into uniform grids. Empirically, we found that using standard distance in clustering method such as Euclidean distance or cosine distance do not perform well when the clustering vector is sparse. We therefore factorize M using PMF [17] and obtain the user factor matrices U and item factor matrix V . The matrix factorization (MF) method for the original matrix is

$$\underset{U,V}{argmin} \sum \left([UV^T]_{i,j} - M_{i,j}\right)^2 + \lambda(\|U\|^2 + \|V\|^2) \qquad (4)$$

We define the user latent factor $u_i, i = 1 \cdots n_1$ in U and item latent factor $v_j, j = 1 \cdots n_2$ in V. Then we cluster the user latent factor matrix u_i and item latent factor matrix v_j which have been got. The proposed method uses K-Means clustering method and we finally get k clusters. After obtain user clusters and item clusters, we propose a heuristic anchor-point selecting method. We can get the center points and the group of the users and items through this method. We cluster different types of users and items together, however, each kind of users may also have other properties that belong to another cluster. For example, the audience who love comedy may also love action movie. So we should not only calculate the distance in the cluster but also the distance among different usersąŕ and itemsąŕ clusters.

By clustering latent factors, we get the distance between all the users and all the user clustering center and also the distance between all the items and all the item clustering center. Next we will combine item clustering center and user clustering center together. We represent user cluster center by $u'_x, x = 1 \cdots k$ and item clustering center by $v'_y, y = 1 \cdots k$. Then we combine the user clustering center and the item clustering center as candidate anchor-point $(u'_x, v'_y), (x = 1 \cdots k, y = 1 \cdots k)$(show in Figure 3). And then we can calculate the distance between center points and all the user-item pairs.

We then define the distance between user u_i and user cluster center u'_x is

$$d(u'_x, u_i) = arccos\left(\frac{\langle {u'}_x, u_i\rangle}{\|{u'}_x\| \cdot \|u_i\|}\right) \quad (x = 1 \cdots k, i = \cdots n_1) \qquad (5)$$

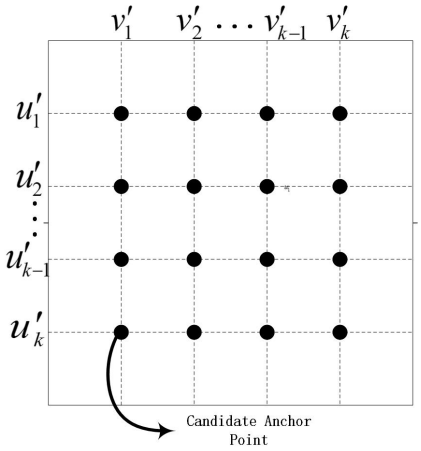


Figure 3: Candidate anchor point.

and the distance between item v_j and item cluster-center v'_y is

$$d(v'_y, v_j) = arccos\left(\frac{\langle {v'}_y, v_j\rangle}{\|{v'}_y\| \cdot \|v_j\|}\right) \quad (y = 1 \cdots k, j = \cdots n_2) \qquad (6)$$

The u_i,u'_x are the i-th row of the matrix U and the x-th center point of the user cluster. The v_j,v'_y are the j-th column of the matrix V and the y-th center point of the item cluster. Now we define the point distance

$$d((u_x{}', v_y{}'), (u_i, v_j)) = d(u_x{}', u_i) \times d(v_y{}', v_j) \qquad (7)$$

which reflects the similarity between the rows u_i and center-point u'_x and columns v_j and center-point v'_y. The greater the distance is the smaller the similarity is. The next step is to calculate the similarity between the anchor-points and the all the (user, item) points in the original matrix. In this paper we use a kernel function

$$K_h((u_x{}', v_y{}'), (u_i, v_j)) \propto (1 - d((u_x{}', v_y{}'), (u_i, v_j)))\mathbf{1}[d((u_x{}', v_y{}'), (u_i, v_j)) < h] \qquad (8)$$

to convert distance to similarity. See for instance [21] for more information on smoothing kernels. We represent similarity of any nodes (u_i, v_j) in the matrix and the anchor-points $(u_x{}', v_y{}')$ by $K_h((u_x{}', v_y{}'), (u_i, v_j))$. So now we get the user and item matrix, center points, and the similarity between center points and entries of original matrix.

