Electronic Commerce Research and Applications 26 (2017) 109–118

Contents lists available at ScienceDirect

Electronic Commerce Research and Applications

journal homepage: www.elsevier.com/locate/ecra

Recommendation in feature space sphere

Zhi-Lin Zhao [a], Chang-Dong Wang [a,*], Yuan-Yu Wan [b], Jian-Huang Lai [a]

[a] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
[b] Department of Computer Science and Technology, Nanjing University, Nanjing, China

ARTICLE INFO

Article history:
Received 2 July 2017
Received in revised form 21 September 2017
Accepted 27 October 2017
Available online 28 October 2017

Keywords:
Recommendation algorithms
Matrix factorization
Feature space

ABSTRACT

Recently, recommendation algorithms have been widely used in many e-commerce platforms to recommend items to users on the basis of their preferences to improve selling efficiency. Matrix factorization methods which extract latent features of users and items by decomposing the rating matrix have achieved success in rating prediction. But almost all of these algorithms are designed to fit the rating matrix directly to get the latent features and ignore the user-item relationship in feature space. To this end, in this paper, we propose a recommendation in feature space sphere (RFSS) which takes into account the relationship between users and items in feature space. Different from the conventional latent feature based recommendation algorithms, the proposed algorithm supposes that if a user likes an item, the user is close to the item in feature space. Meanwhile, the closer a user and an item are in feature space, the higher the predicted rating will be. And an adaptive user-dependent coefficient is introduced to map the user-item distances to the predicted ratings. Extensive experiments on four real-world datasets have been conducted, the results of which show that our proposed method outperforms the state-of-the-art recommendation algorithms.

1. Introduction

Due to the rapid growth of online markets/services, an increasing amount of merchandises/services can be sold/provided in these platforms nowadays. This makes it difficult for users to find something interesting or useful in a short time. As a consequence, recommendation algorithm emerges as required and has been widely used in many online markets/services like Amazon (Linden et al., 2003), YouTube (Davidson et al., 2010), Twitter (Elmongui et al., 2015), Tmall (Zhong et al., 2015) and Yahoo! (Koenigstein et al., 2011), to improve selling efficiency and enhance user experience. The function of recommendation algorithm is to recommend items to target users they are most likely interested in based on the huge amounts of data about user behaviour. Recommendation algorithm usually predicts the ratings to non-purchased items and presents the recommendation lists to the target users in the descending order of the predicted ratings (Guo et al., 2014). On the whole, the traditional recommendation algorithms can be classified into three types: collaborative filtering, content-based recommendation and hybrid recommendation (Jannach et al., 2013). Among them, collaborative filtering, one of the most successful technologies in personalized recommendation, can be separated into memory-based methods and model-based methods. And the basic idea of collaborative filtering is that a user prefers the items liked by the users with similar interest. In particular, matrix factorization is one of the most common model-based collaborative filtering algorithms.

Over the past decade, matrix factorization has attracted an increasing amount of attention. Matrix factorization technique usually learns the latent features of both users and items from the user-item rating matrix, and then predicts the ratings to non-purchased items according to user and item latent features. The most remarkable matrix factorization algorithm is probabilistic matrix factorization (PMF) (Salakhutdinov and Mnih, 2007). Another remarkable method is Non-negative Matrix Factorization (NMF) (Lee and Seung, 2000), where the constraint that all the features should be positive is applied. A sparse linear method (SLIM) (Ning and Karypis, 2011) uses sparse aggregation coefficient to make Top-N recommendation with high quality and efficiency concurrently.

Although matrix factorization methods have achieved remarkable success, there are some deficiencies. Matrix factorization methods focus on users and items in feature space and get the latent feature vectors of users and items by decomposing the rating matrix. But almost all of these algorithms are designed to fit the

* Corresponding author.
E-mail addresses: zhaozhl7@mail2.sysu.edu.cn (Z.-L. Zhao), changdongwang@hotmail.com (C.-D. Wang), wanyy@smail.nju.edu.cn (Y.-Y. Wan), stsljh@mail.sysu.edu.cn (J.-H. Lai).

https://doi.org/10.1016/j.elerap.2017.10.007
1567-4223/

matrix directly to extract latent features and have not considered the user-item relationship in feature space. However, this relationship is useful in recommendation. This is because, different users will give different ratings to the same item and we can infer whether the user likes the item according to the rating. If a user gives a high rating to an item since the user likes the item, the user should be close to the item in feature space and vice versa. Another deficiency is that, although most of existing recommendation algorithms use inner product to fit real rating, this method lacks interpretation about why inner product can be used to predict rating. So, we develop a new prediction method based on similarity, which can explain that the predicted rating is associated with the relationship between user and item in feature space.

To address the above issues, we present a recommendation in feature space sphere (RFSS). This algorithm considers the relationship between users and items in feature space which is measured by Euclidean distance with rating as weight. If a user likes an item, the weight will be large and the user will be close to the item in feature space. And if a user is close to an item in feature space, the predicted rating will be high. Additionally, a user-dependent coefficient which is self-adaptive is used to map the relationship between user and item in feature space.

The contributions are summarized as follows:

1. A regularization term that measures the user-item relationship in feature space by Euclidean distance will be considered in the objective function.
2. An adaptive user-dependent coefficient is introduced to map the cosine-based similarities between users and items to the predicted ratings.
3. Complexity and convergence analysis is conducted to show the convergence property of the proposed method.
4. We conduct experiments on four real world datasets and the results show that the proposed RFSS algorithm outperforms the state-of-the-art recommendation algorithms.

2. Related work

Many efforts have been made in matrix factorization algorithms to improve the performance. Bayesian probabilistic matrix factorization (BPMF) (Ruslan and Andriy, 2008) uses bayesian treatment on the probabilistic matrix factorization to introduce Gaussian-Wishart priors on the hyperparameters of the user and item feature vectors. For improving singular value decomposition (SVD), the biases of users and items are integrated into SVD so as to better fit ratings (Paterek, 2007). In Wen et al. (2014), Cosine Matrix Factorization (CosMF) utilizes cosine similarity to replace inner product for sparse users and items to address the sparsity problem without auxiliary data. But it just considers the angle between two latent feature vectors and ignores their lengths. As an improved version, the expected risk minimized matrix approximation method (ERBMMA) (Li et al., 2017) uses expected risk to achieve better tradeoff between optimization error and generalization error. Chen et al. proposed a cross-domain recommendation algorithm (Chen et al., 2013) which uses PARAFAC tensor decomposition to extract the knowledge from the auxiliary domain and makes use of the knowledge in the target domain to increase user acceptance rate in recommendation lists. Kang et al. proposed a matrix factorization algorithm (Kang et al., 2016) that fills user-item matrix based on the low-rank assumption and keeps the original information at the same time for Top-N recommendation. But the above algorithms do not take the relationships between latent feature vectors into consideration.

Recently, the relationship about users and items in feature space has been considered to improve the performance of matrix factorization recommendation algorithm. Matrix factorization to asymmetric user similarities (MF-AMSD) (Pirasteh et al., 2015) gets the user features by decomposing the asymmetric user similarity matrix, so the similar users are also similar in feature space. Although these user features are used to predict ratings, the processes of feature extraction and rating prediction are independent which will cause error propagation. Recommender Systems with Social Regularization (SR) (Ma et al., 2011) supposes that a user should be close to his trusted users in feature space while the algorithm needs the data in trust network which may not be suitable for universal cases. On the contrary, the algorithm proposed by Paterek (2007), considering the item relationship in feature space, uses the item-item similarity learned as a product of two low-rank vectors to make a rating prediction. But the algorithm only considers item relationship and ignores user relationship in feature space. Following Paterek, Koren proposed a method (Koren, 2008) which combines matrix factorization and the traditional neighborhood based model to learn the latent features of users and items simultaneously, but the drawback is that the relationship between users and items in feature space has not yet been considered. Sparse covariance matrix factorization (SCMF) (Shi et al., 2013) uses sparse covariance prior to find the correlation between latent features. The algorithm connects users and items by placing the same prior to latent feature vectors which however cannot ensure a user will be close to the item he prefers in feature space.

