GLOMA: Embedding Global Information in Local Matrix Approximation Models for Collaborative Filtering

Chao Chen,[1] Dongsheng Li,[1*] Qin Lv,[2] Junchi Yan,[31] Li Shang,[2] Stephen M. Chu[1]
[1]IBM Research – China, Shanghai, P.R. China, 201203
[2]Univeristy of Colorado Boulder, Boulder, Colorado, USA, 80309
[3]East China Normal University, Shanghai, P.R. China, 200062
chench.resch@gmail.com, {ldsli, schu}@cn.ibm.com, {qin.lv, li.shang}@colorado.edu, jcyan@sei.ecnu.edu.cn

Abstract

Recommender systems have achieved great success in recent years, and matrix approximation (MA) is one of the most popular techniques for collaborative filtering (CF) based recommendation. However, a major issue is that MA methods perform poorly at detecting strong localized associations among closely related users and items. Recently, some MA-based CF methods adopt clustering methods to discover meaningful user-item subgroups and perform ensemble on different clusterings to improve the recommendation accuracy. However, ensemble learning suffers from lower efficiency due to the increased overall computation overhead.

In this paper, we propose GLOMA, a new clustering-based matrix approximation method, which can embed global information in local matrix approximation models to improve recommendation accuracy. In GLOMA, a MA model is first trained on the entire data to capture global information. The global MA model is then utilized to guide the training of cluster-based local MA models, such that the local models can detect strong localized associations shared within clusters and at the same time preserve global associations shared among all users/items. Evaluation results using MovieLens and Netflix datasets demonstrate that, by integrating global information in local models, GLOMA can outperform five state-of-the-art MA-based CF methods in recommendation accuracy while achieving descent efficiency.

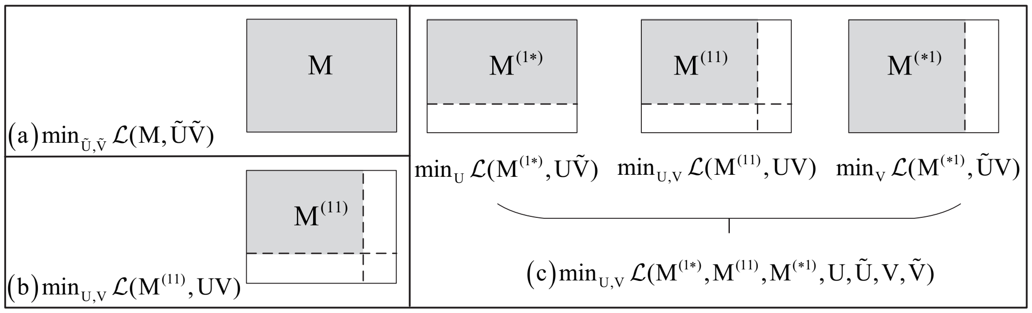
Introduction

In today's recommender systems, matrix approximation (MA) is one of the most commonly-used collaborative filtering (CF) methods. The goal is to predict users' missing ratings on targeted items. More formally, given partially observed user-item rating matrix M with low-rank, user i and item j are characterized by vectors of latent factors U_{i*} and V_{*j}, respectively, then the unknown rating $M_{i,j}$ can be predicted by the dot product of U_{i*} and V_{*j} (Su and Khoshgoftaar 2009). Although MA-based CF methods have achieved good success, Koren et al. (2008) pointed out that MA models can effectively estimate overall structures that relate simultaneously to most or all items, but they perform poorly at detecting strong associations among a small set of items. Therefore, it is desirable to consider local associations to enhance model accuracy.

To this end, many clustering-based ensemble methods have been proposed (Zhang et al. 2013; Lee et al. 2013; Chen et al. 2015), in which clustering methods are first applied to discover strong local associations and then MA methods can be applied in parallel on the submatrices corresponding to the clustering-based partitions. Utilizing the parallelism of modern many-core and distributed architectures, the running time can be largely reduced. However, the prediction quality of local models based on submatrices is generally low, because 1) local models may easily overfit due to insufficient training data in clusters and 2) the overall associations shared among all users/items are not captured. To address the above issues, clustering methods have to be run many times in order to produce different approximated matrices containing different types of information, and then ensemble methods are used to improve recommendation quality. However, ensemble learning leads to another problem: the overall computation overhead is usually very high, e.g., WEMAREC (Chen et al. 2015) required $8\times$ more overall computation than RSVD (Paterek 2007).