4.4 Choose Anchor-Points By Two Criteria

After get the anchor-points and the similarities. Our challenge now is to select appropriate anchor-points from these cluster centers $(u'_x, v'_y), (x = 1 \cdots k, y = 1 \cdots k)$. We consider three aspects: the local matrices coverage more intensive areas; the local matrices coverage more uniform; the anchor-point contains more information. So we propose an heuristic anchor-point selecting method. So we have to get the similarity between anchor-points and the density of all the cluster combinations. In the previous section we have defined the similarity, now we come to calculate the density of the cluster. As show in Figure 4.

$\rho(u'_x, v'_y) = \frac{R_n}{u_{num} \times v_{num}}$ means the density that the local matrix $u'_x - v'_y$ coverage, $d((u_x{}', v_y{}'), (u_i{}', v_j{}'))$ means the distance between $u'_x - v'_y$-th anchor-point and $u'_i - v'_j$-th

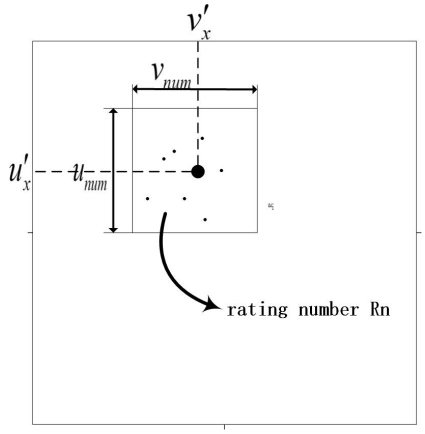


Figure 4: Cluster density.

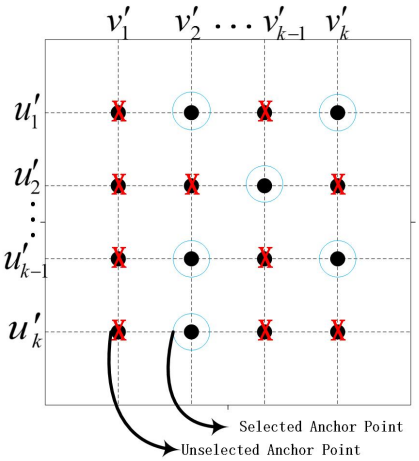


Figure 5: Heuristically select anchor-point.

anchor-point. So our anchor-points selecting method is show in Algorithm 1.

Algorithm 1 Algorithm of Selecting Anchor-Points

Require:
$\rho(u'_x, v'_y)$: area density
$d((u_x{}', v_y{}'), (u_i, v_j))$: distance between anchor-points
q: anchor-points number
k: user/item candidate anchor-points number
α: proportion coefficient

Ensure:
Φ: a collection of selected anchor-points

1: Randomly select the first anchor-point $(\dot{u}_1, \dot{v}_1)$
2: **for** $l = 2 \to q$ **do**
3: $\underset{x,y}{arg\max}_{with x,y=1\cdots k} \quad (\alpha\rho(u'_x, v'_y) + \frac{1-\alpha}{l-1}\sum_{p=1}^{l-1} d(u'_x, v'_y)(\dot{u}_p, \dot{v}_p))$
4: Expand Φ by $(\dot{u}_l, \dot{v}_l)$ with $\dot{u}_l = u'_x$,$\dot{v}_l = v'_y$
5: **end for**

We choose q anchor-points $\Phi = (\dot{u}_l, \dot{v}_l, l = 1 \cdots q)$ from $k * k$ candidate anchor-points(show in Figure 5). α is the adjustment coefficient to balance density and distance($0 \leq \alpha \leq 1$).