To address the above issues, we propose a recommendation in feature space sphere (RFSS) which can improve the quality of recommendation by considering the user-item relationship in feature space.

3. The proposed algorithm

In recommendation algorithms, a user-item rating matrix $R = [r_{ij}]_{m\times n}$ is used to represent the rating relation between m users and n items. Each entry r_{ij} denotes the rating of user i to item j within a certain numerical interval $[R_{min}, R_{max}]$ which will vary in different datasets and if user i does not rate item $j, r_{ij} = 0$. I_{ij} is the indicator function that is equal to 1 if user i has rated item j or 0 otherwise. In the matrix factorization recommendation algorithms, α_i and β_j are d-dimensional vectors representing the latent features of user i and item j respectively.

3.1. Objective function

Different from the conventional latent factor based recommendation algorithms (Gao et al., 2013), in our proposed algorithm, we suppose that all the latent feature vectors are laying on the unit sphere surface in the feature space, so $\alpha_i\alpha_i^T = 1$ for $i = 1, \ldots, m$ and $\beta_j\beta_j^T = 1$ for $j = 1, \ldots, n$. As we will see below, the advantage of unit constraint is that the similarity between user and item in feature space can be calculated easily. Besides, the Euclidean distance can be confined to a certain range which can avoid some extreme conditions.

Therefore, the user latent feature vectors and item latent feature vectors are in the same feature space. All the items have their own features and the features of users can been represented by the features of items they have rated. In the feature space, if user i likes item j, the user is close to the item, i.e. their latent feature vectors are similar. So, Euclidean distance is suitable to measure user-item relationship. The kind of user-item relationship can be reconstructed by minimizing the term $\sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij}\|\alpha_i - \beta_j\|_2^2 I_{ij}$ under the constraint of unit sphere surface representation, in which the rating r_{ij} can be viewed as a weight and *the higher the rating is, the closer user i and item j will be enforced to be.* Besides, this term can be viewed as a regularization term when learning the latent feature vectors α_i and β_j. The value of $\|\alpha_i - \beta_j\|_2^2$ is bounded, according to Cauchy-Schwartz and Minkowski inequality:

$$0 \leqslant r_{ij}\|\alpha_i - \beta_j\|_2^2 = r_{ij} \mid (\alpha_i - \beta_j)(\alpha_i - \beta_j)^T \mid \leqslant r_{ij}\|\alpha_i - \beta_j\|_2\|\alpha_i - \beta_j\|_2 \leqslant r_{ij}(\|\alpha_i\|_2 + \|\beta_j\|_2)(\|\alpha_i\|_2 + \|\beta_j\|_2) = 4R_{max} \quad (1)$$

But our goal is to predict ratings to non-purchased items, we suppose *the closer the latent feature vectors of user i and item j are, the higher the predicted rating will be.* Although inner product is commonly used by matrix factorization algorithms to predict ratings, it lacks interpretation. So, we propose a cosine-based similarity method, and the similar between user i and item j is:

$$s_{ij} = \frac{\alpha_i\beta_j^T}{\|\alpha_i\|_2^2\|\beta_j\|_2^2} = \alpha_i\beta_j^T. \quad (2)$$

There are two L2-norm terms in the denominator of the cosine function, which makes the problem difficult to solve. That is the reason why we suppose features lie on the surface of unit sphere. Because the situation user i is close to item j in the unit sphere of the feature space means the user likes the item, the value of the predicted rating of the user to the item tends to be larger as they get closer. Another advantage of cosine-based similarity is that upper bound and lower bound are certain and would not cause infinitely large or infinitely small predicted ratings.

So, the Euclidean distance is used to model the relationship between users and items in feature space while the cosine operation is used to infer ratings according to latent features. Traditional matrix factorization methods such as PMF (Salakhutdinov and Mnih, 2007) use inner product to predict ratings but they lack the explanation why it can be used and what it represents. And the proposed algorithm predicts the ratings according to the cosine distances between users and items in feature space.

However, the value of rating r_{ij} can not be determined by only the similarity s_{ij}, since the range of similarity is $[0, 1]$ which might be different from that of rating (e.g. in case of rating range of $[1, 5]$). Moreover, for different users, even the same values of similarity to the same item may reflect various levels of affection these users to the item which will be discussed later. To this end, we get the predicted rating p_{ij} of user i to item j by the linear function:

$$p_{ij} = l_i s_{ij} = l_i\alpha_i\beta_j^T, \quad (3)$$

where the adaptive user-dependent coefficient l_i varies from one user to another. We just consider user-dependent coefficient because users have their own idea and items are objective, so item-dependent coefficient is unnecessary. The views from users to the same item will vary but the properties of items will not change. So, we map similarity to rating according to humanity rather than using item scaling term. If a user only likes and has purchased one kind of items (i.e. the items are similar), the user is close to all the items he has rated in the unit sphere surface of the feature space and all the similarities will be large. So the coefficient l_i will be small for the user. On the contrary, if the interest of a user is wide (i.e. these purchased items distribute widely in different categories), he is relatively far from all the rated items in the unit sphere surface of the feature space because all the purchased items are far from each other and the user will lie in the center of these items approximately by definition. So, the coefficient l_i will be large for the user. From another point of view, some users would like to give high ratings to items while other users low, so the values of l_i will be large for the former and small for the latter. The more complicated and common scenario is that some users give high rating to some items and give low rating to the other items, and in this case the magnitude of the coefficient l_i is difficult to be determined by intuition. In general, the values of the coefficient l_i should be diverse for different users and should be learned from data automatically. In order to get a good prediction, the prediction error of the term $\sum_{i=1}^{m}\sum_{j=1}^{n}\|r_{ij} - l_i\alpha_i\beta_j^T\|_2^2 I_{ij}$ should be reduced. The user and item latent feature vectors in the term of prediction error are impacted by the user-item relationship.

We should notice that the proposed RFSS algorithm aims to find suitable positions for all users and items in the unit sphere feature space according to the rating records, and predict ratings according to the relationships in the feature space. So the meanings of the two latent feature vectors α_i and β_j are the same in our algorithm. The value of a feature in the d-dimensional space represents the preference of a user or an item. As mentioned before, the drawback of PMF (Salakhutdinov and Mnih, 2007) is that there is no explanation why inner product can be used to predict ratings. So we propose a novel algorithm from a different perspective, which can explain the meaning of measuring distances and predicting ratings in the feature space. Although Eq. (3) is similar to the method used in PMF, the meaning is quite different. The proposed method deduces Eq. (3) from the spatial relationship and cosine-based similarity method. Besides, there are a user-dependent coefficient and a spatial constraint.