In this paper, we propose GLOMA, a scalable and accurate MA-based method which can capture both localized relationships in submatrices and global structure among all users and items. Similar to existing clustering-based ensemble methods, GLOMA adopts clustering method to explore the localized relationships. However, instead of relying on ensemble techniques, GLOMA attempts to embed a global MA model, which contains the global information, into the learning process of local MA models. With the help of this embedded model, every local model can be trained on data which includes all the ratings corresponding to users and items in the same cluster, such that the overall structures can be obtained and the insufficient data issue can be alleviated. By leveraging multi-task feature learning techniques, GLOMA can capture more abundant and diverse features than standard MA methods, and has the potential to achieve better recommendation accuracy. To further improve the accuracy and efficiency of GLOMA, a new clustering method following the idea from (Chen et al. 2015) is proposed, in which every user or item is characterized by its rating distribution. Then, the dimension of the data space can be largely reduced, resulting in the reduction of running time. The pro-

[*]Corresponding author.



M
$(a)\min_{\tilde{U},\tilde{V}} \mathcal{L}(M, \tilde{U}\tilde{V})$
$M^{(11)}$
$(b)\min_{U,V} \mathcal{L}(M^{(11)}, UV)$
$M^{(1*)}$
$M^{(11)}$
$M^{(*1)}$
$\min_U \mathcal{L}(M^{(1*)}, U\tilde{V})$
$\min_{U,V} \mathcal{L}(M^{(11)}, UV)$
$\min_V \mathcal{L}(M^{(*1)}, \tilde{U}V)$
$(c)\min_{U,V} \mathcal{L}(M^{(1*)}, M^{(11)}, M^{(*1)}, U, \tilde{U}, V, \tilde{V})$

Figure 1: Comparison of matrix approximation models for collaborative filtering: (a) standard low-rank model, (b) clustering-based model, and (c) the proposed GLOMA model, where the rows and columns of the rating matrix M represent the users and items respectively and shaded area represent the training data used for each individual model.

posed GLOMA method is evaluated using two real-world benchmark datasets (MovieLens and Netflix), and the experimental results demonstrate that the GLOMA method outperforms five state-of-art MA-based CF methods in recommendation accuracy while achieving descent efficiency.

Related Work

Collaborative filtering (CF) is widely used in the business world for simplicity of implementation and high quality of recommendation. Breese et al. (1998) categorized CF approaches into two classes: memory-based and model-based algorithms. Memory-based algorithms, such as user-based (Herlocker et al. 1999) and item-based (Sarwar et al. 2001) methods, build neighborhood relationship for every user, and usually use the weighted sum of the ratings to predict missing values. However, the user/item similarities cannot be calculated accurately without sufficient ratings, so these approaches suffer from the data sparsity problem.

In contrast, model-based algorithms first learn a model from the training data then use it to make predictions. Matrix approximation (MA) methods, which as illustrated in Figure 1(a) learn user features $\tilde{U}$ and item features $\tilde{V}$ from the entire rating matrix, have been empirically and theoretically demonstrated to have the capacity of recovering the rating matrix accurately from a small number of observations (Srebro and Jaakkola 2003; Candès and Tao 2010). Specifically, Billsus et al. (1998) first introduced SVD to the domain of CF, and proved that MA-based methods can effectively alleviate the data sparsity issue. Salakhutdinov et al. (2007) explained the MA-based CF algorithms in a Bayesian perspective, developed a probabilistic matrix factorization (PMF) method, and moreover constructed BPMF, a Bayesian extension of PMF method (Salakhutdinov and Mnih 2008). However, as summarized by Koren et al. (2008), these MA-based models are generally effective at estimating overall structure that relates simultaneously to most or all items, but perform poorly at detecting strong associations among a small set of closely related items.

To address this issue, clustering-based CF methods have been proposed, where clustering techniques (Chen et al. 2015) and community detection methods (Zhang et al. 2013) are adopted to find user-item subgroups with strong correlations. For example, as shown in Figure 1(b), the entire rating matrix is divided into four submatrices. The expensive MA task is equivalently divided into smaller subproblems which can be solved in parallel. However, due to insufficient training data in each detected user-item cluster, there can be severe overfitting and the recommendation quality can be much worse. Therefore, ensemble techniques are always adopted to achieve better accuracy, such as DFC (Mackey, Jordan, and Talwalkar 2011), LLORMA (Lee et al. 2013), and WEMAREC (Chen et al. 2015), all of which attempted to use multiple local models to describe every user-item rating, and then use the weighted sum of the predictions from multiple local models to estimate the missing ratings. In other words, these clustering and ensemble based methods try to discover various strong associations contained in user-item subgroups, and aim to improve prediction accuracy by identifying and leveraging more diverse clusters.

Some recent works also attempted to train a singleton model integrated with multiple types of relations. For example, Singh et al. (2008) shared parameters among factors when decomposing multiple matrices represented for multiple relations to learn different type of user behaviors. Yuan et al. (2014) used group sparsity regularization to automatically transfer information among multiple types of behaviors. In additional, Chen et al. (2016) assumes every user-item rating is depicted by a Gaussian mixture model with three component, each of which containing relational information of different level.