4.5 Combine Local Matrix

Then, it estimates a low-rank approximation for each neighborhood by minimizing the squared reconstruction error, weighted by the proximity of the reconstruction site to the anchor point. Formally, each local model is learned by

$$\arg\min_{U,V} \sum_{(i,j)\in A}^{K} K((\dot{u}_l, \dot{v}_l), (u_i, v_j))([UV^T]_{i,j} - M_{i,j}])^2 + \lambda(\|U\|^2 + \|V\|^2) \quad (9)$$

where $K((\dot{u}_l, \dot{v}_l), (u_i, v_j))$ is a two-dimensional smoothing kernel that measures the proximity of the reconstruction site (u_i, v_j) to the anchor point $(\dot{u}_l, \dot{v}_l)$. This kernel function may be defined in several ways. After these models are estimated, they are combined using

$$\hat{M}_{i,j} = \sum_{t=1}^{q} \frac{K_h((\dot{u}_t, \dot{v}_t), (u_i, v_j))}{\sum_{s=1}^{q} K_h((\dot{u}_s, \dot{v}_s), (u_i, v_j))} [U_t V_t{}^T]_{i,j} \quad (10)$$

to create the estimate.

4.6 Algorithm Framework

Algorithm 2 describes the framework of the proposed CLLORMA. Through selecting anchor-points by distance and density, CLLORMA can predict the unknown user-item ratings. The algorithm includes two main parts: (1) Selection of anchor-points (Line 1-3). (2) Combination of local matrices (Line 4-16). It is the main time-consuming component.

5. EXPERIMENTS

In this section, extensive experiments on three real datasets illustrate the traits of CLLORMA from three aspects. We first validate the effectiveness of CLLORMA through comparing it with representative methods. Then we thoroughly exploit the efficiency of the proposed method. Finally, we illustrate the effect of parameter α on performances.

5.1 Datasets

In order to validate the effect on different types and sizes of datasets, we use three popular datasets: MovieLens 100K, MovieLens 1M and Douban Movie. More details of our datasets can be found in Table 1.

Table 1: Statistics of Datasets

Name	#Users	#Items	#Rating	Density
MovieLens 100K	943	1682	100000	6.30%
MovieLens 1M	6940	3952	1000209	3.65%
DoubanMovie	1031	1474	70131	4.61%

MovieLens data sets [10],[11] were collected by the GroupLens Research Project at the University of Minnesota. It is the oldest public dataset of the recommender system. MovieLens 100K consists of 100,000 ratings (1-5) from 943 users on 1682 movies, and each user has rated at least 20 movies. It has dense rating information at the density of 6.3%. MovieLens 1M contains 1,000,209 anonymous ratings of approximately 3,900 movies made by 6,040 MovieLens users who joined MovieLens in 2000. It has sparse density rating information at the density of 4.2%. Douban Movie [19],[20] is a well known so-cial media network in China. The dataset consists of 70131 rating (1-5) from 1031 users on 1474 movies. It has medium dense rating information at the density of 4.6%.

Algorithm 2 Algorithm Framework of CLLORMA

Require:
M: original rating matrix
h: kernel function
q: local matrix number
α: balance parameters defined above
λ: regularization coefficient

Ensure:
$\hat{T}(s_t) = U^{(t)}V^{(t)^T}, t = 1, ..., q$: a group of latent factor of users and items