Because $\alpha_i\alpha_i^T = 1$ and $\beta_j\beta_j^T = 1$ are not convex set, by relaxing these constraints into convex set $\alpha_i\alpha_i^T \leqslant 1$ and $\beta_j\beta_j^T \leqslant 1$, Eq. (3) can be taken as an approximate cosine-based similarity.To better understand the constraint and the whole algorithm, an example is shown in Fig. 1 which describes the spatial relationship of α_i and β_j. In order to visualize the data element, the dimensionality d is set to be 2 in this example. If we strictly restrict the vectors

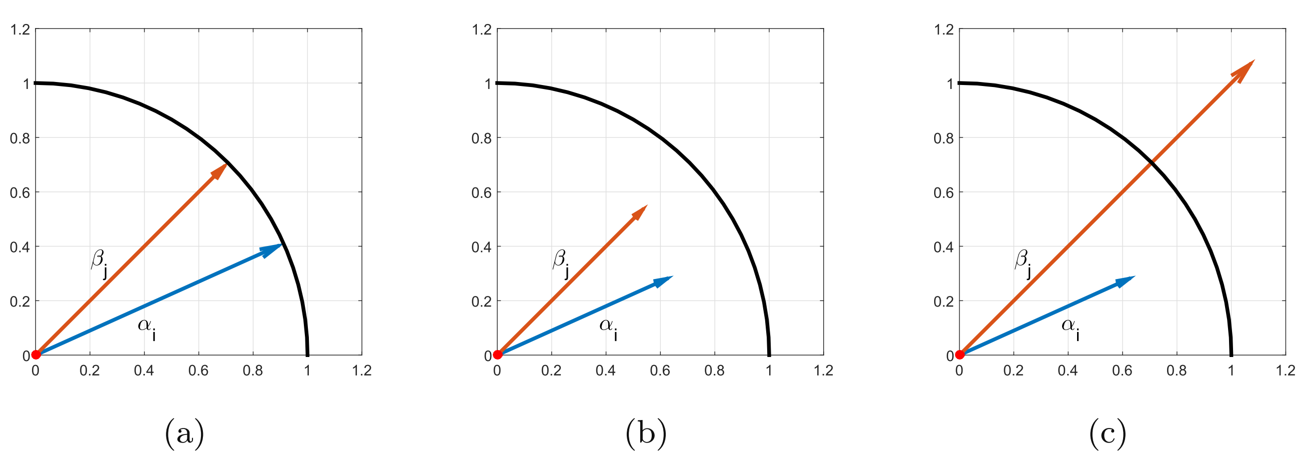

Fig. 1. Illustration of the RFSS model in the two-dimensional case.

have to reach the surface of the unit sphere feature space as shown in Fig. 1(a), the expression abilities of vectors will be very limited. If we relax the constraint to the inside as well as the surface of the sphere as shown in Fig. 1(b), the domain of vectors will be convex which ensures the convergence of the function (Boyd and Vandenberghe, 2004). If we remove the constraint, some poor predictions will occur as shown in Fig. 1(c). Without this restricted condition, the predicted rating will be high even the user and the item are far from each other because the predicting method is based on the approximate cosine-based similarity which does not consider the length of vectors.

As the assumption of the proposed algorithm, if a user likes an item, the user should be close to the item in feature space, and the predicted value should be high. That is also why cosine similarity is used to predict ratings. So the relaxed constraint can make sure that the approximate cosine-based similarity can provide rational predictions because the length of vectors is in the range of $[0, 1]$. The term $\sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij}\|\alpha_i - \beta_j\|_2^2 I_{ij}$ may also mitigate those poor predictions to some extent but it focuses more on measuring the relationships of users and items in the feature space instead of generating the sound rating predictions of the approximate cosine-based similarity.

Considering the user-item relationship, we can get the objective function:

$$\begin{aligned} \min \quad & \mathcal{L} = \tfrac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n}\|r_{ij} - l_i\alpha_i\beta_j^T\|^2 I_{ij} + \tfrac{\lambda}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij}\|\alpha_i - \beta_j\|_2^2 I_{ij}, \\ \text{s.t.} \quad & \alpha_i\alpha_i^T \leqslant 1, i = 1\ldots m, \\ & \beta_j\beta_j^T \leqslant 1, j = 1\ldots n, \end{aligned} \tag{4}$$

where λ is a regularization coefficient which is used to adjust the influence of the second term.

3.2. Optimization

The constraint terms can be moved into the objective function and the problem becomes:

$$\min \quad \mathcal{L}_c = \mathcal{L} + \sum_{i=1}^{m}\mathcal{G}(\alpha_i) + \sum_{j=1}^{n}\mathcal{G}(\beta_j), \tag{5}$$

where $\mathcal{G}$ is an indicator function:

$$\mathcal{G}(\theta) = \begin{cases} 0, & \text{if } \theta\theta^T \leqslant 1 \\ 1, & \text{otherwise} \end{cases}. \tag{6}$$

Because when we get the optimal solution of $\mathcal{L}_c$, all the indicator functions $\mathcal{G}(\alpha_i)$and $\mathcal{G}(\beta_j)$ are equal to 0 and all the constraints of variables α_i and β_j will be satisfied. But the optimization problem is still complicated. In order to split the problem $\mathcal{L}_c$ into some subproblems which can be easily resolved (optimized) in an alternate way, some new variables x_i and y_j will be introduced and this problem can be rewritten in ADMM (Alternating Direction Method of Multipliers) (Boyd et al., 2011) form as follows:

$$\begin{aligned} & \min \mathcal{L}_g = \mathcal{L} + \sum_{i=1}^{m}\mathcal{G}(x_i) + \sum_{j=1}^{n}\mathcal{G}(y_j), \\ & \text{s.t.}\ \alpha_i - x_i = 0,\ i = 1\ldots m, \\ & \quad \beta_j - y_j = 0,\ j = 1\ldots n. \end{aligned} \tag{7}$$

As in the method of multipliers, we form the augmented Lagrangian (using the scaled dual variable) (Hestenes, 1969; Powell, 1969):

$$\begin{aligned} \mathcal{L}_\rho = \; & \tfrac{1}{2} um_{i=1}^{m}\sum_{j=1}^{n}\|r_{ij} - l_i\alpha_i\beta_j^T\|_2^2 I_{ij} + \tfrac{\lambda}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij}\|\alpha_i - \beta_j\|_2^2 I_{ij} \\ & + \sum_{i=1}^{m}\mathcal{G}(x_i) + \sum_{i=1}^{m}(\rho_{\alpha_i,x_i}/2)\|\alpha_i - x_i + u_i\|_2^2 \\ & + \sum_{j=1}^{n}\mathcal{G}(y_j) + \sum_{j=1}^{n}(\rho_{\beta_j,y_j}/2)\|\beta_j - y_j + v_j\|_2^2, \end{aligned} \tag{8}$$

where u_i and v_j are scaled dual variable, $\rho_{\alpha_i,x_i} > 0$ and $\rho_{\beta_j,y_j} > 0$ are penalty parameters. In order to update all the parameters in the $(k+1)$th iteration, we can express ADMM as

$$\begin{aligned} & \alpha_i^{k+1} = \arg\min_{\alpha_i}(\mathcal{L}(\alpha_{-i}^k, \alpha_i, \beta^k, l^k) + (\rho_{\alpha_i,x_i}/2)\|\alpha_i - x_i^k + u_i^k\|_2^2), \\ & \beta_j^{k+1} = \arg\min_{\beta_j}(\mathcal{L}(\alpha^{k+1}, \beta_{-j}^k, \beta_j, l^k) + (\rho_{\beta_j,y_j}/2)\|\beta_j - y_j^k + v_j^k\|_2^2), \\ & l_i^{k+1} = \arg\min_{l_i}\mathcal{L}(\alpha^{k+1}, \beta^{k+1}, l_{-i}^k, l_i), \\ & x_i^{k+1} = \arg\min_{x_i}\Big(\mathcal{G}(x_i) + (\rho_{\alpha_i,x_i}/2)\|\alpha_i^{k+1} - x_i + u_i^k\|_2^2\Big) = \Pi_{\mathcal{G}}(\alpha_i^{k+1} + u_i^k), \\ & y_j^{k+1} = \arg\min_{y_j}\Big(\mathcal{G}(y_j) + (\rho_{\beta_j,y_j}/2)\|\beta_j^{k+1} - y_j + v_j^k\|_2^2\Big) = \Pi_{\mathcal{G}}(\beta_j^{k+1} + v_j^k), \\ & u_i^{k+1} = u_i^k + \alpha_i^{k+1} - x_i^{k+1}, \\ & v_j^{k+1} = v_j^k + \beta_j^{k+1} - y_j^{k+1}, \end{aligned} \tag{9}$$

where $\Pi_{\mathcal{G}}$denotes the projection onto $x_i x_i^T \leqslant 1$ (or $y_j y_j^T \leqslant 1$).

