Volume 1 | Issue 4 Article 1

2018

A Survey of Matrix Completion Methods for Recommendation Systems

Andy Ramlatchan
NASA Langley Research Center, Hampton, VA 23666, USA and the Department of Computer Science, Old Dominion University, Norfolk, VA 23666, USA.

Mengyun Yang
the Department of Computer Science, Central South University, Changsha 410083, China the Department of Science, Shaoyang University, Shaoyang 422000, China.

Quan Liu
the Department of Computer Science, Central South University, Changsha 410083, China.

Min Li
the Department of Computer Science, Central South University, Changsha 410083, China.

Jianxin Wang
the Department of Computer Science, Central South University, Changsha 410083, China.

See next page for additional authors

BIG DATA MINING AND ANALYTICS
ISSN 2096-0654 05/06 pp308–323
Volume 1, Number 4, December 2018
DOI: 10.26599/BDMA.2018.9020008

A Survey of Matrix Completion Methods for Recommendation Systems

Andy Ramlatchan, Mengyun Yang, Quan Liu, Min Li, Jianxin Wang, and Yaohang Li*

Abstract: In recent years, the recommendation systems have become increasingly popular and have been used in a broad variety of applications. Here, we investigate the matrix completion techniques for the recommendation systems that are based on collaborative filtering. The collaborative filtering problem can be viewed as predicting the favorability of a user with respect to new items of commodities. When a rating matrix is constructed with users as rows, items as columns, and entries as ratings, the collaborative filtering problem can then be modeled as a matrix completion problem by filling out the unknown elements in the rating matrix. This article presents a comprehensive survey of the matrix completion methods used in recommendation systems. We focus on the mathematical models for matrix completion and the corresponding computational algorithms as well as their characteristics and potential issues. Several applications other than the traditional user-item association prediction are also discussed.

Key words: matrix completion; collaborative filtering; recommendation systems

1 Introduction

Technology has given corporations and consumers more analytical capabilities than ever before, largely due to the birth of big data, and the possibilities that spring up from its utilization. Users can easily answer almost any encountered question, and in many cases, can answer unexpected questions. Personal mobile devices can collect data on every communication a person makes, every image a person captures or receives, every video a person records or receives, and every online transaction a person makes. More importantly, corporations can now store all the needed information. This is of incredible value to such corporations because the entire activities of a person can inform them on his/her particular daily habits, and this can be aggregated from an entire group. On the other hand, the huge amount of data also makes it difficult for the users to make decisions that best fit their needs. A similar difficulty is presented in the corporations providing commodities and services, as it becomes difficult to process the data to understand the user behaviors.

Fortunately, the recent advances in the field of recommendation systems (a.k.a. recommender systems or recommender engines), a sub-field of machine learning, have provided the capability of making predictions based on the past activities of a user or his/her associations with other users' behaviors. Many computational algorithms have been developed for recommendation systems, which can predict the future interests of users based on past preferences considering how much and how little

• Andy Ramlatchan is with NASA Langley Research Center, Hampton, VA 23666, USA and the Department of Computer Science, Old Dominion University, Norfolk, VA 23666, USA. E-mail: andy.ramlatchan@nasa.gov.
• Mengyun Yang is with the Department of Computer Science, Central South University, Changsha 410083, China and the Department of Science, Shaoyang University, Shaoyang 422000, China. E-mail: yangmengyun@csu.edu.cn.
• Quan Liu, Min Li, and Jianxin Wang are with the Department of Computer Science, Central South University, Changsha 410083, China. E-mail: {liuquan, limin, jxwang}@csu.edu.cn.
• Yaohang Li is with the Department of Computer Science, Old Dominion University, Norfolk, VA 23529, USA. E-mail: yaohang@cs.odu.edu.
∗ To whom correspondence should be addressed.
Manuscript received: 2018-01-21; accepted: 2018-03-20

a user may prefer one item over another, such as user rankings. Recommendation systems have attracted much attention in both research and practice, since they can narrow complex and difficult decisions into a few recommendations. The recommendation system techniques have been applied in diverse fields, including movies[1], music[2], television[3], books[4], e-learning[5], web search[6], jokes[7], news[6], bioinformatics[8, 9], and engineering[10].

Generally, a recommendation system is a subset of the information filtering systems, whose goal is to predict the rating a user would give to an item of commodity. The recommendations are typically made through either content-based filtering or collaborative filtering approaches. The content-based filtering approaches utilize a set of discrete features that characterize a commodity and build a user profile that indicates the items the user liked in the past. Then, items with similar properties are recommended. Instead of using item features and user profiles, the collaborative filtering approaches produce recommendations based on a user as well as other users' past behaviors. The fundamental assumption under collaborative filtering is that if the users share similar ratings in the past on the same set of items, then they would likely rate the other items similarly. Content-based filtering and collaborative filtering can be combined to build hybrid recommendation systems, which often demonstrate better recommendation precision than pure recommendation approaches.

In literature, a few surveys overview different aspects of recommendation systems. Bobadilla et al.[11] presented the evolution of recommendation systems. Kunaver and Požrl[12] reviewed the work done in the area of recommendation diversity. Burke[13] discussed the implementation issues in hybrid recommendation systems. Desrosiers and Karypis[14] provided a survey on recommendation methods based on neighborhood information. He et al.[15] emphasized on the influences of human factors in recommendation systems. Campos et al.[16] developed a review on recommendation approaches dealing with temporal context information. Yang et al.[17] investigated how social network information can be adopted by recommendation systems. Klašnja-Milicevic et al.[18] studied recommendation systems for online-based education and learning. Yera and Martínze[19] examined the fuzzy tools used in recommendation systems. Recently, Kotkov et al.[20] considered serendipity within recommendation systems. In this article, we focus on the matrix completion methods in collaborative filtering approaches. This is because the collaborative filtering problem can often be modeled as a matrix completion problem, whose goal is to fill out the unknown values where the users are not inclined to certain items. We overview the mathematical models for matrix completion used in recommendation systems. We then survey the computational algorithms designed for these models, analyze their characteristics, and discuss the potential issues.

The rest of this survey article is organized as follows. In Section 2, the matrix completion problem and low-rank assumption are discussed. Various matrix completion models are analyzed in Section 3, and the computational algorithms considering these models are described in Section 4. Then, in Section 5, the uses of recommendation systems based on matrix completion on several applications other than traditional user-item association predictions are discussed. Finally, Section 6 summarizes our conclusions and research directions.

2 Matrix Completion Problem

A typical collaborative filtering scenario in recommendation systems can be modeled as a matrix completion problem. Given a list of m users $\{u_1, u_2, \ldots, u_m\}$ and n items $\{i_1, i_2, \ldots, i_n\}$, the preferences of users toward the items can be represented as an incomplete $m \times n$ matrix A, where each entry either represents a certain rating or is unknown. The ratings in A can be explicit indications, such as scores given by the users in scales $1-5$ or ordinal favorability (e.g., strongly agree, agree, neutral, disagree, and strongly disagree). These ratings can also be implicit indications, e.g., item purchases, website/store visits, or link click-throughs. It is generally assumed a user rates a specific item only once. As a result, recommendations can be made by filling out the unknown entries and then ranking them according to the predicted values.

Denoting Λ as the complete set of N entries in A with known ratings, the general matrix completion problem is defined as finding a matrix R such that

$$R_{ui} = A_{ui},$$

for all entries $(u,i) \in \Lambda$. In addition, we denote $\bar{\Lambda}$ as the complement set to Λ, and $P_\Lambda(A)$ as an orthogonal projector onto Λ which is an $m \times n$ matrix with the known elements of A preserved and the unknown elements as 0 s[21]. However, since the number of known entries is less than the overall number of entries, there exist infinitely many solutions. Nevertheless, it is commonly believed that only a few latent factors[22] influence how much a user likes an item. For example, studies show that the attributes of actor/actress, director, and decade contribute most to a user's preference to a movie. This relatively small number of influence factors compared to the total number of users or items in the rating matrix A provides a guiding framework to fill in the missing values and to select the correct complete matrix. This corresponds to the low-rank assumption in matrix completion, i.e., the rating matrix A is low-rank or approximately low-rank. The low-rank assumption in matrix completion also agrees with the well-known Occam's razor principle in machine learning, whose goal is to find the "simplest" complete matrix X that is consistent with the known ratings in A.

3 Mathematical Models

Starting from the baseline model, we investigate various mathematical models, deterministic and probabilistic, that have been developed to address the matrix completion problem. The fundamental assumption is that a low-dimensional representation of users and items exists, although probably unknown, which can be used to accurately model the user-item association. Such low-dimensional representation is often characterized by a low-rank matrix. We also study models that employ various regularization methods and incorporate various constraints in the completed matrix.

3.1 Baseline model

Denoting μ as the average rating among all known ratings in the rating matrix A, the baseline model[23] fills out a missing element R_{ui} by

$$R_{ui} = \mu + b_u + b_i \tag{1}$$

where b_u and b_i represent the observed deviations of user u and item i from μ, respectively. The training parameters b_u and b_i can be estimated by solving the following least squares problem:

$$\min_{b_*} \|P_\Lambda(R) - P_\Lambda(A)\|_{\mathrm{F}}^2 + \lambda \left(\sum_u (b_u)^2 + \sum_i (b_i)^2 \right) \tag{2}$$

where λ is the regularization parameter. The first term $\|P_\Lambda(R) - P_\Lambda(A)\|_{\mathrm{F}}^2 = \sum_{(u,i)\in\Lambda}(R_{ui} - A_{ui})^2$ attempts to minimize the training error, while the second term $\lambda(\sum_u (b_u)^2 + \sum_i (b_i)^2)$ serves as the regularizing term to avoid overfitting by penalizing the magnitudes of b_u and b_i.