Different from existing works, we develope a new way to solve the problem of MA-based methods by directly embedding a previously-trained model containing global information into the procedure of training local models. As illustrated in Figure 1(c), $\tilde{U}$ and $\tilde{V}$ are the embedded features, and U and V are desired features. The intuition behind the idea is, if U could be trained on the submatrix $M^{(1*)}$ by introducing $\tilde{V}$ that can describe all the items, U would be able to learn the global relations of its corresponding users, because $M^{(1*)}$ has the entire historical records of the user set. Similarly, item features V follows the same theory. Based on this, we share the latent factors U and V across multiple tasks such that the proposed GLOMA method can learn different tasks simultaneously to achieve better performance by using multi-task feature learning techniques (Evgeniou and Pontil 2007; Ando and Zhang 2005).

GLOMA Algorithm Design

In this section, we first formulate the GLOMA problem, then introduce a gradient-based learning algorithm to solve the problem, and finally discuss its application in the model update scenario.

Problem Formulation

We first introduce the notations used in this paper. Upper case letters represent matrices, such as a matrix $R \in \mathbb{R}^{m \times n}$ with m users and n items. R_{i*} is the i-th row vector, R_{*j} is the j-th column vector, and $R_{i,j}$ is the entry in the i-th row and j-th column. In addition, the Frobenius norm is adopted in this paper, which is defined as $||R|| := \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} R_{ij}^2}$.

As we mentioned before, the dilemma for the existing clustering-based CF methods without using ensemble strategy is the degraded recommendation quality due to the lack of sufficient data in each local model, while the advantage of high-efficiency and localized associations contained in certain small-scale user-item subgroups diminishes when more data are used to train local models.

To address this issue, a straightforward idea is to keep the local model training in its own submatrix, and additionally user/item features are trained separately in expanded data in order to learn the global information among all the ratings of related users and items. For example, given a $f \times g$ clustering, for arbitrary submatrix $M^{(rc)}$ ($c \in [f]$, $r \in [g]$), the submatrix $M^{(r*)}$ has all historical records for users of the subgroup, so if user features U were trained on $M^{(r*)}$, then user features U would be able to obtain the overall structures of those users by solving the squared error optimization function

$$\min_{U,\dot{V}} \mathcal{L}(M^{(c*)}, U\dot{V}) = \sum_i \sum_j (M_{i,j}^{(c*)} - U_{i,*}\dot{V}_{*,j})^2. \quad (1)$$

However, the problem is that the quality of user features U depends on the quality of item features $\dot{V}$ which cannot be learned properly due to insufficient item-related data. Therefore, it is reasonable to replace the untrained or randomly-initialized features $\dot{V}$ with a well-trained item features $\tilde{V}$, and the optimization problem in (1) becomes the following

$$\min_{U} \mathcal{L}(M^{(r*)}, U\tilde{V}) = \sum_i \sum_j (M_{i,j}^{(r*)} - U_{i,*}\tilde{V}_{*,j})^2. \quad (2)$$

And similarly for item features V

$$\min_{V} \mathcal{L}(M^{(*c)}, \tilde{U}V) = \sum_i \sum_j (M_{i,j}^{(*c)} - \tilde{U}_{i,*}V_{*,j})^2. \quad (3)$$

In other words, we attempt to use the previously-trained model to help local models capture the global information by planting $\tilde{U}$ and $\tilde{V}$ into the learning process of local models. Then, learning the GLOMA model can be viewed as minimizing the loss function (e.g., squared error) subject to the constraints related to embedded model

$$\begin{aligned} &\min\nolimits_{U,V} \mathcal{L}(M^{(cr)}, UV) \quad s.t. \\ &\mathcal{L}(M^{(c*)}, U\tilde{V}) \le \epsilon_1, \ \mathcal{L}(M^{(*r)}, \tilde{U}V) \le \epsilon_2 \end{aligned} \quad (4)$$

Using Lagrange multipliers, the objective function with l_2 regularizer of GLOMA method can be further presented as

$$\min\nolimits_{U,V} \mathcal{L}(M^{(cr)}, UV) + \lambda_U \parallel U \parallel^2 + \lambda_V \parallel V \parallel^2 \quad (5)$$

$$+\pi_1 \mathcal{L}(M^{(c*)}, U\tilde{V}) + \lambda_1 \parallel \tilde{V} \parallel^2 \quad (6)$$

$$+\pi_2 \mathcal{L}(M^{(*r)}, \tilde{U}V) + \lambda_2 \parallel \tilde{U} \parallel^2 \quad (7)$$

where λ_U, λ_V, λ_1 and λ_2 are the regularization parameters, and hyper-parameters π_1 and π_2 control the impacts of the introduced user/item features $\tilde{U}$ and $\tilde{V}$.