We can update the parameters α_i and β_j by Gradient Descent method and the gradients have the following forms:

$$\begin{aligned} \nabla\alpha_i^k = & -\sum_{j=1}^{n}(r_{ij} - l_i\alpha_i^k(\beta_j^k)^T) l_i\beta_j^k I_{ij} + \lambda\sum_{j=1}^{n} r_{ij}(\alpha_i^k - \beta_j^k) I_{ij} \\ & + (\rho_{\alpha_i,x_i}/2)(\alpha_i^k - x_i^k + u_i^k) = \sum_{j=1}^{n}((l_i^2\alpha_i^k(\beta_j^k)^T - (l_i + \lambda)r_{ij})\beta_j^k \\ & + \lambda r_{ij}\alpha_i^k) I_{ij} + (\rho_{\alpha_i,x_i}/2)(\alpha_i^k - x_i^k + u_i^k), \\ \nabla\beta_j^k = & -\sum_{i=1}^{m}(r_{ij} - l_i\alpha_i^{k+1}(\beta_j^k)^T) l_i\alpha_i^k I_{ij} - \lambda\sum_{i=1}^{m} r_{ij}(\alpha_i^{k+1} - \beta_j^k) I_{ij} \\ & + (\rho_{\beta_j,y_j}/2)(\beta_j^k - y_j^k + v_j^k) = \sum_{i=1}^{m}((l_i^2\alpha_i^{k+1}(\beta_j^k)^T - (l_i + \lambda)r_{ij})\alpha_i^{k+1} \\ & + \lambda r_{ij}\beta_j^k) I_{ij} + (\rho_{\beta_j,y_j}/2)(\beta_j^k - y_j^k + v_j^k). \end{aligned} \tag{10}$$

As pointed out in Eckstein and Bertsekas (1992), ADMM allows us to solve the minimizations only approximately at first, and then more accurately as the iterations progress. So, the parameters α_i and β_j are not necessary to reach the best when the other parameters are fixed in each iteration and Gradient Descent method will be used to get new α_i and β_j which are better than those of the last iteration:

$$\begin{aligned} \alpha_i^{k+1} & = \alpha_i^k - \mu\nabla\alpha_i^k, \\ \beta_i^{k+1} & = \beta_i^k - \mu\nabla\beta_j^k, \end{aligned} \tag{11}$$

where μ is the learning rate.

Since l_i^{k+1} minimizes $\mathcal{L}(\alpha^{k+1}, \beta^{k+1}, l_{-i}^k, l_i)$, the gradient of l_i is:

$$\nabla l_i^k = -\sum_{j=1}^{n}\Big(r_{ij} - l_i\alpha_i^{k+1}(\beta_j^{k+1})^T\Big)\alpha_i^{k+1}(\beta_j^{k+1})^T I_{ij}. \tag{12}$$

The formula of l_i^{k+1} can be obtained by setting $\nabla l_i^k = 0$:

$$l_i^{k+1} = \frac{\sum_{j=1}^{n} r_{ij}\alpha_i^{k+1}\left(\beta_j^{k+1}\right)^T I_{ij}}{\sum_{j=1}^{n}\left(\alpha_i^{k+1}\left(\beta_j^{k+1}\right)^T\right)^2 I_{ij}}. \tag{13}$$

In order to get the update method of x_i, we should get the projection of $\vartheta = \alpha_i^{k+1} + u_i^k$ within the unit sphere space $x_i x_i^T \leqslant 1$. The projection problem can be transformed into an equivalent constrained minimization problems when $\alpha_i^{k+1} + u_i^k > 1$:

$$\begin{aligned} \min \quad & \|x_i - \vartheta\|_2^2, \\ \text{s.t.} \quad & x_i x_i^T \leqslant 1. \end{aligned} \tag{14}$$

The Lagrangian function of the above objective is:

$$L(x_i, v) = \|x_i - \vartheta\|_2^2 + v(x_i x_i^T - 1), \tag{15}$$

where v is a Lagrange multiplier. When

$$x_i = \frac{\vartheta}{1+v}, \tag{16}$$

the Lagrangian function $L(x_i, v)$ gets the minimal value. So the Lagrange dual function is:

$$\begin{aligned} g(v) \quad & = \min_{x_i} L(x_i, v) = \|(\tfrac{1}{1+v} - 1)\vartheta\|_2^2 + v\left(\tfrac{1}{(1+v)^2}\vartheta\vartheta^T - 1\right) \\ & = \tfrac{v}{1+v}\vartheta\vartheta^T - v, \end{aligned} \tag{17}$$

where the Lagrange multiplier v can be obtained by:

$$v = \arg\max_{v} g(v) = \sqrt{\vartheta\vartheta^T} - 1. \tag{18}$$

So the formula of x_i in the $(k+1)^{th}$ iteration is:

$$x_i^{k+1} = \Pi_{\mathcal{G}}(\vartheta) = \begin{cases} \vartheta, & \text{if } \vartheta\vartheta^T \leqslant 1 \\ \frac{\vartheta}{\sqrt{\vartheta\vartheta^T}}, & \text{otherwise} \end{cases}, \tag{19}$$

and the update method of y_j can be derived in the same way.

The primal and dual residuals of α_i and x_i in the kth iteration are $w_{\alpha_i,x_i}^k = \alpha_i - x_i$ and $s_{x_i}^k = \rho_{\alpha_i,x_i}^k(x_i^{k-1} - x_i^k)$ respectively. And the two terms of β_j and y_j are $w_{\beta_j,y_j}^k = \beta_j - y_j$ and $s_{y_j}^k = \rho_{\alpha_i,x_i}^k(y_j^{k-1} - y_j^k)$. The penalty parameters ρ_{α_i,x_i}^k can be updated in each iteration by He et al. (2000); Wang and Liao, 2001:

$$\rho_{\alpha_i,x_i}^{k+1} = \begin{cases} \tau^{incr}\rho_{\alpha_i,x_i}^k, & \text{if } \|w_{\alpha_i,x_i}^k\|_2 > \phi\|s_{x_i}^k\|_2 \\ \rho_{\alpha_i,x_i}^k/\tau^{decr}, & \text{if } \|s_{x_i}^k\|_2 > \phi\|w_{\alpha_i,x_i}^k\|_2 \\ \rho_{\alpha_i,x_i}^k, & \text{otherwise.} \end{cases}, \tag{20}$$

where the typical choices of the parameters are $\tau^{incr} = \tau^{decr} = 2$ and $\phi = 10$. The update of ρ_{β_j,y_j} can be derived similarly.

3.3. Rating adjustment

A local minimum of the objective function will be found when the iteration converges and the predicted rating p_{ij} of user i to a non-purchased item j can be predicted by Eq. (3). And we must ensure that all predicted ratings p_{ij} are in an interval $[R_{min}, R_{max}]$. When the predicted rating goes beyond the range, some adjustments must be applied:

$$p_{ij} = \begin{cases} R_{max}, & \text{if } p_{ij} > R_{max} \\ R_{min}, & \text{if } p_{ij} < R_{min} \\ p_{ij}, & \text{otherwise} \end{cases}. \tag{21}$$

3.4. Overview

For clarity, the proposed algorithm is summarized in Algorithm 1. Since it is difficult to directly optimize α_i and β_j from the complex objective function Eq. (4), we divide α_i into α_i and x_i, and β_j into β_j and y_j based on the divide and conquer idea of ADMM. In the proposed algorithm, α_i and β_j should satisfy the unit sphere constraint. Due to ADMM, it is unnecessary for us to restrict α_i and β_j directly. As shown in Algorithm 1, x_i and y_j will be projected to the unit sphere feature space in each iteration, the theoretical guarantee of which has been proved in the project algorithm (Parikh and Boyd, 2014). And the theory of ADMM (Boyd et al., 2011) can make sure that $\alpha_i = x_i$ and $\beta_j = y_j$ when the objective function converges, so α_i and β_j will satisfy the spatial constraint in the end.