3.2 SVD model

The fundamental idea of the Singular Value Decomposition (SVD) model proposed by Sarwar et al.[24] is to decompose the rating matrix A into a user feature matrix, a singular value matrix, and an item feature matrix of low-rank. Starting from a normalized matrix A_{norm}, by filling out the missing elements with preliminary simple predictions, the SVD model carries out an SVD operation on A_{norm} such that

$$A_{norm} = U\varSigma V^{\mathrm{T}} \tag{3}$$

where $\varSigma$ is a diagonal matrix with descendently sorted singular values deposited in its diagonal entries, and the U and V columns contain the corresponding left and right singular vectors, respectively. By truncating the diagonal matrix $\varSigma$ to a top-r rank $\varSigma_r$, then $U_r\varSigma_r^{\frac{1}{2}}$ and $V_r\varSigma_r^{\frac{1}{2}}$ represent the latent factor vectors for users and items, respectively. The dot product of the u-th row of $U_r\varSigma_r^{\frac{1}{2}}$ and the i-th row of $V_r\varSigma_r^{\frac{1}{2}}$ yields the predicted u-th user rating of the i-th item. Sarwar et al.[25] employed a "folding-in" technique to build an SVD by incrementally adding new users and items so that the SVD model can be scalable and built faster; however, this may lead to quality loss. Instead of carrying out the dot product operation, Billsus and Pazzani[26] used the latent vectors as feature vectors to train an artificial neural network to predict the user ratings.

3.3 Matrix factorization model

The matrix factorization model is a generalization of the SVD model, which intends to find a low-rank matrix factorization to approximate A. Assuming an r-dimensional vector x_u is associated with each user u and measures the latent factors influencing the preference of items, and an r-dimensional vector y_i is associated with each item i and represents the latent factors influencing i, the matrix factorization model uses the dot product $y_i^{\mathrm{T}}x_u$ to capture the correlation between user u and item i. The predicted rating then becomes

$$R_{ui} = y_i^{\mathrm{T}}x_u \tag{4}$$

Assuming the columns of X and Y contain all x_u and y_i vectors, respectively, the goal of the matrix completion is to estimate

$$R = Y^{\mathrm{T}}X \tag{5}$$

The parameters to be learned are the user feature vectors x_u and the item feature vectors y_i, which can be done by minimizing the Frobenius norm error as follows:

$$\min_{x_*, y_*} \|P_\Lambda(R) - P_\Lambda(A)\|_F^2 \tag{6}$$

The potential problem of model (6) is that minimizing the Frobenious norm can easily lead to overfitting by biasing to the known entries.

3.4 *l*2-regularized matrix factorization model

To avoid overfitting the observed user-item ratings, the $l2$-norm regularized matrix factorization method[27] uses $l2$-norm to regularize the learning parameters by penalizing their magnitudes. Based on the matrix factorization model (6), this can be done by minimizing the regularized $l2$-norm error of x_u and y_i in addition to the Frobenious norm error term as follows[28]:

$$\min_{x_*, y_*} \|P_\Lambda(R) - P_\Lambda(A)\|_F^2 + \lambda_1 \left(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2 \right) \tag{7}$$

where λ_1 is a constant controlling the extent of regularization.

A more sophisticated $l2$-regularized matrix factorization model can be built on top of the baseline model by considering the user deviation b_u and the item deviation b_i. Then, each predicted rating $\widehat{R}_{ui}$ in $\hat{R}$ becomes

$$\widehat{R}_{ui} = \mu + b_u + b_i + y_i^T x_u \tag{8}$$

The parameters to be learned become b_u, b_i, x_u, and y_i, which can be done by minimizing the regularized $l2$-norm error as follows:

$$\min_{b_*, x_*, y_*} \|P_\Lambda(\widehat{R}) - P_\Lambda(A)\|_F^2 + \lambda_2 \sum_{(u,i)\in\Lambda} \left(b_u^2 + b_i^2 + \|x_u\|^2 + \|y_i\|^2\right) \tag{9}$$

where λ_2 is the regularization parameter. Because there are more training parameters, this model often yields a more accurate prediction.

Prediction accuracy in regularized matrix factorization algorithm can often be improved by incorporating additional information or factors. Vozalis and Marqaritis[29] utilized demographic data as an additional source of information. A more famous example is the SVD++ method[30], which is considered as the model with the highest accuracy in the Netflix Prize[31]. The SVD++ enhances the regularized SVD model by considering implicit feedback as an additional indication of user preferences. In the SVD++, in addition to the latent factor x_u associated with each user u, which measures the latent factors of u influencing the preference of items, a set of item vectors are incorporated, relating to each item rated by the user u. Then, the user vector becomes $x_u + |W(u)|^{-\frac{1}{2}} \sum_{j\in R(u)} p_j$, and the predicted rating of user u for item i is calculated as

$$\widehat{\widehat{R}}_{ui} = \mu + b_u + b_i + y_i^T \left(x_u + |W(u)|^{-\frac{1}{2}} \sum_{j\in R(u)} p_j \right) \tag{10}$$

where $W(u)$ denotes the set of items associated with user u. The parameters to be learned are b_u, b_c, x_u, p_j, and y_c, which can be done by minimizing the regularized squared error as follows:

$$\min_{b_*, x_*, p_*, y_*} \|P_\Lambda(\widehat{\widehat{R}}) - P_\Lambda(A)\|_F^2 + \lambda_3 \sum_{(u,i)\in\Lambda} \left(b_u^2 + b_i^2 + \|y_i\|^2 + \left\| x_u + |W(u)|^{-\frac{1}{2}} \sum_{j\in R(u)} p_j \right\|^2 \right) \tag{11}$$

where λ_3 is the regularization parameter for model (11).

3.5 *l*1-regularized SVD model and *l*1/*l*2-regularized SVD model

Other regularization methods other than the $l2$-norm can also be incorporated to the SVD model. The $l1$-regularized SVD model[32] can generate sparse solutions, and the minimization problem then becomes

$$\min_{x_*, y_*} \|P_\Lambda(R) - P_\Lambda(A)\|_F^2 + \lambda_4 \sum_{(u,i)\in\Lambda} (|x_u| + |y_i|) \tag{12}$$

where λ_4 is the regularization parameter controlling the extent of the $l1$-norms of the decomposed matrices y_i and x_u.

Considering that $l1$-regularization can generate sparse solutions, while $l2$-regularization often leads to more accurate predictions, the $l1/l2$-regularized SVD model attempts to benefit from both by combining the $l1$-norm and $l2$-norm. As a result, the corresponding minimization objective function becomes

$$\min_{x_*, y_*} \|P_\Lambda(R) - P_\Lambda(A)\|_F^2 + \lambda_5 \sum_{(u,i)\in\Lambda} (\alpha(\|x_u\|^2 + \|y_i\|^2) + (1-\alpha)(|x_u| + |y_i|)) \tag{13}$$

where α is a tunable parameter to balance the $l1$-norm and $l2$-norm terms, and where λ_5 is the regularization parameter controlling the extent of the $l1$- and $l2$-norms combination.

3.6 Spectral regularization model

Instead of applying regularization on the decomposed matrices, Mazumder et al.[33], inspired by Candes and Tao[21], proposed a spectral regularization model that uses the nuclear (trace) norm of the recovered matrix R, which is defined as the sum of the singular values in R. The objective of the matrix regularization model is to balance the minimization of the approximation errors in the known entries and the nuclear norm of R,

$$\min_R \frac{1}{2}\|P_\Lambda(R) - P_\Lambda(A)\|_{\mathrm{F}}^2 + \lambda_6\|R\|_* \tag{14}$$

where λ_6 is the regularization parameter controlling the extent of the nuclear norm. Note that Formula (14) is a convex model for completing matrix A.

3.7 Rank minimization model

Under the low-rank assumption, the matrix completion problem can be formulated as a matrix rank optimization problem such that

$$\begin{aligned} &\min_R rank(R), \\ &\text{s.t., } R_{ui} = A_{ui}, (u,i) \in \Lambda \end{aligned} \tag{15}$$

where $rank(R)$ denotes the rank of matrix R. Unfortunately, finding the exact solution for the above rank optimization problem is well-known to be NP-hard[34]. Nevertheless, the low-rank matrix approximation is the general principle used in the matrix completion algorithms for recommendation systems.

3.8 Nuclear norm minimization model

The rank optimization problem can be relaxed to a nuclear norm (trace norm) optimization problem[35] by minimizing the sum of the singular values of R. Then, the matrix completion problem is reformulated as a convex optimization problem such as

$$\begin{aligned} &\min_R \|R\|_*, \\ &\text{s.t., } R_{ui} = A_{ui}, (u,i) \in \Lambda \end{aligned} \tag{16}$$

where $\|\cdot\|_*$ denotes the nuclear norm. Candes and Recht[36] showed that the solution obtained by optimizing the nuclear norm is equivalent to that obtained by rank minimization in Formula (15), under certain mild conditions.