The key characteristics of GLOMA are summarized as follows:

- The terms in Equation (5) are the same as the optimization objective of clustering-based CF method, which ensure that the learned U and V can accurately capture the strong associations in certain user-item subgroups. Actually, the clustering-based method can be viewed as a special case of GLOMA by setting $\pi_1 = \pi_2 = 0$ and $\lambda_1 = \lambda_2 = 0$.
- The terms in Equation (6) and Equation (7) can help local models estimate the overall structure and avoid overfitting based on the embedded latent factors $\tilde{U}$ and $\tilde{V}$, and parameters π_1 and π_2 can be employed to control the contribution of individual user/item-specific data extension. Meanwhile, empirical results show that each of these two terms can help improve model performance, and the model satisfying these two terms will not only produce better recommendations than basic clustering-based methods, but also perform better than five state-of-the-art CF algorithms.

Learning Algorithm

In collaborative filtering, the root mean square error (RMSE) is usually adopted as the evaluation metric, which can be computed as

$$\mathcal{D}(M, \hat{M}) = \sqrt{\frac{1}{|T|} \sum_{(u,i) \in T} (\mathrm{M}_{u,i} - \hat{\mathrm{M}}_{u,i})^2} \quad (8)$$

where T denotes the set of ratings in the data and $|T|$ is the number of ratings. Therefore, we directly use RMSE as the loss function in Equations (5) to (7). As shown by Li et al. (2016), minimizing this objective function is a difficult non-convex optimization problem. To tackle this problem, we develop an iterative-based learning algorithm based on stochastic gradient descent (SGD).

Specifically, the standard subroutine in SGD just needs to compute the partial derivatives in terms of latent factors of U and V, and then iteratively updates the parameters until convergence, since the embedded features $\tilde{U}$ and $\tilde{V}$ are well-trained. However, we empirically found that continuously updating $\tilde{U}$ and $\tilde{V}$ during the training process of U and V can always produce better prediction accuracy than fixing the embedded features. The reason is that, as explained in (Chen et al. 2016), multi-task feature learning would help $\tilde{U}$ and $\tilde{V}$ unify localized relationships in user-item subgroups and common associations among all users and items, and moreover the embedded features $\tilde{U}$ and $\tilde{V}$ and desired features U and V will boost the goodness of each other. Please note that this brings another benefit – the well-trained condition of embedded model is largely relieved, which indicates that an embedded model of low accuracy can also help GLOMA produce good recommendations.

After computing the partial derivatives of parameters, there are three different ways to update the latent factors depending on which training submatrix the given training case $M_{i,j}$ belongs to. First, if the given training case $M_{i,j}$ is in $M^{(rc)}$ we modify the parameters by moving in the opposite direction of the gradient, yielding:

$$\begin{aligned} U_{i,r} &\leftarrow U_{i,r} + \gamma \cdot (\sigma_0^{-1}\Delta_0 V_{r,j} + \sigma_1^{-1}\pi_1\Delta_1\tilde{V}_{r,j} - \lambda_U U_{i,r}) \\ V_{r,j} &\leftarrow V_{r,j} + \gamma \cdot (\sigma_0^{-1}\Delta_0 U_{i,r} + \sigma_2^{-1}\pi_2\Delta_2\tilde{U}_{i,r} - \lambda_V V_{r,j}) \\ \tilde{U}_{i,r} &\leftarrow \tilde{U}_{i,r} + \gamma \cdot (\sigma_2^{-1}\pi_2\Delta_2 V_{r,j} - \lambda_2\tilde{U}_{i,r}) \\ \tilde{V}_{r,j} &\leftarrow \tilde{V}_{r,j} + \gamma \cdot (\sigma_1^{-1}\pi_1\Delta_1 U_{i,r} - \lambda_1\tilde{V}_{r,j}) \end{aligned} \tag{9}$$

where γ is the learning rate. We set σ_0, σ_1 and σ_2 as the RMSE in $M^{(rc)}$, $M^{(r*)}$, $M^{(*c)}$ respectively, and Δ_0, Δ_1, Δ_2 as the prediction errors $M_{i,j} - U_{i*}V_{*j}$, $M_{i,j} - U_{i*}\tilde{V}_{*j}$, $M_{i,j} - \tilde{U}_{i*}V_{*j}$.