Algorithm 1 RFSS

Input: rating matrix R, regularization coefficient λ, learning rate μ
Initialize: randomly initialize $\alpha_i, \beta_j, l_i, x_i, y_j, u_i, v_j, \rho_{\alpha_i,x_i}, \rho_{\beta_j,y_j}$
while not converge do
  Compute gradients $\nabla\alpha_i$ and $\nabla\beta_j$ according to Eq. 10.
  $\alpha_i = \alpha_i - \mu\nabla\alpha_i$.
  $\beta_j = \beta_j - \mu\nabla\beta_j$.
  $l_i = \frac{\sum_{j=1}^{n} r_{ij}\alpha_i(\beta_j)^T I_{ij}}{\sum_{j=1}^{n}(\alpha_i(\beta_j)^T)^2 I_{ij}}$.
  $x_i = \Pi_{\mathcal{G}}(\alpha_i + u_i)$.
  $y_j = \Pi_{\mathcal{G}}(\beta_j + v_j)$.
  $u_i = u_i + \alpha_i - x_i$.
  $v_j = v_j + \beta_j - y_j$.
  Update ρ_{α_i,x_i} and ρ_{β_j,y_j} according to Eq. (20).
end while
Predict p_{ij} according to Eq. (3) and adjust it by Eq. (21).
Output: p_{ij}

4. Complexity and convergence analysis

Similar to PMF (Salakhutdinov and Mnih, 2007), the most time-consuming part is the derivation of user and item latent feature vectors. So, the time complexity is also $\mathcal{O}(tmnd)$ where t is the number of iterations.

Lemma 1. If $x = \Pi_{\mathcal{G}}(\vartheta)$, $\|x\|_2^2 \leqslant \|\vartheta\|_2^2$

proof 1. According to Eq. (22), we can get that:

$$\|x\|_2^2 = \begin{cases} \|\vartheta\|_2^2, & \text{if } \vartheta\vartheta^T \leqslant 1 \\ 1, & \text{otherwise} \end{cases} \leqslant \min\{1, \|\vartheta\|_2^2\} \leqslant \|\vartheta\|_2^2. \tag{22}$$

Lemma 2. If $\|x_i\|_2^2 \leqslant \|\vartheta_i\|_2^2$ for $i = 1, \ldots, m$, there is a variable δ making the following equation true,

$$\delta\mathbb{E}[\|x_i\|_2^2] = \mathbb{E}[\|\vartheta_i\|_2^2] \tag{23}$$

where $\delta \geqslant 1$.

Theorem 1. According to the proposed optimization method, $(\alpha^{k*}, x^{k*}, \beta^{k*}, y^{k*})$ is an optimal solution of the k^{th} iteration to the RFSS algorithm and the convergence rate is,

$$|f^k - f^*| = \max\left\{O\left(\frac{1}{2}\right), O(\rho^{k-1}\sqrt{\delta_{k-1}})\right\}, \tag{24}$$

where $f^k = \mathcal{L}_g(\alpha^{k*}, x^{k*}, \beta^{k*}, y^{k*})$is the value of the kth iteration and f^* is the optimal value of the RFSS algorithm.

proof 2.

For convenience, we suppose $\rho^k_{\alpha_i,x_i} = \rho^k_{\beta_j,y_j} = \rho^k$. Because the parameter l will get the optimal value in each iteration, so we just discuss the parameters α, x and the analyses on β, y are similar.

$$\begin{aligned}\mathcal{L}_\rho(\alpha^{(k+1)*}, x^{(k+1)*}, u^{k*}) &= \min \mathcal{L}_\rho(\alpha, x, u^{k*}) \leqslant \min_{\alpha=x} \mathcal{L}_\rho(\alpha, x, u^{k*}) \\ &= \min_{\alpha=x} \mathcal{L}_g(\alpha, x, u^{k*}) = f^*,\end{aligned} \tag{25}$$

we can get,

$$\begin{aligned}\mathcal{L}_g(\alpha^{(k+1)*}, x^{(k+1)*}) &= \mathcal{L}_\rho(\alpha^{(k+1)*}, x^{(k+1)*}, u^{k*}) - \tfrac{\rho}{2}(\sum_{i=1}^m \|u_i^{k*} + \alpha_i^{(k+1)*} \\ &\quad - x_i^{(k+1)*}\|_2^2) \\ &\leqslant f^* - \tfrac{\rho}{2}(\sum_{i=1}^m \|u_i^{k*} + \alpha_i^{(k+1)*} - x_i^{(k+1)*}\|_2^2) \\ &\leqslant f^* + 2\rho_k\left(\sum_{i=1}^m (u_i^{k*} + \alpha_i^{(k+1)*}) x_i^{(k+1)*}\right) \\ &\leqslant f^* + 2\rho_k\left(\sum_{i=1}^m \|u_i^{k*} + \alpha_i^{(k+1)*}\|_2 \|x_i^{(k+1)*}\|_2\right) \\ &= f^* + 2\rho_k\sqrt{\delta_k}\sum_{i=1}^m \|x_i^{(k+1)*}\|_2 \\ &\leqslant f^* + 2\rho_k\sqrt{\delta_k} m.\end{aligned} \tag{26}$$

The second inequality uses Lemma 1 and Average Value inequality. The third inequality uses Cauchy-Schwartz inequality. The second equality uses Lemma 2. On the other hand, we can get,

$$\begin{aligned}\mathcal{L}_g \quad &(\alpha^{(k+1)*}, x^{(k+1)*}) \geqslant \mathcal{L}_g(\alpha^{(k+1)*}, x^{k+1}) - \sum_{i=1}^m \mathcal{G}(x_i^{k+1}) \\ &\geqslant f^* - \sum_{i=1}^m \mathcal{G}(x_i^{k+1}) = f^* - m * \mathbb{E}[\mathcal{G}(x_i^{k+1})] = f^* - \tfrac{m}{2}\end{aligned} \tag{27}$$

We complete the proof by considering the other parameters,

$$\begin{aligned}f^* - \tfrac{m+n}{2} &\leqslant \mathcal{L}_g(\alpha^{(k+1)*}, x^{(k+1)*}, \beta^{(k+1)*}, y^{(k+1)*}) \\ &\leqslant f^* + 2\rho_k\sqrt{\delta_k}(m+n).\end{aligned} \tag{28}$$

5. Experiments

In this section, extensive experiments have been conducted on four datasets to evaluate the effectiveness of our RFSS algorithm. The values of the parameters d and λ will be adjusted to analyse their effects on our algorithm and performance comparison with ten state-of-the-art recommendation algorithms will be reported. Source code is available athttps://github.com/sysulawliet/RFSS.

5.1. Dataset description

The four datasets used in our experiments are Jester[1], MovieLens[2], FindFoods[3] and BaiduMovie[4].

1. The Jester dataset collects continuous ratings of joke from April 1999 to May 2003 provided by University of California. The dataset is relatively dense because each user has rated many items and there are some jokes almost all users have rated. There are negative ratings on the datasets, so we map the range of the rating from the interval [−10, 10] to [0, 20].
2. The MovieLens dataset contains ratings of the online movie recommender service MovieLens. All the users have rated at least 20 movies but the user-item rating matrix is still quite sparse since the number of movies is far larger than 20.
3. The BaiduMovie dataset comes from the Movie Recommendation Algorithm Contest of Baidu. The data only stands for the active users and each user has made enough records.
4. The FindFoods dataset, which is collected by Stanford University, consists of reviews of find foods from Amazon up to October 2012.

The properties of the four datasets are summarized in Table 1. The four datasets split randomly with 80% as training set and 20% as testing set.