If the application is "noisy", the nuclear norm minimization problem can be modeled as

$$\begin{aligned} &\min_R \|R\|_*, \\ &\text{s.t., } |R_{ui} - A_{ui}| < \delta, (u,i) \in \Lambda \end{aligned} \tag{17}$$

where δ is the tolerance parameter to relax the $R_{ui} = A_{ui}, (u,i) \in \Lambda$ condition in Formula (16).

3.9 Matrix factorization minimization model

For recommendation systems that involve the completion of a large matrix, handling the intermediate $m \times n$ matrices $R^{(j)}$ at each iteration step is costly from both computation and storage point of view. Instead of storing the complete recovered matrix, the matrix factorization minimization model uses an r-rank matrix factorization, $R = XY$, to represent the completed matrix R. Then, the matrix completion problem is formulated as a non-convex quadratic optimization problem by minimizing the sum of the Frobenius norms of X and Y:

$$\begin{aligned} &\min_{X,Y}(\|X\|_{\mathrm{F}}^2 + \|Y\|_{\mathrm{F}}^2), \\ &\text{s.t., } P_\Lambda(XY) = P_\Lambda(A) \end{aligned} \tag{18}$$

Assuming that X is $m \times r$ and Y is $r \times n$, and $r \ll m, n$, the storage requirement of XY becomes $(m + n) \times r$, which is significantly less than that of the $m \times n$ matrix R. Recht et al.[37] shows that if r is sufficiently greater than the rank of the optimal solution of the nuclear norm minimization model, the non-convex quadratic optimization is equivalent to the nuclear norm minimization.

An alternative matrix factorization model is designed for matrix completion by Wen et al.[38], which leads to the low-rank matrix fitting (LMaFit) algorithm:

$$\begin{aligned} &\min_{X,Y}(\|XY - Z\|_{\mathrm{F}}^2), \\ &\text{s.t., } P_\Lambda(Z) = P_\Lambda(A) \end{aligned} \tag{19}$$

Similar to Formula (18), although the minimization is convex, the constraint is not. Consequently, Formula (19) is also a non-convex optimization model, which cannot guarantee globally optimal solutions.

3.10 Probabilistic model

The matrix completion problem is addressed by statistical models starting from the probabilistic Latent Semantic Analysis (pLSA) model[2]. In pLSA, the focus is on the conditional probability $P(A_{ui}|u,i)$ that a user u rates an item i with rating A_{ui}. The fundamental idea is to derive a low-dimensional representation of the observed user-item ratings in terms of their affinity to hidden variables c[1]. The probability of co-occurrence is modeled as a mixture of conditionally independent multinomial distributions:

$$P(\theta; u, i) = \sum_c P(c)P(u|c)P(i|c) = P(u)\sum_c P(c|u)P(i|c) \tag{20}$$

where θ is a vector of the unknown parameters. Then, by incorporating a variational distribution $V(c; u, i)$[5] and defining a risk function $R(\theta, V)$, such that

$$R(\theta, V) = -\frac{1}{N}\sum_{(u,i)\in\Lambda}\sum_c V(c; u, i)(\log P(u|c) + \log P(c|i)) \tag{21}$$

the model maximizes the negative log-likelihood function:

$$L(\theta) = -\frac{1}{N}\sum_{(u,i)\in\Lambda} P(\theta; u, i) = -\frac{1}{N}\sum_{(u,i)\in\Lambda}\sum_c(\log P(u|c) + l\log P(c|i)) = -\frac{1}{N}\sum_{(u,i)\in\Lambda}\sum_c V(c; u, i)\left(\frac{\log(P(u|c)\log P(c|i))}{V(c; u, i)}\right) = R(\theta, V) - \frac{1}{N}H(V(c; u, i)) \tag{22}$$

where $H(V)$ is the entropy of variational distribution V.

In addition to pLSA, numerous probabilistic models can be used to predict user-item ratings, including Bayesian probabilistic matrix factorization[39], regression-based latent factor model[40], latent Dirichlet allocation[41], probabilistic factor analysis[42], and restricted Boltzmann machine[43]. However, these models are not covered in this paper. Interested readers can find the details in the above references.

3.11 Constraints on the completed matrix

Many applications require the completed matrix to have a certain property. For example, if the matrix to be completed is a covariance matrix, it is expected to be semi-positive definite. Moreover, the predicted negative value becomes meaningless in many applications. For example, in the user-item affinity prediction problem, it is difficult to explain the meaning of a predicted negative rating. The non-negative matrix completion intends to guarantee that all elements recovered are non-negatives. The non-negative matrix completion problem becomes a constraint satisfaction problem by adding the non-negative constraints:

$$\begin{aligned}&\min_R \frac{1}{2}\|P_\Lambda(R) - P_\Lambda(A)\|_{\mathrm{F}}^2 + \lambda_7\|R\|_*,\\ &\text{s.t., } R \geqslant 0\end{aligned} \tag{23}$$

Similarly, assuming A is an $n \times n$ symmetric matrix, the semi-positive definite constraint[44] can be imposed in a similar way, such that

$$\begin{aligned}&\min_R \frac{1}{2}\|P_\Lambda(R) - P_\Lambda(A)\|_{\mathrm{F}}^2 + \lambda_8\|R\|_*,\\ &\text{s.t., } R \succeq 0\end{aligned} \tag{24}$$

where $R \succeq 0$ indicates that R is semi-positive definite and λ_7 and λ_8 are regularization parameters.

4 Computational Algorithms for Matrix Completion in Recommendation Systems

In this section, considering the mathematical models described in Section 3, we review several popularly used recommendation algorithms based on matrix completion, including Alternative Least Square (ALS), spectral regularization with soft threshold, Alternating Direction Method of Multipliers (ADMM), Proximal Forward-Backward Splitting (PFBS), Singular Value Thresholding (SVT), Accelerated Proximal Gradient (APG), Fixed Point Continuation (FPC), nonlinear Successive Over-Relaxation (SOR), Stochastic Gradient Descent (SGD), and Expectation Maximization (EM) algorithms.

4.1 Alternative least square

The ALS algorithm is designed for the $l2$-regularized matrix factorization model (Formula (7)). However, because of the term $y_i^{\mathrm{T}}x_u$ for calculating R_{ui}, the objective function in Formula (7) is non-convex and optimizing Formula (7) is NP-hard. Nevertheless, if x_u can be fixed by treating its variables as constants, the minimization objective of Formula (7) would become a convex function of y_i[22]. Alternately, y_i can then be fixed by treating its variables as constants and then the objective of Formula (7) becomes a convex function of x_u. Therefore, in ALS, when one is fixed, the other is calculated, and this process is repeated until convergence is reached. This derivation process for the user vectors x_u for all u can be expressed as

$$x_u^{(j+1)} = Y^{(j)\mathrm{T}}\left(Y^{(j)}Y^{(j)\mathrm{T}} + \lambda_1 I_r\right)^{-1} x_u^{(j)},$$

and similarly, the process for calculating the item vectors y_i for all i is

$$y_i^{(j+1)} = X^{(j)\mathrm{T}}\left(X^{(j)}X^{(j)\mathrm{T}} + \lambda_1 I_r\right)^{-1} y_i^{(j)},$$

where I_r is an $r \times r$ identity matrix.

The benefit of using the ALS approach is that it can be computationally parallelized, since the calculation for each vector does not depend on the results of the other; therefore, it is an efficient optimization technique.

Replacing the Gram matrix XX^{T} in the ALS algorithm with a kernel $K(x_i{}^{\mathrm{T}}, x_j{}^{\mathrm{T}})$ function which measures the similarity between observation vectors may lead to better prediction results[27]. Paterek[27] reported that $K(x_i{}^{\mathrm{T}}, x_j{}^{\mathrm{T}}) = \mathrm{e}^{2(x_i{}^{\mathrm{T}} x_j - 1)}$ is a good choice. The ALS algorithm can also be accelerated by integrating with other approaches. For example, Hastie et al.[45] combined Soft-Impute and ALS algorithms to obtain a Soft-Impute-ALS algorithm which outperforms both.

4.2 Spectral regularization with soft threshold

The Soft-Impute algorithm[33] is designed for the spectral regularization model (Formula (14)) by replacing the unknown elements from a soft-thresholded SVD at every iteration step. Starting from an initial matrix $R^{(0)}$, Soft-Impute carries out the following iterations,

$$R^* \leftarrow P_\Lambda(A) + P_{\bar{\Lambda}}(R^{(j)}),$$
$$R^{(j+1)} \leftarrow D_\lambda(R^*),$$

where D_λ is the matrix shrinkage operator on threshold λ defined as the shrinkage of the singular values less than λ and their associated singular vectors, i.e.,

$$D_\lambda(R) = \sum_{k}^{\sigma_k \geqslant \lambda} (\sigma_k - \lambda) u_k v_k^{\mathrm{T}},$$

where σ_k is the k-th singular value of R, and u_k and v_k are the corresponding left and right singular vectors, respectively. In the Soft-Impute algorithm, $[P_\Lambda(A) - P_\Lambda(R^{(j)})] + R^{(j)}$ replaces $P_\Lambda(A) + P_{\bar{\Lambda}}(R^{(j)})$ during iterations such that the first part $[P_\Lambda(A) - P_\Lambda(R^{(j)})]$ is sparse and the second part $R^{(j)}$ is low-rank, which can be efficiently stored and manipulated. Moreover, partial SVD algorithms are used to fast-calculate the D_λ operator at each iteration step.