Second, for a given training case $M_{i,j} \in M^{(r*)} - M^{(rc)}$ the parameters can be updated as follows

$$\begin{aligned} U_{i,r} &\leftarrow U_{i,r} + \gamma \cdot (\sigma_1^{-1}\pi_1\Delta_1\tilde{V}_{r,j} - \lambda_U U_{i,r}) \\ \tilde{V}_{r,j} &\leftarrow \tilde{V}_{r,j} + \gamma \cdot (\sigma_1^{-1}\pi_1\Delta_1 U_{i,r} - \lambda_1\tilde{V}_{r,j}) \end{aligned} \tag{10}$$

Finally, given training case $M_{i,j} \in M^{(*c)} - M^{(rc)}$ we have

$$\begin{aligned} \tilde{U}_{i,r} &\leftarrow \tilde{U}_{i,r} + \gamma \cdot (\sigma_2^{-1}\pi_2\Delta_2 V_{r,j} - \lambda_2\tilde{U}_{i,r}) \\ V_{r,j} &\leftarrow V_{r,j} + \gamma \cdot (\sigma_2^{-1}\pi_2\Delta_2\tilde{U}_{i,r} - \lambda_V V_{r,j}) \end{aligned} \tag{11}$$

Application in Model Update Scenario

Naturally, the training of the embedded model would impose extra computational overhead, but at the same time, GLOMA sheds some light on how to use out-of-date models which were trained on obsolete data to boost the performance of up-to-date models which will be trained over the latest data. Recall that a low-accuracy embedded model can still improve the model accuracy. A model can not only be used to produce recommendations, but also help to train the new model instead of being discarded when new data arrives. As such, the embedded model is not merely extra computational burden but also "wisdom from the past".

Domain-specific Data-projected Clustering

Obviously, the clustering method would impact the model performance in both accuracy and efficiency, but unfortunately we found the popular K-Means approach is too heavy (i.e., it takes nearly 48 hours in Netflix data even longer than training GLOMA model takes), because it is trying to find the subgroups where users give similar ratings on the same item such that every user (item) is characterized by its all historical ratings. It means the dimension of the data space is very high, $O(n)$ for every user and $O(m)$ for every item,

Algorithm 1 Domain-specific Data-projected Clustering

```
Input: data samples $S$, number of samples $|S|$, number of clusters $\kappa$, and KL divergence $D_{\mathrm{KL}}(\cdot \parallel \cdot)$.
Output: cluster assignment function $\mathscr{C} : [|S|] \to [\kappa]$.
 1: Randomly divide all samples into $\kappa$ clusters.
 2: while not converged do
 3:     // for each cluster $k$, set $c_k$ as the center of the assigned samples
 4:     for $k$ in $[\kappa]$ do
 5:         $c_k = \sum_{\mathscr{C}(i)=k} x_i / N_k$, where $N_k$ is the number of data samples belonging to the $k$-th cluster.
 6:     end for
 7:     //for each sample $i$ with value $x_i$, assign $i$ to the closest cluster
 8:     for $i$ in $[|S|]$ do
 9:         $\mathscr{C}(i) = \arg\min_{1 \le k \le \kappa} D_{\mathrm{KL}}(x_i \parallel c_k)$;
10:     end for
11: end while
12: return $\mathscr{C}$.
```

resulting in the high computational complexity and clustering quality suffering from the curse of high dimensionality and data sparsity.

In order to lower the cost and enhance the recommendation qualify of GLOMA, we propose a domain-specific data-projected based clustering algorithm (DSDP), which attempts to discover the subgroups where users have similar rating tendencies on a set of items. It means the rating distribution for every user on certain item set is similar to each other. In other words, we project the data of historical ratings with length of $O(n)$ ($O(m)$) for every user (item) into the data of each rating value's frequency with length of $O(1)$ (e.g., rating value in Neflix is discrete number from 1 to 5) for every user/item. To do so, the dimension of data space is largely reduced, and thus the running time of clustering will decrease significantly as well. Meanwhile, we adopt the Kullback-Leibler (KL) divergence to measure the difference between two probability distributions, which can be defined as

$$D_{\mathrm{KL}}(x \parallel c) = \sum_z x_z \ln \frac{x_z}{c_z}, \tag{12}$$

where x and c are the data sample and cluster center, respectively. x_z (c_z) is the z-th dimensional value of x (c).

Finally, we can have the user-item subgroups after performing the Algorithm 1 on both users and items, respectively. In detailed, lines 4-6 first compute the center of the cluster which has the smallest mean distance to all samples in the cluster, then lines 8-10 assign every sample to its closest center, and finally repeat these two steps until converged. The experimental section highlights the empirically effectiveness and efficiency of the proposed clustering method. Additionally, the demonstration about that the proposed clustering method will converge to a local optimum is in the supplementary material due to limited spaces.