5.2. Evaluation methodology

In order to evaluate the quality of the recommendation algorithms, two widely used evaluation metrics, namely Mean Absolute Error (MAE) and Root Mean Square Error (RMSE), will be used to measure the accuracy of the predicted ratings, which are defined as follows:

$$MAE = \frac{1}{T}\sum_{i,j} |r_{ij} - p_{ij}|, \; RMSE = \sqrt{\frac{1}{T}\sum_{i,j}(r_{ij} - p_{ij})^2}, \tag{29}$$

where T denotes the number of tested ratings. By definition, a smaller MAE and RMSE value means better prediction quality of an algorithm. RMSE is sensitive to large errors while MAE is sensitive to the accumulation of small errors.

5.3. Parameter analysis

We analyse the effect of the dimensionality d of latent feature vector and the regularization coefficient λ on the performance of the RFSS algorithm. We vary the value of λ from 0 to 1 with $d = 2, 5$ and 10 and set the step size to 0.1. The results are shown in Fig. 2 and Fig. 3. In general, the performance of the algorithm will become better as the increase of the dimensionality d because using more latent features to describe users and items leads to more accurate characterizations. In other words, the ratings can be fitted better by high dimensional vectors. On the MovieLens and BaiduMovie datasets, the algorithm has similar effects when d is equal to 5 and 10 since it is enough to construct the latent features of users and items with 5 dimensions when the other parameters are fixed. However, the performance will be worse when d increases from 5 to 10 on the FindFoods dataset and the reason might be due to the overfitting when $d = 10$. We observe that the algorithm on each dataset has different sensitivities to λ due to the inherent differences of data properties. Generally speaking, the performance first increases and then decreases as the increasing of parameter λ and the algorithm can achieve the best results when λ is around 0.5. When $\lambda = 0$, our algorithm degenerates into the traditional matrix factorization methods which do not consider the user-item relationship in feature space and has a poor effect. On the other hand, the performance is not good as well when $\lambda = 1$ since over-emphasizing the user-item relationship in feature space leads to the under-fitting of ratings. So, the RFSS algorithm can get a good performance with suitable influence of the user-item relationship in feature space.

[1] http://eigentaste.berkeley.edu/dataset.
[2] http://grouplens.org/datasets/movielens.
[3] http://snap.stanford.edu/data/web-FineFoods.html.
[4] http://openresearch.baidu.com.

Table 1
The details of the four datasets.

Datasets	Jester	MovieLens	BaiduMovie	FindFoods
#users	59,132	6040	9722	256,059
#items	140	3952	7889	74,258
#ratings	1,761,439	1,000,209	1,262,741	568,454
density	0.2127	0.0419	0.0164	0.0002

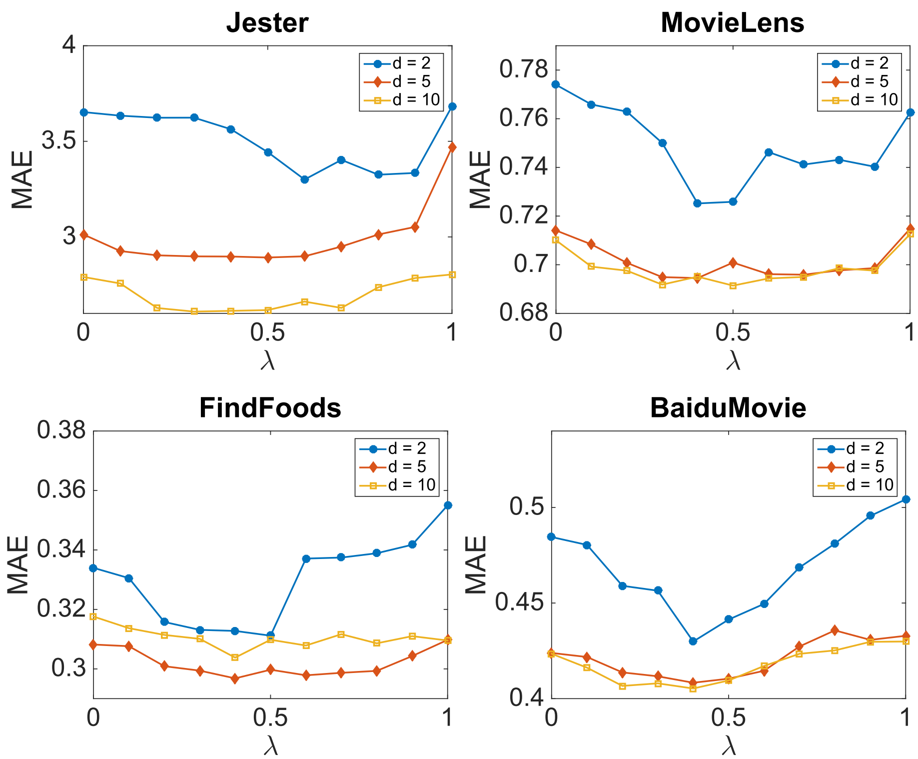

Fig. 2. The values of MAE with different d and λ values on the four datasets.

5.4. Comparison experiments

We present the comparison results on the predicted ratings in terms of accuracy between the proposed RFSS algorithm and ten state-of-the-art recommendation algorithms, namely **UBCF** (User-Based Collaborative Filtering) (Wang et al., 2006), **IBCF** (Item-Based Collaborative Filtering) (Sarwar et al., 2001), **SO** (Slope One) (Lemire and Maclachlan, 2005), **BS** (Bayesian Similarity) (Guo et al., 2013), **IRSVD** (Improving Regularized Singular Value Decomposition) (Paterek, 2007), **SVD++** (Singular Value Decomposition Plus Plus) (Koren, 2008), **CosMF** (Cosine Matrix Factorization) (Wen et al., 2014), **SCMF** (Sparse Covariance Matrix Factorization) (Shi et al., 2013), **MF-AMSD** (Matrix factorization to asymmetric user similarities) (Pirasteh et al., 2015), **ERBMMA** (expected risk minimized matrix approximation method) (Li et al., 2017). UBCF and BS consider the distances between users while IBCF and SO consider the distances between items, and the four recommendation algorithms use the distances and rating records to predict ratings to non-purchased items. IRSVD, SVD++, CosMF, SCMF, MF-AMSD and ERBMMA are matrix factorization methods which find latent features of users and items and then make predictions based on these features. Except CosMF, the other five algorithms use inner product to predict ratings and all of them do not consider the relationship between users and items in feature space. We set $\lambda = 0.5$ in our algorithm for all datasets. Without loss of generality, we set dimensionality $d = 5$ for all matrix factorization methods.

The comparison results in terms of MAE and RMSE are reported in Fig. 4 and Fig. 5 respectively. The values of percentage gain obtained by the proposed RFSS algorithm over the existing methods are shown in Table 2. The accuracies generated by SO are quite poor on the four datasets, i.e., its MAE or RMSE is quite large and RFSS can achieve the improvement from 15% to 40%. The performances of UBCF, IBCF, SCMF and MF-AMSD are similar on the Jester and MovieLens datasets and RFSS can make about 5% improvement. Each item on FindFoods has many ratings so the item similarities can be calculated accurately which makes IBCF works well. RMSE is sensitive to large errors while MAE is sensitive to the accumulation of small errors. Because there are some cold users and items in the dataset leading to some large prediction errors, IBCF outperforms RFSS in terms of RMSE on FindFoods. Besides, SCMF is significantly better than MF-AMSD on the two datasets. The performance of the BS algorithm is pretty good on the Jester and FindFoods datasets. Compared with BS, RFSS can make about 3% improvement on the MovieLens and BaiduMovie datasets but more than 12% on the other two. The IRSVD algorithm does not work as well as our method but is still impressive on all datasets and RFSS can achieve 1% to 14% improvement, and the performance of ERBMMA is at least 1% better than IRSVD on the four datasets except BaiduMovie. RFSS can achieve 2% to 8% improvement over SVD++, which just considers the relationships within users and within items. Although CosMF uses cosine to predict ratings, the unrestricted vectors make it predict ratings for some pairs of user