4.3 Proximal forward-backward splitting algorithm

The PFBS[46–49] is a soft-thresholding algorithm popularly used in signal analysis and image processing. Given the spectral regularization model (Formula (14)), the PFBS solution is formulated by the fixed point equation:

$$R = D_{\delta\lambda_6}(R + \delta P_\Lambda(A - R)),$$

for $\delta > 0$. Here, D_{λ_6} is the proximity operator of $\lambda_6 \|R\|_*$. Then, given $Y^{(0)}$ as the initial matrix, a simplified PFBS algorithm can be expressed using the following iteration steps:

$$R^{(j+1)} \leftarrow D_{\delta_j \lambda_6}(Y^{(j)}),$$
$$Y^{(j+1)} \leftarrow R^{(j+1)} + \delta_{j+1} P_\Lambda(A - R^{(j+1)}).$$

4.4 Alternating direction method of multipliers

The ADMM[50] algorithm adopts the form of a decomposition-coordination procedure to break an optimization problem into small local sub-problems and coordinates the solutions of these sub-problems to the global problem. The ADMM combines the advantages of dual decomposition and augmented Lagrangian methods for optimization problems.

The ADMM algorithm for matrix completion starts from the following model, which is equivalent to model (14) by introducing a separation matrix variable Y, such that

$$\min_R \frac{1}{2} P_\Lambda(Y) - P_\Lambda(A)\|_{\mathrm{F}}^2 + \lambda_6 \|R\|_*,$$
$$\text{s.t., } Y = R.$$

Then, the augmented Lagrangian function becomes

$$L(R, Y, Z) = \frac{1}{2}\|P_\Lambda(Y) - P_\Lambda(A)\|_{\mathrm{F}}^2 +$$
$$\lambda_6 \|R\|_* + \langle Z, Y - R \rangle + \frac{\rho}{2}\|Y - R\|_{\mathrm{F}}^2,$$

where $Z \in \mathbf{R}^{m \times n}$ is the Lagrange multiplier of the linear constraint, ρ is the penalty parameter for the violation of the linear constraint, and $\langle \cdot \rangle$ denotes the standard trace inner product. Applying the original ADMM[51] algorithm to the augmented Lagrangian function, the following iterative scheme can be obtained:

$$R^{(j+1)} \leftarrow \arg\min_X L(R, Y^{(j)}, Z^{(j)}),$$
$$Y^{(j+1)} \leftarrow \arg\min_Y L(R^{(j+1)}, Y, Z^{(j)}).$$

The updated Lagrange multiplier $Z^{(j+1)}$[52, 53] is generalized as

$$Z^{(j+1)} \leftarrow Z^{(j)} + \gamma\rho(Y^{(j+1)} - R^{(j+1)}),$$

where γ denotes the learning rate with a suggested range of $0 < \gamma < \frac{\sqrt{5}+1}{2}$. Here, $R^{(j+1)}$ can be obtained by applying the matrix shrinkage operator, i.e.,

$$R^{(j+1)} = D_{\frac{\lambda_6}{\rho}}\left(Y^{(j)} + \frac{Z^{(j)}}{\rho}\right),$$

and $Y^{(j+1)}$ can be obtained using the inverse operator:

$$Y^{(j+1)} = R^{(j+1)} \frac{Z^{(j)}}{\rho} + P_\Lambda\left(A - \left(R^{(j+1)} - \frac{Z^{(j)}}{\rho}\right)\right).$$

The ADMM algorithm is particularly suitable for handling matrix completion problems with additional constraints. Similar to the general matrix completion, ADMM have been used to address a model equivalent to the non-negative matrix completion model (Formula (23)) by introducing a separation matrix variable[54, 55]:

$$\min_R \frac{1}{2} P_\Lambda(Y) - P_\Lambda(A)\|_F^2 + \lambda_8 \|R\|_*,$$
$$\text{s.t., } Y = R \text{ and } Y \geqslant 0.$$

All iteration steps are similar to that of the general matrix completion problem excluding that for obtaining $Y^{(j+1)}$, such that

$$(Y^{(j+1)})_\Lambda = Q_+\left(\frac{1}{\rho+1} P_\Lambda(A + \rho R^{(j)} - Z^{(j)})\right),$$
$$(Y^{(j+1)})_{\bar{\Lambda}} = Q_+\left(P_{\bar{\Lambda}}\left(R^{(j)} - \frac{Z^{(j)}}{\rho}\right)\right),$$

where Q_+ is an operator that projects the parameter matrix X onto the non-negative space, such that

$$Q_+(X)_{ui} = \begin{cases} X_{ui}, & \text{if } X_{ui} < 0; \\ 0, & \text{otherwise.} \end{cases}$$

In the above method, Q_+ is computed to generate $Y^{(j+1)}$, which cannot strictly guarantee non-negative elements in $R^{(j+1)}$. Nevertheless, when an appropriate penalty parameter ρ is selected, $\|Y - R\|_F^2$ becomes small when convergence is approached, which can satisfy the non-negative requirements of many practical applications.

For the semi-positive definite matrix completion model (Formula (24))[44], $R \in S_+^n$ must be satisfied, where S_+^n denotes the cone (manifold) of positive semidefinite matrices in the space of symmetric $n \times n$ matrices. To satisfy this constraint, the iteration to obtain $R^{(j+1)}$[56] becomes

$$R^{(j+1)} = P_{S_+^n}\left(Y^{(j+1)} + \frac{Z^{(j)} - I_n}{\rho}\right),$$

where I_n is an $n \times n$ identity matrix, and $P_{S_+^n}$ is a projector operator, which is computed by carrying out an eigen decomposition on its parameter matrix and then eliminating the eigenvalues less than 0 and their corresponding eigenvectors.

4.5 Singular value thresholding

The SVT algorithm[35] is a first-order algorithm for solving the nuclear norm optimization problem using

$$\min_R \frac{1}{2}\|R\|_F^2 + \tau\|R\|_*,$$
$$\text{s.t., } R_{ui} = A_{ui}, (u,i) \in \Lambda$$

with a threshold parameter τ. An iterative gradient ascent approach formulated as Uzawa's algorithm[22] or linearized Bregman iterations[45] is applied, such that

$$R^{(i)} \leftarrow D_\tau(Y^{(i)}),$$
$$Y^{(i+1)} \leftarrow Y^{(i)} + \delta P_\Lambda(A - R^{(i)}).$$

Here, δ is the step size. Unlike the ADMM, PFBS, and Soft-Impute algorithms, which lead to solutions of spectral norm regularization model (Formula (14)), the SVT algorithm actually converges to the approximated solution of the nuclear norm minimization model (Formula (16)). This is because a very large τ value is usually picked so that the $\|R\|_*$ term dominates the $\frac{1}{2}\|R\|_F^2$ term in the minimization objective.

The SVT algorithm considers the global pattern of A and seeks a complete matrix X with minimized nuclear norm to recover the missing entries in A. However, it has a problem of computational cost, where the matrix shrinkage operator D_τ, which requires calculating the SVD to obtain the singular values and vectors of $Y^{(i)}$, is repeatedly computed at every iteration step. Cai and Osher[57] reformulated $D_\tau(Y^{(i)})$ by projecting $Y^{(i)}$ onto a 2-norm ball and then applying complete orthogonal decomposition and polar decomposition to the projection, which saves 50% or more computational time compared to the SVT implementation with full SVD. A more popular alternative strategy is to compute partial SVD for the singular values of interest. This is because only those singular values over τ are concerned in D_τ. The partial SVD implementations based on Krylov subspace algorithms, such as Lanczos algorithm with reorthogonalization, can efficiently accelerate the SVT algorithm if the number of singular values exceeding τ is significantly less than $\min(m,n)$. However, if this number gets over $0.2\min(m,n)$, the computational cost of partial SVD using Krylov subspace method can exceed that of full SVD[25]. Alternatively, recent studies show that the partial SVD calculation based on randomized SVD[58], rank revealing technique[59], single-pass SVD[60], and subspace reuse[61] can keep D_τ computation cost low throughout SVT iterations.

4.6 Accelerated proximal gradient

Toh and Yun[62] proposed an APG algorithm for the nuclear norm regularized model (Formula (14)). In the APG algorithm, for a given Y, a quadratic approximation of $\frac{1}{2}\|P_\Lambda(R) - P_\Lambda(A)\|_{\mathrm{F}}^2$ at Y is given such that

$$\frac{1}{2}\|P_\Lambda(R) - P_\Lambda(A)\|_{\mathrm{F}}^2 \approx \frac{1}{2}\|P_\Lambda(Y) - P_\Lambda(A)\|_{\mathrm{F}}^2 + \langle P_\Lambda(Y) - P_\Lambda(A), R - Y\rangle + \frac{1}{2\tau}\|R - Y\|_{\mathrm{F}}^2,$$

where $\tau > 0$ is a proximal parameter. Substituting the quadratic approximation in Formula (14), the minimization model becomes

$$\min_R \lambda_6\tau\|R\|_* + \frac{1}{2}\|R - (Y - \tau(P_\Lambda(Y) - P_\Lambda(A)))\|_{\mathrm{F}}^2.$$

Then, APG generates $(R^{(j)}, Y^{(j)}, t^{(j+1)})$ by the following iterative scheme:

$$Y^{(j)} \leftarrow R^{(j)} + \frac{t^{(j-1)} - 1}{t^{(j)}}(R^{(j)} - R^{(j-1)}),$$
$$R^{(j)} \leftarrow D_{\lambda_6\tau}(Y^{(j)} - \tau(P_\Lambda(Y^{(j)}) - P_\Lambda(A))),$$
$$t^{(j+1)} \leftarrow \frac{1 + \sqrt{1 + 4(t^{(j)})^2}}{2}.$$

Linear search-like, continuation, and truncation techniques have been applied to accelerate APG.