1: $\arg\min_{U,V} \sum_{(i,j)\in A} ([UV^T]_{i,j} - M_{i,j})^2 + \lambda(\|U\|^2 + \|V\|^2)$
2: Run the K-Means algorithm on U and V to obtain the center points $(u'_1, ..., u'_k)$ and $(v'_1, ..., v'_k)$, the density $\rho(u'_x, v'_y)$ in each class, and the distance $d((u_x{}', v_y{}'), (u_i{}', v_j{}'))$of the points among different cluster centers.
3: From Algorithm 1, we get anchor-points $\Phi = (\dot{u}_l, \dot{v}_l, l = 1 \cdots q)$
4: **for** $x = 1 \to q$ **do**
5: **for** $i = 1 \to n_1$ **do**
6: $[K_h^{(\dot{u}_x)}]_i = (1 - d(u_i, \dot{u}_x)^2)\mathbf{1}[d(u_i, \dot{u}_x) < h]$
7: **end for**
8: **end for**
9: **for** $y = 1 \to q$ **do**
10: **for** $j = 1 \to n_2$ **do**
11: $[K_h^{(\dot{v}_y)}]_j = (1 - d(v_j, \dot{v}_y)^2)\mathbf{1}[d(v_j, \dot{v}_y) < h]$
12: **end for**
13: **end for**
14: **for** all $t = 1, \ldots, q$ in parallel **do**
15: $U^{(t)}, V^{(t)} = \arg\min_{U,V} \sum_{(i,j)\in A}^{K} [K_h^{(\dot{u}_x)}]_i [K_h^{(\dot{v}_y)}]_j ([UV^T]_{i,j} - M_{i,j}])^2 + \lambda(\|U\|^2 + \|V\|^2)$
16: **end for**

5.2 Metrics

We use Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) to evaluate the performance of different methods. The metric MAE is defined as:

$$MAE = \frac{1}{T} \sum_{i,j\in B} \left| M_{i,j} - \hat{M}_{i,j} \right| \tag{11}$$

where $M_{i,j}$ is the rating user i gave to item j , B is the test set and $\hat{M}_{i,j}$ denotes the rating user i gave to item j as predicted by a method. Particularly, $\hat{M}_{i,j}$ can be calculated by (10) in our model. Moreover, T is the number of tested ratings. The metric RMSE is defined as:

$$RMSE = \sqrt{\frac{1}{T} \sum_{i,j\in B} \left(M_{i,j} - \hat{M}_{i,j} \right)^2} \tag{12}$$

From the definitions, we can see that a smaller MAE or RMSE means better performance.

5.3 Compared Method

In order to validate the effectiveness of CLLORMA, we compare four versions of CLLORMA with the two state of the art matrix factorization methods.

• PMF. This method is a typical matrix factorization method proposed by Salakhutdinov and Minh [17].And in fact it is equivalent to global basic low-rank matrix factorization.

• LLORMA. This method is a typical matrix factorization method proposed by Joonseok Lee et al. [11]. And in fact it is equivalent to basic local low-rank matrix factorization.

• CLLORMA-R. This local low-rank matrix factorization method randomly selects anchor-point from the cluster center which have been obtained.

• CLLORMA-De. This local low-rank matrix factorization method considers cluster rating density to select anchor-point from the cluster centers which have been obtained.

• CLLORMA-Di. This local low-rank matrix factorization method considers distance between anchor-points from the cluster centers which have been obtained.

• CLLORMA. This local low-rank matrix factorization method considers the density and distance evenly to select anchor-point from the cluster centers which have been obtained.

5.4 Effectiveness Experiments

This section will validate the effectiveness of CLLORMA through comparing its different variations to baselines. For a fair comparison of PMF, LLORMA and CLLORMA, we use the same parameters in both methods. For all the experiments in this paper the λ is set to a trivial value 0.001 and the latent factor number is fixed to 10. In local matrix model such as LLORMA and CLLORMA series methods, we used the Epanechikov kernel and set $h = 0.8$, the balance parameter $\alpha = 0.5$, the maximum number of iterations $q = 50$, and we user the L_2 regularization coefficient. The different parameter is anchor-points number, the LLORMA has 50 anchor-point while the CLLORMA series methods only has 25 points.

For these three datasets, we user different ratios(50%, 60%, 70%, 80%) of data as training data. For example, the training data 80% means that we select 80% of the ratings from user-item rating matrix as the training data to predict the remaining 20% of ratings. The random selection was carried out 10 times independently in all the experiments.We report the average results on three different datasets and also record the improvement of all methods compared to the baseline PMF.

The performance of all the methods are show in Tables 2-4. And we can get the following conclusions. Four versions of CLLORMA always perform better than the original method on each data set and all ratios. CLLORMA is the best one of our four versions of methods. It always performs better than CLLORMA-De and CLLORMA-Di while these two methods perform better than CLLORMA-R. By comparing the three data sets, we can find that on denser datasets the CLLORMA can performs better.