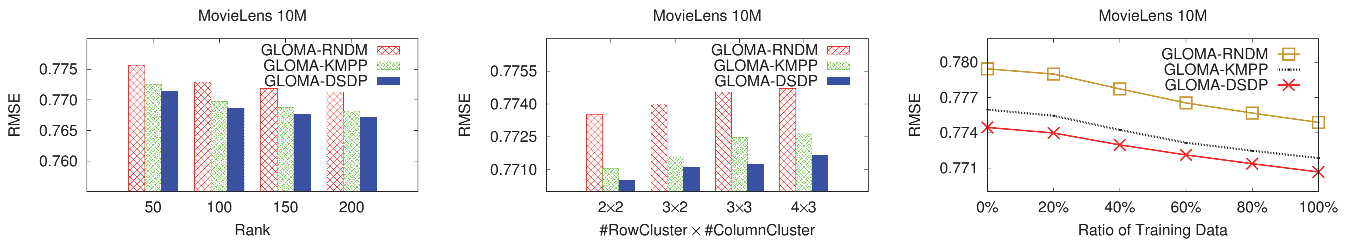


Figure 2: Effects of different clustering methods on GLOMA, while the rank varies from 50 to 200 (left), the number of row and column clusters varies in $\{2 \times 2, 3 \times 2, 3 \times 3, 4 \times 3\}$ (middle), and the ratio of training data used to build embedded model varies from 0% to 100% (right).

Table 1: Computational efficiency comparisons of the proposed domain-specific data-projected clustering method (DSDP) and K-means++ method (KMPP) in seconds on MovieLens 1M and MovieLens 10M with the numbers of row and column clusters being 2×2.

Method	MovieLens (1M)	MovieLens (10M)
DSDP	**0.471± 0.002**	**4.242 ± 0.675**
KMPP	156.942±2.093	7554.522 ± 464.806

Experimental Results

In this section, we evaluate the proposed matrix approximation CF method and domain-specific data-projected clustering method using three real-world datasets, which have been widely used for evaluating recommendation algorithms – MovieLens 1M (10^6 ratings), MovieLens 10M (10^7 ratings), and Netflix (10^8 ratings). For each dataset, we split it into train and test sets randomly by setting the ratio between train set and test set as $9 : 1$. The results are presented by averaging the results over five different random train-test splits. Recall that the root mean square error (RMSE) is adopted as the evaluation metric for recommendation accuracy which is defined in Equation (8).

For parameter setting, we use learning rate $v = 0.0008$ for gradient decent method, $\lambda = 0.06$ for all L_2-regularization coefficients, $\epsilon = 0.0001$ for gradient descent convergence threshold, and $T = 120$ for maximum number of iterations. The source codes of all experiments are publicly available [1].

Sensitivity Analysis

In this study, we varied the GLOMA hyper-parameters to better understanding its dependencies, where in default the embedded model is trained over the entire training dataset.

Impact of Clustering Methods Figure 2 analyzes the impact of different partition methods, where we name the GLOMA algorithm with random partition (GLOMA-RNDM), k-means++ method (GLOMA-KMPP), and the proposed data-projected clustering method (GLOMA-DSDP). We can see in the Figure 2 (left) the recommendation accuracy of all three methods with the number of row and column clusters being 2×2 increases as the rank increases, whereas in the Figure 2 (middle) the prediction accuracy of all three methods with setting rank = 20 decreases as the clustering size increases, this is because the data in every local model is becoming less and less as the data is divided into more clusters, resulting in data insufficiency for accurate models.

Meanwhile, Figure 2 (right) also investigates the effect of the embedded model with fixing the rank to 50, where these embedded models are trained over different ratio of the training dataset. Obviously, we can see that the prediction accuracy of all three methods increases as more data is used to train the embedded model. This is because the quality of introduced model will be better with having involving more training data such that it can better boost the performance of local models as mentioned before. Moreover, we have to note that the points at 0% means embedded model is randomly established, and even in such scenario all GLOMA algorithms can obtain the RMSE less than 0.7800, which actually is still better than the RSVD method. This is because when training every local model, the embedded model will be updated individually, such that a weak global model can be learnt, although the quality is relatively low due to lack of adequately training.

In all above three conditions, we have to note that GLOMA-DSDP and GLOMA-KMPP always outperform GLOMA-RNDM, which indicates the meaningful user-item subgroups can definitely improve the recommendation quality. Furthermore, GLOMA-DSDP is the best one of all, which demonstrates the proposed DSDP clustering method can effectively improve the model performance. Additionally, Table 1 compares the running time of the proposed DSDP and KMPP method, it can be clear to see that the DSDP method is $300X$ - $1700X$ faster than KMPP method and this advantage in running time will become larger in larger rating matrix due to the reduction in dimensionality.

Impact of Latent Factors Figure 3 further studies the contribution of user and item features in embedded model by modifying the parameters π_1 and π_2 defined in Equation (6) and (7). Notably, the point at $(0, 0)$ represents the typical clustering-based CF method, as we can see it does not perform well. Moreover, the points with $\pi_1 = 0$ means that the constraint of Equation (6) is disabled and only user features $\tilde{U}$ of embedded model can work effectively, and similarly

[1]https://github.com/ldscc/StableMA.git.