4.7 Fixed point continuation

Recently, Ma et al.[63] designed an FPC algorithm, which is a matrix extension of the fixed point iterative algorithm for the $l1$-regularized problem[49], to solve the nuclear norm regularized linear least squares problem (Formula (14)). The fundamental idea of the FPC algorithm is based on an operator splitting technique. As an extended result from Ref. [49], R^* is the optimal solution to Formula (14) if and only if

$$0 \in \lambda_6\partial\|R^*\|_* + P_\Lambda(R^*) - P_\Lambda(A).$$

Considering the following equivalent model,

$$0 \in \tau\lambda_6\partial\|R^*\|_* + R^* - \left(R^* - \tau\left(P_\Lambda(R^*) - P_\Lambda(A)\right)\right),$$

FPC applies operator splitting by setting

$$Y^* = R^* - \tau(P_\Lambda(R^*) - P_\Lambda(A)),$$

and the above model becomes

$$0 \in \tau\lambda_6\partial\|R^*\|_* + R^* - Y^*.$$

Then, $R^* = D_{\tau\lambda_6}(Y^*)$ is the optimal solution of

$$\min_R \tau\lambda_6\|R\|_* + \frac{1}{2}\|R - Y^*\|_{\mathrm{F}}^2.$$

This leads to the following FPC iteration scheme:

$$Y^{(j)} \leftarrow R^{(j)} - \tau(P_\Lambda(R^{(j)}) - P_\Lambda(A)),$$
$$R^{(j+1)} \leftarrow D_{\tau\lambda_6^{(j+1)}}(Y^{(j)}),$$
$$\lambda_6^{(j+1)} \leftarrow \max(\eta\lambda_6^{(j)}, \overline{\lambda_6}),$$

where the parameter $0 < \eta < 1$ specifies the reduction rate of λ_6 and $\overline{\lambda_6} > 0$. The global convergence of the FPC algorithm is also given in Ref. [63].

4.8 Nonlinear successive over-relaxation algorithm

Based on the matrix factorization model (Formula (17)), Wen et al.[38] addressed the matrix completion problem using a nonlinear SOR algorithm (LMaFit). The LMaFit algorithm introduces a Lagrange multiplier Δ, such that $\Delta = P_\Lambda(\Delta)$, to Formula (19) and obtains the Lagrange function:

$$L(X, Y, Z, \Delta) = \frac{1}{2}\|XY - Z\|_F^2 - \langle\Delta, P_\Lambda(Z) - P_\Lambda(A)\rangle.$$

Differentiating $L(X, Y, Z, \Delta)$ and introducing an SOR-style weight parameter ω, the LMaFit iterations are obtained such as

$$Z_\omega \leftarrow \omega Z^{(j)} + (1 - \omega)X^{(j)}Y^{(j)},$$
$$X(\omega) \leftarrow Z_\omega Y^{(j)\mathrm{T}}(Y^{(j)}Y^{(j)\mathrm{T}})^\dagger,$$
$$Y(\omega) \leftarrow (X(\omega)^{\mathrm{T}}X(\omega))^\dagger(X(\omega)^{\mathrm{T}}Z_\omega),$$
$$P_{\bar{\Lambda}}(Z(\omega)) \leftarrow P_{\bar{\Lambda}}(X(\omega)Y(\omega)),$$
$$P_\Lambda(Z(\omega)) \leftarrow P_\Lambda(A),$$

where $\dagger$ denotes the Moore-Penrose pseudo-inverse operation, and $\bar{\Lambda}$ is the complement of Λ. Then, the residual ratio

$$\gamma(\omega) = \frac{\|P_\Lambda(A - X(\omega)Y(\omega))\|_{\mathrm{F}}}{P_\Lambda(A - X^{(j)}Y^{(j)})\|_{\mathrm{F}}}$$

is monitored. If $\gamma(\omega) < 1$, $X(\omega), Y(\omega)$, and $Z(\omega)$ are taken as $X^{(j+1)}, Y^{(j+1)}, Z^{(j+1)}$, respectively, in the next iteration; otherwise, adjust ω accordingly. More details on setting ω can be found in Ref. [38]. At every LMaFit iteration, the computationally costly SVD operations are avoided and only least square operations are needed, which makes the LMaFit algorithm computationally efficient in large-scale matrix completion problems.

Notice that when $\omega = 1$, the SOR iteration is equivalent to the GaussSeidel (GS) iteration. Nevertheless, when ω is appropriately set, the SOR iterations in the LMaFit lead to significant convergence acceleration compared to GS iterations.

4.9 Stochastic gradient descent

The $l2$ matrix factorization regularization problem (Formula (8)) can be solved by SGD optimization[64], which iterates over all known ratings. For each $(u, i) \in \Lambda$, the prediction error e_{ui} is calculated as

$$e_{ui} = R_{ui} - \hat{R}_{ui}.$$

Then, the parameters b_u, b_i, x_u, and y_i are trained iteratively according to the opposite directions of the gradients:

$$b_u^{(j+1)} \leftarrow b_u^{(j)} + \gamma(e_{ui}^{(j)} - \lambda_2 b_u^{(j)}),$$
$$b_i^{(j+1)} \leftarrow b_i^{(j)} + \gamma(e_{ui}^{(j)} - \lambda_2 b_i^{(j)}),$$
$$x_u^{(j+1)} \leftarrow x_u^{(j)} + \gamma(e_{ui}^{(j)} y_i^{(j)} - \lambda_2 x_u^{(j)}),$$
$$y_i^{(j+1)} \leftarrow y_i^{(j)} + \gamma(e_{ui}^{(j)} x_u^{(j)} - \lambda_2 y_i^{(j)}),$$

where γ is the learning rate. Takacs et al.[65] extended the above model by dedicating different learning rates (γ) and regularization (λ) values for different learning parameters to obtain better accuracy.

In the SVD++ algorithm for Formula (11), the SGD iteration scheme accordingly becomes

$$b_u^{(j+1)} \leftarrow b_u^{(j)} + \gamma(e_{ui}^{(j)} - \lambda_3 b_u^{(j)}),$$
$$b_i^{(j+1)} \leftarrow b_i^{(j)} + \gamma(e_{ui}^{(j)} - \lambda_3 b_i^{(j)}),$$
$$x_u^{(j+1)} \leftarrow x_u^{(j)} + \gamma(e_{ui}^{(j)} y_i^{(j)} - \lambda_3 x_u^{(j)}),$$
$$y_i^{(j+1)} \leftarrow y_i^{(j)} + \gamma(e_{ui}^{(j)}(x_u^{(j)} + |W(u)|^{-\frac{1}{2}} \sum_{k\in R(u)} p_k^{(j)}) - \lambda_3 y_i^{(j)}),$$
$$p_l^{(j+1)} \leftarrow p_l^{(j)} + \gamma(e_{ui}^{(j)} |R(u)|^{-\frac{1}{2}} y_i^{(j)} - \lambda_3 p_l{}^{(j)}), \quad \text{for } \forall l \in R(u).$$

Compared to the SVD model, the SVD++ model often results in improved accuracy as it considers implicit feedback; however, the tradeoff is that there are significantly more parameters to train, which makes the SVD++ model difficult to scale to very large datasets.

The SGD optimization method can also be applied to the $l1$ matrix factorization regularization problem (Formula (11)) and the $l1/l2$ matrix factorization regularization problem (Formula (12)). Defining a vector sign function $\text{SGN}(x)$, such that

$$\text{SGN}(x) = \begin{bmatrix} \text{sgn}(x_1) \\ \vdots \\ \text{sgn}(x_n) \end{bmatrix},$$

where $\text{sgn}(\cdot)$ denotes the signum function for a scalar, for model (11), the iteration scheme for updating the latent factor vectors x_u and y_i becomes

$$x_u^{(j+1)} \leftarrow x_u^{(j)} + \gamma(e_{ui}^{(j)} y_i^{(j)} - \lambda_4 \text{SGN}(x_u^{(j)}))$$

and

$$y_i^{(j+1)} \leftarrow y_i^{(j)} + \gamma(e_{ui}^{(j)} x_u^{(j)} - \lambda_4 \text{SGN}(y_i^{(j)})),$$

respectively. For Formula (12), the iteration scheme for updating x_u and y_i then becomes

$$x_u^{(j+1)} \leftarrow x_u^{(j)} + \gamma(e_{ui}^{(j)} y_i^{(j)} - \frac{\lambda_5 \alpha}{2} \text{SGN}(x_u^{(j)}) - \lambda_5(1-\alpha) x_u^{(j)})$$

and

$$y_i^{(j+1)} \leftarrow y_i^{(j)} + \gamma(e_{ui}^{(j)} x_u^{(j)} - \frac{\lambda_5 \alpha}{2} \text{SGN}(y_i^{(j)}) - \lambda_5(1-\alpha) y_i^{(j)}),$$

respectively. The other iteration steps are similar to that of SGD for the $l2$-regularized matrix factorization model (Formula (8)).