5.5 Efficiency Study

Experiments in this section will validate the efficiency of CLLORMA compared to LLORMA. We use the ratio 80% of MovieLens 100K as training set, and the rest of data as test set. We gradually increase the number of local matrix and simultaneously record the MAE and RMSE of both methods on the same matrix number.

As show in Figure 6, both methods improve as matrix number increases. When the local matrix number is less than 10, the RMSE and MAE of CLLORMA decline faster than LLORMA. When the local matrix number close to 25, the RMSE and MAE of CLLORMA obtain good convergence result. However, the RMSE and MAE of LLORMA

Table 2: Performances of different methods on MovieLens 100K (the baseline of improved performance is PMF)

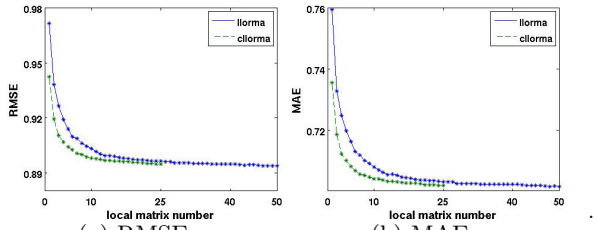
Training	Metrics	PMF	LLORMA	CLLORMA-R	CLLORMA-De	CLLORMA-Di	CLLORMA
80%	MAE	0.7260	0.7081	0.6995	0.6988	0.6988	**0.6987**
	Improve		2.47%	3.65%	3.75%	3.75%	3.76%
	RMSE	0.9209	0.9033	0.9001	0.8926	0.8916	**0.8910**
	Improve		1.91%	2.26%	3.07%	3.18%	3.24%
70%	MAE	0.7377	0.7150	0.7131	0.7134	0.7125	**0.7111**
	Improve		3.08%	3.33%	3.29%	3.42%	3.61%
	RMSE	0.9359	0.9090	0.9084	0.9078	0.9078	**0.9060**
	Improve		2.87%	2.94%	3.00%	3.00%	3.19%
60%	MAE	0.7458	0.7282	0.7234	0.7165	0.7231	**0.7163**
	Improve		2.36%	3.00%	3.93%	3.04%	3.96%
	RMSE	0.9475	0.9207	0.9207	0.9118	0.9204	**0.9113**
	Improve		2.83%	2.83%	3.77%	2.86%	3.82%
50%	MAE	0.7579	0.7367	0.7332	0.7339	0.7328	**0.7324**
	Improve		2.80%	3.26%	3.17%	3.31%	3.36%
	RMSE	0.9611	0.9359	0.9330	0.9332	0.9320	**0.9317**
	Improve		2.62%	2.92%	2.90%	3.03%	3.06%

Table 3: Performances of different methods on MovieLens 1M (the baseline of improved performance is PMF)

Training	Metrics	PMF	LLORMA	CLLORMA-R	CLLORMA-De	CLLORMA-Di	CLLORMA
80%	MAE	0.6735	0.6696	0.6597	0.6599	0.6598	**0.6596**
	Improve		0.58%	2.05%	2.02%	2.03%	2.06%
	RMSE	0.8535	0.8531	0.8455	0.8459	0.8459	**0.8445**
	Improve		0.05%	0.94%	0.89%	0.89%	1.05%
70%	MAE	0.6804	0.6755	0.6640	0.6636	0.6639	**0.6635**
	Improve		0.72%	2.41%	2.47%	2.43%	2.48%
	RMSE	0.8624	0.8596	0.8499	0.8496	0.8498	**0.8492**
	Improve		0.32%	1.45%	1.48%	1.46%	1.53%
60%	MAE	0.6844	0.6814	0.6674	0.6672	0.6672	**0.6670**
	Improve		0.44%	2.48%	2.51%	2.51%	2.54%
	RMSE	0.8682	0.8666	0.8538	0.8543	0.8539	**0.8537**
	Improve		0.18%	1.66%	1.60%	1.65%	1.67%
50%	MAE	0.6930	0.6823	0.6758	0.6758	0.6758	**0.6756**
	Improve		1.54%	2.48%	2.48%	2.48%	2.51%
	RMSE	0.8787	0.8791	0.8634	0.8637	0.8634	**0.8632**
	Improve		-0.05%	1.74%	1.71%	1.74%	1.76%