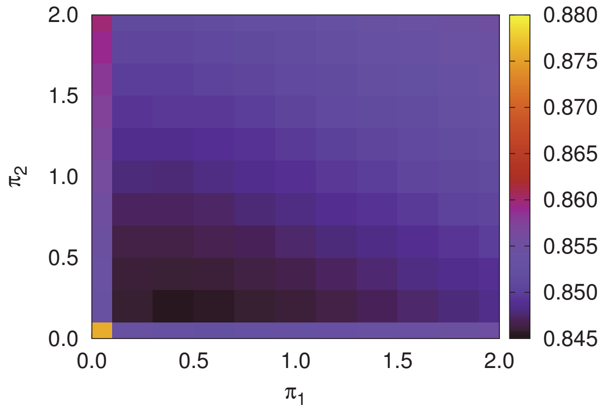


Figure 3: Effects of π_1 and π_2 controlling the contribution of embedded model on GLOMA with rank = 10, while both π_1 and π_2 range in $[0.0, 2.0]$ on MovieLens 1M.

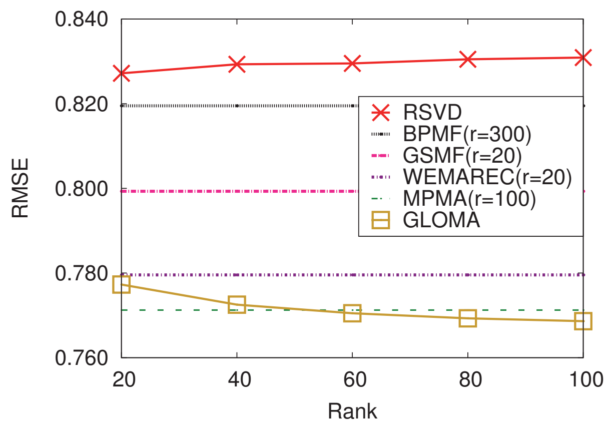


Figure 4: Effects of latent factors on GLOMA with the numbers of row and column clusters being 2×2, while the rank varies in $[20, 100]$ on MovieLens 10M.

points with $\pi_2 = 0$ means that only item features $\tilde{V}$ are actually able to function. Evidently, we can see the RMSEs in both situations are significantly higher than others, which proves that introducing both the user features and item features from the model containing overall structures among all users and items can definitely improve the model performance. In additional, the optimal performance is achieved at nearly $(0.5, 0.3)$, where both π_1 and π_2 are less than 1. It really makes sense since latent factors U and V are used to make prediction such that the terms in Equation (5) should be more important. Therefore, we adopt $\pi_1 = 0.5$ and $\pi_2 = 0.3$ in the following experiments.

Figure 4 studies the impact of rank with the numbers of row and column clusters being $g \times f = 2 \times 2$. It can be seen clearly that the accuracy of RSVD gradually decreases as the rank increase from 20 to 100, whereas the accuracy of the GLOMA method increases. The reason is that the GLOMA methods attempt to capture more information than RSVD method does, such that more latent factors are required. In additional, GLOMA outperform the other five state-of-the-art CF methods with parameter setting in their original papers, RSVD (Paterek 2007), BPMF (Salakhutdinov and Mnih 2008), GSMF (Yuan et al. 2014), WEMAREC (Chen et al. 2015), MPMA (Chen et al. 2016).

Performance Comparison

In this section, we compare the performance of GLOMA ($r = 200$) with five fore-mentioned baselines on MovieLens 10M and Netflix datasets, and moreover we re-tune the parameters for some methods to achieve better accuracy than using the default setting in their original papers, where RSVD ($r = 50$) (Paterek 2007), BPMF ($r = 300$) (Salakhutdinov and Mnih 2008), GSMF ($r = 20$) (Yuan et al. 2014), WEMAREC ($r = 100$) (Chen et al. 2015) and MPMA ($r = 200$) (Chen et al. 2016). Notably, RSVD and BPMF are standard matrix approximation methods, and WEMAREC is clustering-based ensemble method which has been shown to be more accurate than single methods due to better generalization performance. Particularly, we emphasize the comparison among GSMF, MPMA and GLOMA, because GSMF and MPMA are the latest work related to the proposed GLOMA, all of which attempts to capture various associations within user-item subgroups. As shown in Table 2, the GLOMA method significantly outperforms all five compared methods on both two datasets. This confirms that GLOMA can indeed achieve better performance than both state-of-the-art single methods and ensemble methods.

Table 2: Recommendation accuracy comparison of the proposed GLOMA method and the other five state-of-the-art methods. Bold faces mean that the method performs statistically significantly better in the setting, at the level of 95% confidence level.