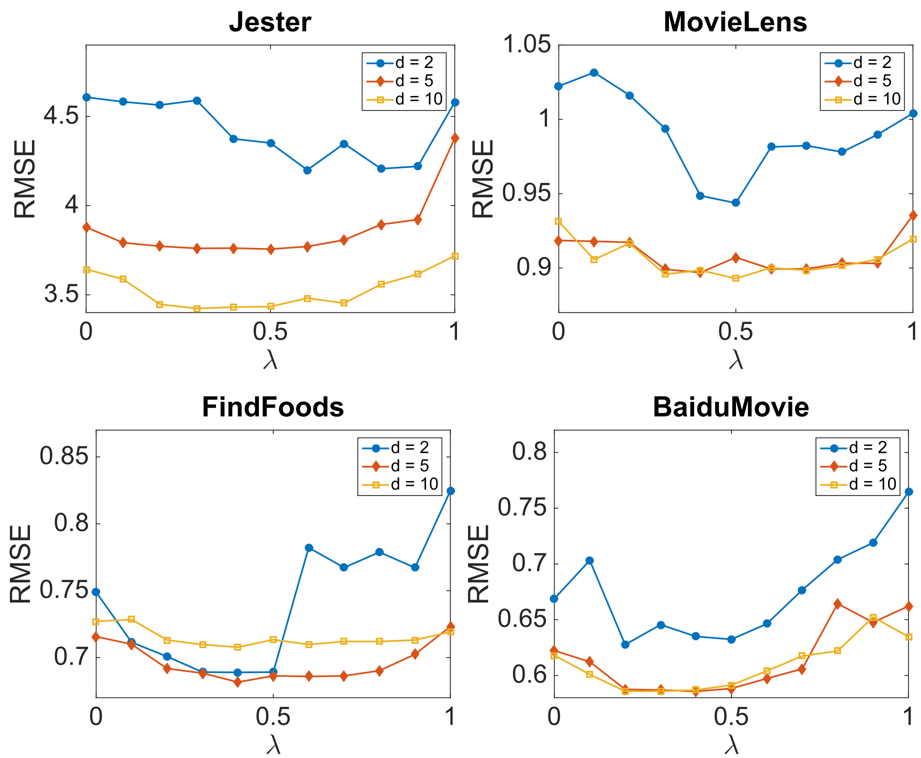

Fig. 3. The values of RMSE with different d and λ values on the four datasets.

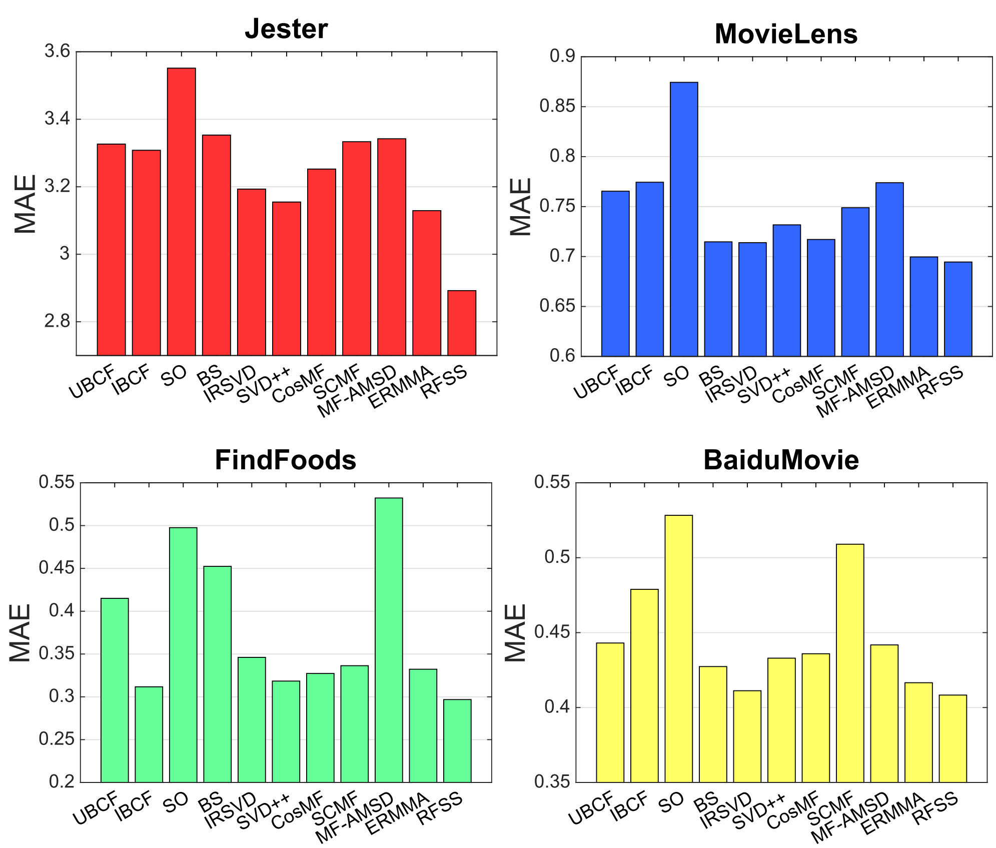

Fig. 4. The values of MAE on the four datasets of the eleven recommendation algorithms.

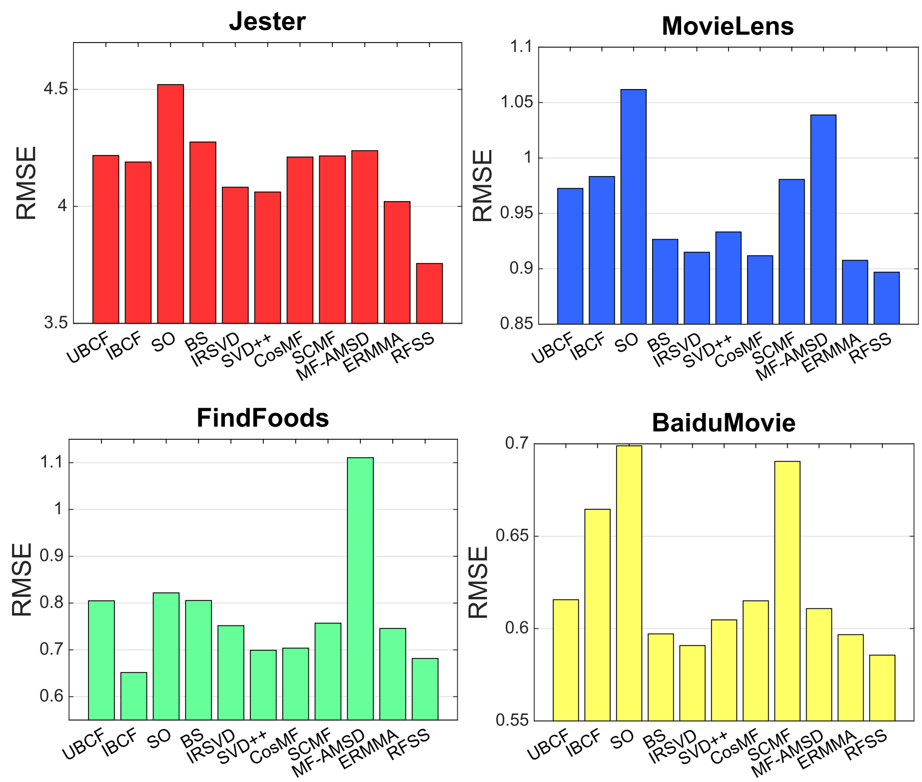

Fig. 5. The values of RMSE on the four datasets of the eleven recommendation algorithms.