Table 4: Performances of different methods on Douban Movie (the baseline of improved performance is PMF)

Training	Metrics	PMF	LLORMA	CLLORMA-R	CLLORMA-De	CLLORMA-Di	CLLORMA
80%	MAE	0.5630	0.5589	0.5556	0.5562	0.5561	**0.5554**
	Improve		0.73%	1.31%	1.21%	1.23%	1.35%
	RMSE	0.7084	0.7067	0.7027	0.7024	0.7025	**0.7022**
	Improve		0.24%	0.80%	0.85%	0.83%	0.88%
70%	MAE	0.5625	0.5580	0.5552	0.5547	0.5552	**0.5546**
	Improve		0.80%	1.30%	1.39%	1.30%	1.40%
	RMSE	0.7082	0.7069	0.7032	0.7028	0.7031	**0.7027**
	Improve		0.18%	0.71%	0.76%	0.72%	0.78%
60%	MAE	0.5674	0.5653	0.5626	0.5615	0.5617	**0.5614**
	Improve		0.37%	0.85%	1.04%	1.00%	1.06%
	RMSE	0.7162	0.7167	0.7123	0.7119	0.7122	**0.7118**
	Improve		-0.01%	0.54%	0.60%	0.56%	0.61%
50%	MAE	0.5697	0.5665	0.5641	0.5639	0.5645	**0.5638**
	Improve		0.56%	0.98%	1.02%	0.91%	1.04%
	RMSE	0.7182	0.7179	0.7143	0.7142	0.7145	**0.7141**
	Improve		0.04%	0.54%	0.56%	0.52%	0.57%



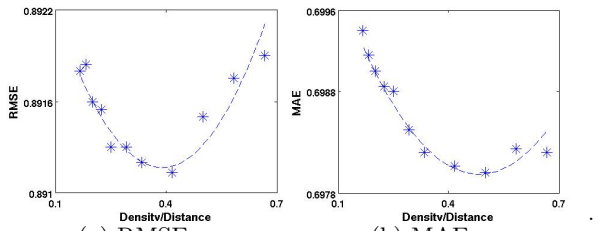
(a) RMSE (b) MAE

Figure 6: The comparison of the LLORMA and CLLORMA.

converge when the local matrix number close to 50. Overall, CLLORMA at least double the efficiency of local low-rank matrix approximation. The experiments show that CLLORMA not only performs better on accuracy but also on efficiency.

5.6 Parameter Study On α

In this section, we study the impact of parameter α which is already described in Algorithm 1. The parameter α controls the proportion of the distance and the density when we select the anchor-points from the clustering center points. The density ratio will be decreased and the distance ratio will be increased when α is reduced. So the anchor-points will be changed with different α. In this experiment, we will observe the performance of CLLORMA with α from 0.1 to 0.7. We use the ratio 20% of MovieLens 100K as test set. Figure 7 graphs the RMSE and MAE with different balance parameters α.



(a) RMSE (b) MAE

Figure 7: Performance of LLORMA and CLLORMA on RMSE and MAE with varying α.

The result changes into an arc along with the α grows. From the results, we come to a conclusion that the parameters α influence the experiment results by changing the density - distance ratio and CLLORMA will achieve the best result on a specific α. We can imagine that for different data sets, the suitable α may be not the same, but there is always a best α to achieve the best result.