	MovieLens (10M)	Netflix
RSVD	0.8271 ± 0.0009	0.8534 ± 0.0001
BPMF	0.8195 ± 0.0006	0.8420 ± 0.0003
GSMF	0.8012 ± 0.0011	0.8420 ± 0.0006
WEMAREC	0.7734 ± 0.0003	0.8098 ± 0.0009
MPMA	0.7702 ± 0.0004	0.8083 ± 0.0006
GLOMA	$\mathbf{0.7672 \pm 0.0001}$	$\mathbf{0.8011 \pm 0.0003}$

Conclusion

Standard matrix approximation based collaborative filtering methods have a major drawback that they perform poorly at detecting strong associations among a small set of closely related items. In order to address this issue, recent work adopt ensemble methods or multi-task feature learning techniques to learn the localized relations in order to produce accurate recommendations. In this paper, we develop an extension of clustering-based MA method, where a previous-trained standard MA model is introduced to help to train the local models such that the GLOMA model can unify global latent factors and local latent factors of users and items to improve recommendation accuracy. Experimental study on two real-world datasets demonstrates that the proposed GLOMA method can outperform five state-of-the-art MA-based collaborative filtering methods in recommendation accuracy, and also the proposed DSDP clustering method can efficiently discover the meaningful user-item subgroups and effectively improve the performance of GLOMA model.

Acknowledgement

This work was supported in part by the National Natural Science Foundation of China (61233016, 61602176, 61672231), China Postdoctoral Science Foundation Funded Project (2016M590337), and the National Science Foundation of USA (1251257, 1334351, 1442971).

References

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

Billsus, D., and Pazzani, M. J. 1998. Learning collaborative information filters. In *Proceedings of The 15th International Conference on Machine Learning (ICML '98)*, 46–54.

Breese, J. S.; Heckerman, D.; and Kadie, C. 1998. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial nItelligence (UAI'98)*, 43–52. Morgan Kaufmann Publishers Inc.

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

Chen, C.; Li, D.; Zhao, Y.; Lv, Q.; and Shang, L. 2015. WEMAREC: Accurate and scalable recommendation through weighted and ensemble matrix approximation. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*, 303–312.

Chen, C.; Li, D.; Lv, Q.; Yan, J.; Chu, S. M.; and Shang, L. 2016. Mpma: Mixture probabilistic matrix approximation for collaborative filtering. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI '16)*, 1382–1388.

Evgeniou, A., and Pontil, M. 2007. Multi-task feature learning. In *Advances in Neural Information Processing Systems*, 41–48.

Herlocker, J. L.; Konstan, J. A.; Borchers, A.; and Riedl, J. 1999. An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '99)*, 230–237.

Koren, Y. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD '08)*, 426–434. ACM.

Lee, J.; Kim, S.; Lebanon, G.; and Singer, Y. 2013. Local low-rank matrix approximation. In *Proceedings of The 30th International Conference on Machine Learning (ICML '13)*, 82–90.

Li, D.; Chen, C.; Lv, Q.; Yan, J.; Shang, L.; and Chu, S. 2016. Low-rank matrix approximation with stability. In *Proceedings of The 33rd International Conference on Machine Learning (ICML '16)*, 295–303.

Mackey, L. W.; Jordan, M. I.; and Talwalkar, A. 2011. Divide-and-conquer matrix factorization. In *Advances in Neural Information Processing Systems*, 1134–1142.

Mnih, A., and Salakhutdinov, R. 2007. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, 1257–1264.

Paterek, A. 2007. Improving regularized singular value decomposition for collaborative filtering. In *KDD CUP'07*, 5–8.

Salakhutdinov, R., and Mnih, A. 2008. Bayesian probabilistic matrix factorization using markov chain monte carlo. In *Proceedings of The 25th International Conference on Machine Learning (ICML '08)*, 880–887.

Sarwar, B.; Karypis, G.; Konstan, J.; and Riedl, J. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web (WWW'01)*, 285–295.

Singh, A. P., and Gordon, G. J. 2008. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD '08)*, 650–658.

Srebro, N., and Jaakkola, T. 2003. Weighted low-rank approximations. In *Proceedings of The 20th International Conference on Machine Learning (ICML '03)*, 720–727.

Su, X., and Khoshgoftaar, T. M. 2009. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* 2009:4:2–4:2.

Yuan, T.; Cheng, J.; Zhang, X.; Qiu, S.; and Lu, H. 2014. Recommendation by mining multiple user behaviors with group sparsity. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI '14)*, 222–228.

Zhang, Y.; Zhang, M.; Liu, Y.; and Ma, S. 2013. Improve collaborative filtering through bordered block diagonal form matrices. In *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '13)*, 313–322.