Table 2
The values of percentage gain obtained by the RFSS algorithm over the ten compared algorithms.

Dataset	Jester		MovieLens		FindFoods		BaiduMovie	
Algorithm	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
UBCF	13.05%	10.94%	9.26%	7.77%	28.48%	15.32%	7.84%	4.88%
IBCF	12.57%	10.35%	10.31%	8.78%	4.78%	−4.61%	14.74%	11.89%
SO	18.56%	16.91%	20.57%	15.53%	40.35%	17.06%	22.71%	16.21%
BS	13.74%	12.14%	2.83%	3.19%	34.39%	15.38%	4.45%	1.93%
IRSVD	9.42%	7.98%	2.71%	1.97%	14.24%	9.33%	0.71%	0.88%
SVD++	8.32%	7.52%	5.08%	3.89%	6.79%	2.51%	5.69%	3.16%
CosMF	11.07%	10.80%	3.15%	1.63%	9.32%	3.14%	6.33%	4.78%
SCMF	13.23%	10.90%	7.26%	8.54%	11.76%	9.97%	19.78%	15.19%
MF-AMSD	13.46%	11.37%	10.26%	13.66%	44.23%	38.63%	7.58%	4.13%
ERMMA	7.57%	6.58%	0.73%	1.18%	10.67%	8.6%	1.98%	1.86%

and item improperly. So RFSS can make 1% to 11% improvement over CosMF. In general, the proposed RFSS algorithm can make a more accurate rating prediction than the existing state-of-the-art recommendation algorithms on the four datasets.

6. Conclusion

In this paper, we have proposed a recommendation in feature space sphere (RFSS). Different from the traditional matrix factorization methods, the proposed algorithm considers the relationship between users and items when finding suitable positions for all users and items in the unit sphere feature space according to ratings. Besides, the predicted ratings are concerned with the distances between users and items in the feature space and the adaptive user-dependent coefficients. We optimize our objective function with constraints by the ADMM algorithm. Extensive experiments have been conducted on four real-world datasets. The results have confirmed that the relationship in feature space has an impact on the quality of recommendation and the proposed method significantly outperforms the existing recommendation algorithms.

Acknowledgments

This work was supported by Guangdong Natural Science Funds for Distinguished Young Scholar (2016A030306014) and Tip-top Scientific and Technical Innovative Youth Talents of Guangdong special support program (2016TQ03X542).

References

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press.
Boyd, S.P., Parikh, N., Chu, E., Peleato, B., Eckstein, J., 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. Found. Trends Mach. Learn. 3 (1), 1–122.
Chen, W., Hsu, W., Lee, M., 2013. Making recommendations from multiple domains. In: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 892–900.
Davidson, J., Liebald, B., Liu, J., Nandy, P., Vleet, T.V., Gargi, U., Gupta, S., He, Y., Lambert, M., Livingston, B., Sampath, D., 2010. The YouTube video

recommendation system. In: Proceedings of the 2010 ACM Conference on Recommender Systems. pp. 293–296.

Eckstein, J., Bertsekas, D.P., 1992. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. Math. Program. 55, 293–318.

Elmongui, H.G., Mansour, R., Morsy, H., Khater, S., El-Sharkasy, A., Ibrahim, R., 2015. TRUPI: twitter recommendation based on users' personal interests. In: Proceedings of the 16th International Conference on Computational Linguistics and Intelligent Text Processing, vol. 2. pp. 272–284.

Gao, S., Luo, H., Chen, D., Li, S., Gallinari, P., Guo, J., 2013. Cross-domain recommendation via cluster-level latent factor model. In: Proceedings of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases. pp. 161–176.

Guo, G., Zhang, J., Yorke-Smith, N., 2013. A novel Bayesian similarity measure for recommender systems. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence. pp. 2619–2625.

Guo, G., Zhang, J., Thalmann, D., Yorke-Smith, N., 2014. Leveraging prior ratings for recommender systems in e-commerce. Electron. Commer. Res. Appl. 13 (6), 440–455.

He, B.S., Yang, H., Wang, S.L., 2000. Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. J. Optim. Theory Appl. 106 (2), 337–356.

Hestenes, M.R., 1969. Multiplier and gradient methods. Optim. Theory Appl. 4, 302–320.

Jannach, D., Zanker, M., Felfering, A., Friedrich, G., 2013. Recommender Systems: An Introduction. Addison Wesley Publishing.

Kang, Z., Peng, C., Cheng, Q., 2016. Top-n recommender system via matrix completion. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. pp. 179–185.

Koenigstein, N., Dror, G., Koren, Y., 2011. Yahoo! music recommendations: modeling music ratings with temporal dynamics and item taxonomy. In: Proceedings of the 2011 ACM Conference on Recommender Systems. pp. 165–172.

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

Lee, D.D., Seung, H.S., 2000. Algorithms for non-negative matrix factorization. Adv. Neural Inf. Process. Syst. 13, 556–562.

Lemire, D., Maclachlan, A., 2005. Slope one predictors for online rating-based collaborative filtering. In: Proceedings of the 2005 SIAM International Conference on Data Mining. pp. 471–475.

Li, D., Chen, C., Lv, Q., Shang, L., Chu, S.M., Zha, H., 2017. ERMMA: expected risk minimization for matrix approximation-based recommender systems. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence. pp. 1403–1409.

Linden, G., Smith, B., York, J., 2003. Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Comput. 7, 76–80.

Ma, H., Zhou, D., Liu, C., Lyu, M.R., King, I., 2011. Recommender systems with social regularization. In: Proceedings of the Forth International Conference on Web Search and Web Data Mining. pp. 287–296.

Ning, X., Karypis, G., 2011. SLIM: sparse linear methods for top-n recommender systems. In: Proceedings of the 11th IEEE International Conference on Data Mining. pp. 497–506.

Parikh, N., Boyd, S.P., 2014. Proximal algorithms. Found. Trends Optim. 1 (3), 127–239.

Paterek, A., 2007. Improving regularized singular value decomposition for collaborative filtering. In: Proceedings of KDD Cup & Workshop. pp. 39–42.

Pirasteh, P., Hwang, D., Jung, J.J., 2015. Exploiting matrix factorization to asymmetric user similarities in recommendation systems. Knowl. Syst., 1–7

Powell, M.J.D., 1969. A method for nonlinear constraints in minimization problems. Optimization 5 (6), 283–298.

Ruslan, S., Andriy, M., 2008. Bayesian probabilistic matrix factorization using markov chain monte carlo. In: Proceedings of the Twenty-Fifth International Conference on Machine Learning. pp. 880–887.

Salakhutdinov, R., Mnih, A., 2007. Probabilistic matrix factorization. Adv. Neural Inf. Process. Syst. 20, 1257–1264.

Sarwar, B., Karypis, G., Konstan, J., Riedl, J., 2001. Item-based collaborative filtering recommendation algorithms. In: Proceedings of the Tenth International World Wide Web Conference. pp. 285–295.

Shi, J., Wang, N., Xia, Y., Yeung, D.-Y., King, I., Jia, J., 2013. SCMF: sparse covariance matrix factorization for collaborative filtering. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence. pp. 2705–2711.

Wang, S.L., Liao, L.Z., 2001. Decomposition method with a variable parameter for a class of monotone variational inequality problems. Optim. Theory Appl. 109 (2), 415–429.

Wang, J., de Vries, A.P., Reinders, M.J., 2006. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 501–508.

Wen, H., Ding, G., Liu, C., Wang, J., 2014. Matrix factorization meets cosine similarity: Addressing sparsity problem in collaborative filtering recommender system. In: Proceedings of the 16th Asia-Pacific Web Conference. pp. 306–317.

Zhong, W., Jin, R., Yang, C., Yan, X., Zhang, Q., Li, Q., 2015. Stock constrained recommendation in tmall. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 2287–2296.