6. CONCLUSIONS

In this paper, we use local matrix factorization to predict the unknown user-item ratings in the original matrix. In order to overcome the disadvantages in local low-rank matrix approximation of randomly selecting anchor-points, we propose a heuristic anchor-point selecting method which considers area density and points distance. CLLORMA makes a better performance on rating prediction accuracy and increase the efficiency for local low-rank matrix approximation. We analyze the performance of CLLORMA in terms of its dependency on the matrix size, training set size, local matrix number, and α, our method performs well in all cases.

Acknowledgements

This work is supported in part by the National Natural Science Foundation of China (No.61375058), National Key Basic Research and Department (973) Program of China (No.2013CB329606), and the Co-construction Project of Beijing Municipal Commission of Education.

7. REFERENCES

[1] A. Bellogín, I. Cantador, and P. Castells. A comparative study of heterogeneous item recommendations in social systems. *Information Sciences*, 221:142–169, 2013.

[2] J. Bennett and S. Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[3] I. Cantador, A. Bellogín, and D. Vallet. Content-based recommendation in social tagging systems. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 237–240. ACM, 2010.

[4] W. Feng and J. Wang. Incorporating heterogeneous information for personalized tag recommendation in social tagging systems. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1276–1284. ACM, 2012.

[5] D. Goldberg, D. Nichols, B. M. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.

[6] S. H. Ha. Helping online customers decide through web personalization. *IEEE Intelligent systems*, (6):34–43, 2002.

[7] W. Hill, L. Stead, M. Rosenstein, and G. Furnas. Recommending and evaluating choices in a virtual community of use. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 194–201. ACM Press/Addison-Wesley Publishing Co., 1995.

[8] P. Kazienko and M. Kiewra. Personalized recommendation of web pages. *Chapter*, 10:163–183, 2004.

[9] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 426–434. ACM, 2008.

[10] J. Lee, S. Bengio, S. Kim, G. Lebanon, and Y. Singer. Local collaborative ranking. In *Proceedings of the 23rd international conference on World wide web*, pages 85–96. ACM, 2014.

[11] J. Lee, S. Kim, G. Lebanon, and Y. Singer. Local low-rank matrix approximation. In *Proceedings of The 30th International Conference on Machine Learning*, pages 82–90, 2013.

[12] C. Luo, W. Pang, Z. Wang, and C. Lin. Hete-cf: Social-based collaborative filtering recommendation using heterogeneous relations. In *Data Mining*

(ICDM), 2014 IEEE International Conference on, pages 917–922. IEEE, 2014.

[13] H. Ma, I. King, and M. R. Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 203–210. ACM, 2009.

[14] H. Ma, H. Yang, M. R. Lyu, and I. King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 931–940. ACM, 2008.

[15] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296. ACM, 2011.

[16] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[17] R. Salakhutdinov and A. Mnih. Probabilistic matrix factorization. Citeseer, 2011.

[18] U. Shardanand and P. Maes. Social information filtering: algorithms for automating "word of mouth". In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217. ACM Press/Addison-Wesley Publishing Co., 1995.

[19] C. Shi, J. Liu, F. Zhuang, P. S. Yu, and B. Wu. Integrating heterogeneous information via flexible regularization framework for recommendation. *arXiv preprint arXiv:1511.03759*, 2015.

[20] C. Shi, Z. Zhang, P. Luo, P. S. Yu, Y. Yue, and B. Wu. Semantic path based personalized recommendation on weighted heterogeneous information networks. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 453–462. ACM, 2015.

[21] M. P. Wand and M. C. Jones. *Kernel smoothing*. Crc Press, 1994.

[22] X. Yang, H. Steck, and Y. Liu. Circle-based recommendation in online social networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1267–1275. ACM, 2012.

[23] X. Yu, H. Ma, B.-J. P. Hsu, and J. Han. On building entity recommender systems using user click log and freebase knowledge. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 263–272. ACM, 2014.

[24] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 283–292. ACM, 2014.

[25] X. Yu, X. Ren, Y. Sun, B. Sturt, U. Khandelwal, Q. Gu, B. Norick, and J. Han. Recommendation in heterogeneous information networks with implicit user feedback. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 347–350. ACM, 2013.