4.10 Expectation maximization algorithm

The parameters of the probabilistic models, such as pLSA, are learned using the EM algorithm[4]. In the EM algorithm, the parameters are estimated iteratively, starting from an initial guess. Each iteration computes an Expectation (E) step and a Maximization (M) step in alternation[66]. The E-step uses the current estimate of the parameters to obtain the distribution for the unobserved variables, given the observed values of the known variables. The M-step re-estimates the model parameters to maximize the log-likelihood function.

In the pLSA model, at the j-th iteration, the E-step calculates the tightest upper bound given the current parameters $\theta^{(j)}$ with respect to the variational distribution $V^{(j)}$, such that

$$V^{(j+1)}(c,u,i;\theta^{(j)}) = P(c|u,i;\theta^{(j)}) = \frac{P(i|c;\theta^{(j)})P(c|u;\theta^{(j)})}{\sum_{c'} P(i|c';\theta^{(j)})P(c'|u;\theta^{(j)})}.$$

The upper bound of the negative log-likelihood function becomes

$$R(\theta^{(j+1)}, V^{(j+1)}) = -\frac{1}{N}\sum_{(u,i)\in\Lambda}\sum_{c} V^{(j+1)}(c;u,i;\theta^{(j)})(\log P(u|c) + \log P(c|i)).$$

The M-step then maximizes the above upper bound of the log-likelihood function $R(\theta^{(j+1)}, V^{(j+1)})$ with respect to $\theta^{(j+1)}$. The EM iterations are repeated until the likelihood improvement is smaller than a pre-determined threshold value.

The EM algorithm is often a non-convex optimization process. It has been shown that each EM iteration either improves the true likelihood or reaches the local maximum.

5 Applications for Recommendation Systems Using Matrix Completion

In addition to the usual applications of user-item association prediction, here we present other applications of recommendation systems based matrix completion.

5.1 Computational drug repositioning

Computational drug repositioning is an important and efficient approach to identify new treatments with known drugs. Luo et al.[8] modeled the drug repositioning problem as a recommendation system (DRRS) to discover new disease indications for drugs. In the DRRS, the related data sources and validated information of drugs and diseases are integrated to construct a heterogeneous drug-disease interaction network (Fig. 1). Then, the heterogeneous network is represented as a large adjacency matrix (Fig. 2) where the unknown drug-disease associations are presented as blank entries. A fast SVT algorithm[61] is used to complete the drug-disease adjacency matrix with predicted scores for unknown drug-disease pairs. The comprehensive experimental results show that the DRRS improves the prediction accuracy compared with the other state-of-the-art approaches in both system-wide and *de novo* predictions.

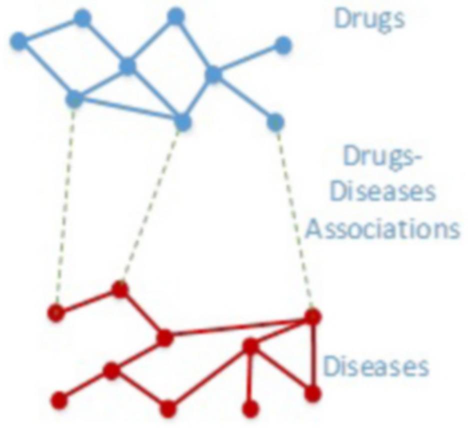


Fig. 1 Heterogeneous drugs-diseases network.

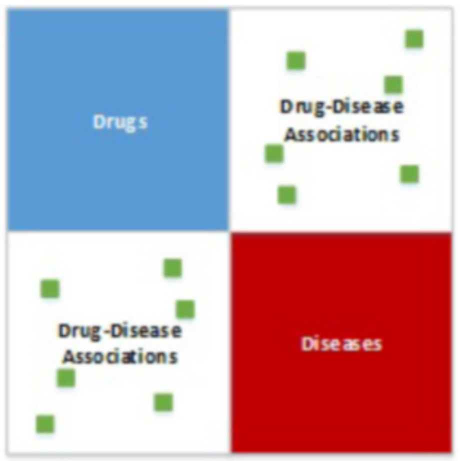


Fig. 2 Association matrix.

5.2 Sports game results predictions

The National Collegiate Athletic Association (NCAA) Men's Division I Basketball Tournament, commonly known as "March Madness", is one of the most popular sporting events in the United States. Every year, 68 out of 364 NCAA Division I teams are selected after the regular season to participate in a single elimination tournament for the NCAA men's basketball championship. By arranging every team on rows and columns, a matrix of games is displayed in Fig. 3, where a blue dot represents a game between two teams in the regular season.

Ji et al.[67, 68] employed matrix completion recommendation systems to predict the March Madness results. Game parameters, including field goals percentage, three pointers percentages, free throw percentages, offensive rebounds, defensive rebounds, assists, turnovers, steals, blocks, and fouls, were predicted by completing the game parameter matrices. These predicted parameters provided a predicted scenario of a game of two teams that have never met in the regular season. These parameters were fed to a neural network to finally predict the outcome of the March Madness playoff games. In 2015 March Madness, this method correctly predicted the outcomes of 49 out of 63 games.

5.3 Business to business electronic commerce

The use of recommendation systems toward electronic

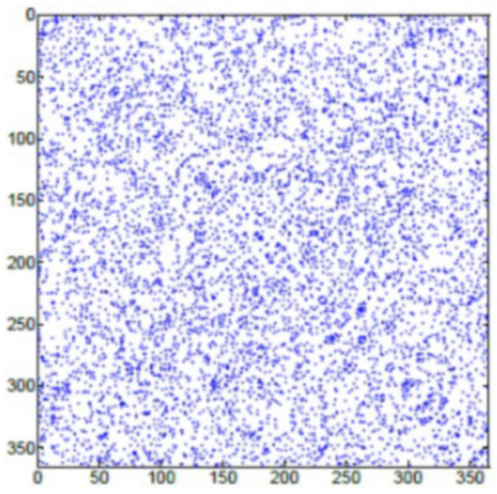


Fig. 3 Game matrix of 364 NCAA division I basketball teams. The *x*- and *y*-axes represent the NCAA teams and each point indicates there is a match between the two team during the regular season.

commercial or e-commerce applications focusing on Business-to-Customer (B2C) approaches has been discussed previously, such as the Netflix problem. These e-commerce B2C applications also include online retailers such as Amazon, Best Buy, Walmart, and most other corporations that dominate the retail industry. However, another application of recommendation systems used in commerce is Business to Business (B2B) transactions, where like those B2C online retailers, these B2B recommendation system users try to minimize the information overload and allow a computational algorithm to provide effective business intelligence.

The attributes of a typical B2B e-commerce recommendation system can be classified into the following main categories: system inputs, system processes, and system outputs. The system includes data collected from the business, which comprise industry specific conditions, supplier data, past and current customer activities, and customer ratings about goods and services[69]. In this way, the B2B recommender functions similarly to the content-based filtering approaches in B2C systems; however, they differ in their outputs, as the B2B system is not focused on delivering a computationally derived associated product to the customer, but on establishing links between a business and another stakeholder and identifying potential opportunities with other businesses. For instance, the system can use website browsing data and consequent purchases to evaluate advertising effectiveness, and then make recommendations for partnerships with marketing companies. Another approach could be in supply chain management, where the system makes recommendations for suppliers based on past performances of deliveries in terms of timeliness and quality, and the sales that resulted from the manufacturers. This data can help the business in negotiating prices, discovering opportunities, and evaluating the return-on-investment on any given decision.

5.4 Gene expression predictions

Over the last 50 years, one of the most dynamic fields of study in biomedical research has been investigations into protein folding and its effects on gene expression. The genomic manifestations of many human diseases and pathological conditions are related to protein folding[70]. This is a process that describes how a protein can exist in four possible states. The first state is the “unfolded state”, in which a protein has been assembled with all the proper chemical components, but is not functional. The second state is the “molten globule”, or partially folded state. The third state is the “native state”, in which the protein is folded into its proper three-dimensional structure and is biologically functional. The fourth possible state is the amyloid fibril state, in which the protein is misfolded and becomes deformed. These latter two states have captivated many biological scientists because their impacts on the expression of genes lead to neurodegenerative diseases, such as Alzheimer’s disease, Parkinson’s disease, Huntington’s disease, Bovine Spongiform Encephalopathy, and Rheumatoid Arthritis.

Advances in high performance computing have allowed researchers to investigate the effects of gene expression, and have led to the use of extremely large datasets to predict how genes are expressed based on their underlying protein structure. One such method is the use of low-rank matrix completion on known and sparse gene expression levels to recommend future gene expressions. A low-rank matrix is formed based on the underlying biological conditions. For instance, it is generally known that many genes interact with each other; therefore, interdependent factors contribute to the protein folding phases, leading to gene transcription, and ultimately gene expression, which can be characterized computationally in a correlated data matrix. Since gene expression values are likely to exist in a low-dimensional linear subspace, the resulting matrix can be considered as a low-rank matrix[71]. Then, the techniques discussed in the previous sections, such as the minimization of the nuclear norm can be applied to recover and complete the matrix, thereby yielding a prediction on the final gene.

5.5 Microblogging recommendations

In a digital age where many people across the globe get their information from social networking platforms, the popular “microblogging” site Tumblr where users can share short messages to a wide audience can employ the advantages of recommendation systems to allow users find other similar messages or microblogs. Since message posts are generally short, a large number of such messages are generated every day, leading to mass amounts of dynamic text data, in addition to images. However, unlike other collaborative filtering approaches where users rank preferences, with Tumblr the ratings are in a more binary form; users simply chose to follow or not follow a post. However, this can be simplified by incorporating users activities and the contents of their posts which can include a combination of text, tags, or images[72]. These activities are then analyzed using machine learning techniques, such as a convolutional neural network, where all relevant features from the vast datasets can be obtained. Additionally, the features can be examined by employing a second neural network known as "word2vec", which transforms text data into a vector where words in similar contexts are closer to each other with multiple degrees of similarity[73]. Ultimately, the missing information from users who do not follow other users can essentially be supplanted by the activities they have performed in their own posts. By incorporating features from users, the matrix completion models can be used to make recommendations in the inductive setting, where predictions can be made for users not present in the training data set.

6 Conclusion

Matrix completion approaches have become important methodologies in recommendation systems, which are often more accurate than the nearest-neighbor approaches. Motivated by the famous Netflix Prize problem, many recommendation system models have been proposed and many computational algorithms have been accordingly developed. This survey aims to provide a comprehensive review of the matrix completion models and algorithms for recommendation systems, although it is unlikely to cover all models and algorithms available.

There have been quite a few research directions that go beyond the recommendation systems based on matrix completion. In reality, the popularity of an item may change over time. This can be solved by incorporating temporal dynamics information into the recommendation model. For example, Koren and Bell[64] proposed models that incorporate time-changing factors to gain insight of how the influences of two items rated by the same user decay over time. In fact, a more general problem of matrix completion is tenor completion, which is related to recovering missing values in high-dimensional data. Liu et al.[74] defined trace norm for tensors and extended the nuclear (trace) norm minimization model for tensor completion. Moreover, the traditional recommendation systems focus on prediction accuracy only. Nevertheless, in practical applications, objectives such as diversity and novelty are also important, although these may conflict with accuracy[75]. Hence, multi-objective optimization algorithms[76–78] are needed to find recommendations with respect to the tradeoffs among conflicting objectives. Furthermore, with the development of modern parallel and distributed computing architectures, much effort has been put on designing efficient parallel algorithms[79, 80] to enable matrix completion techniques make efficient recommendations for large-scale datasets.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (Nos. 61728211 and 1066471).

References

[1] T. Hofmann, Latent semantic models for collaborative filtering, *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 89–115, 2004.

[2] T. Hofmann, Unsupervised learning by probabilistic latent semantic analysis, *Mach. Learn.*, vol. 42, nos. 1&2, pp. 177–196, 2001.

[3] C. D. Charalambous and A. Logothetis, Maximum likelihood parameter estimation from incomplete data via the sensitivity equations: The continuous-time case, *IEEE Trans. Automat. Control*, vol. 45, no. 5, pp. 928–934, 2000.

[4] A. P. Dempster, N. M. Laird, and D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, *J. Roy. Stat. Soc. Ser B Methodol.*, vol. 39, no. 1, pp. 1–38, 1977.

[5] R. M. Neal and G. E. Hinton, *A View of the EM Algorithm That Justifies Incremental, Sparse, and Other Variants*. Norwell, MA, USA: Kluwer, 1998.

[6] H. T. Zhu, Z. Khondker, Z. H. Lu, and J. G. Ibrahim, Bayesian generalized low rank regression models for neuroimaging phenotypes and genetic markers, *J. Am. Stat. Assoc.*, vol. 109, no. 507, pp. 977–990, 2014.

[7] S. N. Wood, Low-rank scale-invariant tensor product smooths for generalized additive mixed models, *Biometrics*, vol. 62, no. 4, pp. 1025–1036, 2006.

[8] H. M. Luo, M. Li, S. K. Wang, Q. Liu, Y. H. Li, and J. X. Wang, Computational drug repositioning using low-rank matrix approximation and randomized algorithms, *Bioinformatics*, vol. 34, no. 11, 1904–1912, 2018.

[9] C. Q. Lu, M. Y. Yang, F. Luo, F. X. Wu, M. Li, Y. Pan, Y. H. Li, and J. X. Wang, Prediction of lncRNA-disease associations based on inductive matrix completion, *Bioinformatics*, doi:10.1093/bioinformatics/bty327.

[10] Y. Liang, D. L. Wu, G. R. Liu, Y. H. Li, C. L. Gao, Z. J. Ma, and W. D. Wu, Big data-enabled multiscale serviceability analysis for aging bridges, *Digit. Commun. Netw.*, vol. 2, no. 3, pp. 97–107, 2016.

[11] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, Recommender systems survey, *Knowl. Based Syst.*, vol. 46, pp. 109–132, 2013.

[12] M. Kunaver and T. Požrl, Diversity in recommender systems — A survey, *Knowl. Based Syst.*, vol. 123, pp. 154–162, 2017.

[13] R. Burke, Hybrid recommender systems: Survey and experiments, *User Model. User-Adapt. Interact.*, vol. 12, no. 4, pp. 331–370, 2002.

[14] C. Desrosiers and G. Karypis, A comprehensive survey of neighborhood-based recommendation methods, in *Recommender Systems Handbook*. Springer, 2010, pp. 107–144.

[15] C. He, D. Parra, and K. Verbert, Interactive recommender systems: A survey of the state of the art and future research challenges and opportunities, *Expert Syst. Appl.*, vol. 56, pp. 9–27, 2016.

[16] P. G. Campos, F. Díez, and I. Cantador, Time-aware recommender systems: A comprehensive survey and analysis of existing evaluation protocols, *User Model. User-Adapt. Interact.*, vol. 24, no. 1–2, pp. 67–119, 2014.

[17] X. W. Yang, Y. Guo, Y. Liu, and H. Steck, A survey of collaborative filtering based social recommender systems, *Comput. Commun.*, vol. 41, pp. 1–10, 2014.

[18] A. Klašnja-Milicevic, M. Ivanovic, and A. Nanopoulos, Recommender systems in e-learning environments: A survey of the state-of-the-art and possible extensions, *Artif. Intell. Rev.*, vol. 44, no. 4, pp. 571–604, 2015.

[19] R. Yera and L. Martínez, Fuzzy tools in recommender systems: A survey, *Int. J. Comput. Intell. Syst.*, vol. 10, no. 1, pp. 776–803, 2017.

[20] D. Kotkov, S. Q. Wang, and J. Veijalainen, A survey of serendipity in recommender systems, *Knowl. Based Syst.*, vol. 111, pp. 180–192, 2016.

[21] E. J. Candes and T. Tao, The power of convex relaxation: Near-optimal matrix completion, *IEEE Trans. Inf. Theory*, vol. 56, no. 5, pp. 2053–2080, 2010.

[22] M. Udell, C. Horn, R. Zadeh, and S. Boyd, Generalized low rank models, *Found. Trends® Mach. Learn.*, vol. 9, no. 1, pp. 1–118, 2016.

[23] Y. Koren, R. Bell, and C. Volinsky, Matrix factorization techniques for recommender systems, *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[24] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. T. Riedl, Application of dimensionality reduction in recommender system-a case study, in *Proc. ACM WebKDD Web Mining for E-Commerce Workshop*, 2000.

[25] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, Incremental singular value decomposition algorithms for highly scalable recommender systems, in *Proc. 6th Int. Conf. on Computers and Information Technology*, 2002.

[26] D. Billsus and M. J. Pazzani, Learning collaborative information filters, in *Proc. 15th Int. Conf. on Machine Learning*, San Francisco, CA, USA: ACM, 1998.

[27] A. Paterek, Improving regularized singular value decomposition for collaborative filtering, in *Proc. KDD and Workshop*, San Jose, CA, USA, 2007.

[28] J. D. M. Rennie and N. Srebro, Fast maximum margin matrix factorization for collaborative prediction, in *Proc. 22nd Int. Conf. on Machine Learning*, Bonn, Germany, 2005.

[29] M. G. Vozalis and K. G. Margaritis, Using SVD and demographic data for the enhancement of generalized collaborative filtering, *Inf. Sci.*, vol. 177, no. 15, pp. 3017–3037, 2007.

[30] Y. Koren, Factorization meets the neighborhood: A multifaceted collaborative filtering model, in *Proc. 14th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Las Vegas, NV, USA, 2008.

[31] B. Hallinan and T. Striphas, Recommended for you: The NETflix prize and the production of algorithmic culture, *New Media Soc.*, vol. 18, no. 1, pp. 117–137, 2016.

[32] Y. C. Ji, W. X. Hong, Y. L. Shangguan, H. Wang, and J. Ma, Regularized singular value decomposition in news recommendation system, in *Proc. 11th Int. Conf. on Computer Science & Education*, Nagoya, Japan, 2016, pp. 621–626.

[33] R. Mazumder, T. Hastie, and R. Tibshirani, Spectral regularization algorithms for learning large incomplete matrices, *J. Mach. Learn. Res.*, vol. 11, pp. 2287–2322, 2010.

[34] M. Kagie, M. van der Loos, and M. van Wezel, Including item characteristics in the probabilistic latent semantic analysis model for collaborative filtering, *AI Commun.*, vol. 22, no. 4, pp. 249–265, 2009.

[35] J. F. Cai, E. J. Candes, and Z. W. Shen, A singular value thresholding algorithm for matrix completion, *SIAM J. Optim.*, vol. 20, no. 4, pp. 1956–1982, 2010.

[36] E. Candès and B. Recht, Simple bounds for recovering low-complexity models, *Math. Program.*, vol. 141, nos. 1&2, pp. 577–589, 2013.

[37] B. Recht, M. Fazel, and P. A. Parrilo, Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization, *SIAM Rev*, vol. 52, no. 3, pp. 471–501, 2010.

[38] Z. W. Wen, W. T. Yin, and Y. Zhang, Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm, *Math. Program. Comput.*, vol. 4, no. 4, pp. 333–361, 2012.

[39] R. Salakhutdinov and A. Mnih, Bayesian probabilistic matrix factorization using Markov chain Monte Carlo, in *Proc. 25th Int. Conf. on Machine Learning*, Helsinki, Finland, 2008.

[40] D. Agarwal and B. C. Chen, Regression-based latent factor models, in *Proc. 15th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, Paris, France, 2009.

[41] D. M. Blei, A. Y. Ng, and M. I. Jordan, Latent Dirichlet allocation, *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[42] J. Canny, Collaborative filtering with privacy via factor analysis, in *Proc. 25th Annu. Int. ACM SIGIR Conf. on Research and Development in Information Retrieval*, Tampere, Finland, 2002.

[43] R. R. Salakhutdinov, A. Mnih, and G. E. Hinton, Restricted boltzmann machines for collaborative filtering, in *Proc. 24th Int. Conf. on Machine Learning*, Corvallis, OR, USA, 2007.

[44] F. F. Xu and P. Pan, A new algorithm for positive semidefinite matrix completion, *J. Appl. Math.*, vol. 2016, p. 1659019, 2016.

[45] T. Hastie, R. Mazumder, J. D. Lee, and R. Zadeh, Matrix completion and low-rank svd via fast alternating least squares, *J. Mach. Learn. Res.*, vol. 16, no. 1, pp. 3367–3402, 2015.

[46] J. F. Cai, R. H. Chan, and Z. W. Shen, A framelet-based image inpainting algorithm, *Appl. Computat. Harmon. Anal.*, vol. 24, no. 2, pp. 131–149, 2008.

[47] P. L. Combettes and V. R. Wajs, Signal recovery by proximal forward-backward splitting, *Multiscale Model. Simul.*, vol. 4, no. 4, pp. 1168–1200, 2005.

[48] I. Daubechies, M. Defrise, and C. De Mol, An iterative thresholding algorithm for linear inverse problems with a sparsity constraint, *Commun. Pure Appl. Math.*, vol. 57, no. 11, pp. 1413–1457, 2004.

[49] E. T. Hale, W. T. Yin, and Y. Zhang, Fixed-point continuation for ℓ_1-minimization: Methodology and convergence, *SIAM J Optim.*, vol. 19, no. 3, pp. 1107–1130, 2008.

[50] B. S. He, H. Yang, and S. L. Wang, Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities, *J. Optim. Theory Appl.*, vol. 106, no. 2, pp. 337–356, 2000.

[51] D. Gabay and B. Mercier, A dual algorithm for the solution of nonlinear variational problems via finite element approximation, *Comput. Math. Appl.*, vol. 2, no. 1, pp. 17–40, 1976.

[52] R. Glowinski, *Numerical Methods for Nonlinear Variational Problems*. Springer, 1984.

[53] R. Glowinski and P. Le Tallec, *Augmented Lagrangian and Operator Splitting Methods in Nonlinear Mechanics*. Philadelphia, PA, USA: SIAM, 1989, pp. 9.

[54] F. Xu and G. He, New algorithms for nonnegative matrix completion, *Pac. J. Optim.*, vol. 11, no. 3, pp. 459–469, 2015.

[55] Y. Y. Xu, W. T. Yin, Z. W. Wen, and Y. Zhang, An alternating direction algorithm for matrix completion with nonnegative factors, *Front. Math. China*, vol. 7, no. 2, pp. 365–384, 2012.

[56] C. H. Chen, B. S. He, and X. M. Yuan, Matrix completion via an alternating direction method, *IMA J. Numer. Anal.*, vol. 32, no. 1, pp. 227–245, 2012.

[57] J. F. Cai and S. Osher, Fast singular value thresholding without singular value decomposition, *Methods Appl. Anal.*, vol. 20, no. 4, pp. 335–352, 2013.

[58] N. Halko, P. G. Martinsson, and J. A. Tropp, Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions, *SIAM Rev.*, vol. 53, no. 2, pp. 217–288, 2011.

[59] H. Ji, W. J. Yu, and Y. H. Li, A rank revealing randomized singular value decomposition (R3SVD) algorithm for low-rank matrix approximations, arXiv: 1605.08134, 2016.

[60] W. J. Yu, Y. Gu, J. Li, S. H. Liu, and Y. H. Li, Single-pass PCA of large high-dimensional data, in *Proc. 26th Int. Joint Conf. on Artificial Intelligence*, Catalina Island, CA, USA, 2010.

[61] Y. H. Li and W. J. Yu, A fast implementation of singular value thresholding algorithm using recycling rank revealing randomized singular value decomposition, arXiv: 1704.05528, 2017.

[62] K. C. Toh and S. Yun, An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems, *Pac. J. Optim.*, vol. 6, no. 3, pp. 615–640, 2010.

[63] S. Q. Ma, D. Goldfarb, and L. F. Chen, Fixed point and bregman iterative methods for matrix rank minimization, *Math. Program.*, vol. 128, no. 1–2, pp. 321–353, 2011.

[64] Y. Koren and R. Bell, Advances in collaborative filtering, in *Recommender Systems Handbook*, F. Ricci, L. Rokach, B. Shapira, and P. B. Kantor, eds. Springer, 2011.

[65] G. Takács, I. Pilászy, B. Németh, and D. Tikk, Matrix factorization and neighbor based algorithms for the NETflix prize problem, in *Proc.* 2008 *ACM Conf. on Recommender Systems*, Lausanne, Switzerland, 2008, pp. 267–274.

[66] C. B. Do and S. Batzoglou, What is the expectation maximization algorithm? *Nat. Biotechnol.*, vol. 26, no. 8, pp. 897–899, 2008.

[67] H. Ji, E. O'Saben, A. Boudion, and Y. H. Li, March madness prediction: A matrix completion approach, in *Proc. Modeling, Simulation, and Visualization Student Capstone Conf.*, Suffolk, UA, USA, 2015.

[68] H. Ji, E. O'Saben, R. Lambi, and Y. H. Li, Matrix completion based model v2.0: Predicting the winning probabilities of march madness matches, in *Proc. Modeling, Simulation, and Visualization Student Capstone Conf.*, Suffolk, VA, USA, 2016.

[69] X. R. Zhang and H. S. Wang, Study on recommender systems for business-to-business electronic commerce, *Commun. IIMA*, vol. 5, no. 4, pp. 53–61, 2005.

[70] T. P. Exarchos, C. Papaloukas, C. Lampros, and D. I. Fotiadis, Mining sequential patterns for protein fold recognition, *J. Biomed. Inf.*, vol. 41, no. 1, pp. 165–179, 2008.

[71] A. Kapur, K. Marwah, and G. Alterovitz, Gene expression prediction using low-rank matrix completion, *BMC Bioinform.*, vol. 17, pp. 243, 2016.

[72] D. Shin, S. Cetintas, K. C. Lee, and I. S. Dhillon, Tumblr blog recommendation with boosted inductive matrix completion, in *Proc. 24th ACM Int. on Conf. on Information and Knowledge Management*, Melbourne, Australia, 2015.

[73] T. Mikolov, K. Chen, G. Corrado, and J. Dean, Efficient estimation of word representations in vector space, in *Proc. Workshop at Int. Conf. on Learning Representations*, Scottsdale, AZ, USA, 2013.

[74] J. Liu, P. Musialski, P. Wonka, and J. P. Ye, Tensor completion for estimating missing values in visual data, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 208–220, 2013.

[75] L. Z. Cui, P. Ou, X. H. Fu, Z. K. Wen, and N. Lu, A novel multi-objective evolutionary algorithm for recommendation systems, *J. Parallel Distrib. Comput.*, vol. 103, pp. 53–63, 2017.

[76] K. Deb, Multi-objective optimization. in *Search Methodologies*, E. K. Burke and G. Kendall, eds. Springer, 2005.

[77] Y. H. Li, MOMCMC: An efficient Monte Carlo method for multi-objective sampling over real parameter space, *Comput. Math. Appl.*, vol. 64, no. 11, pp. 3542–3556, 2012.

[78] W. H. Zhu, A. Yaseen, and Y. H. Li, DEMCMC-GPU: An efficient multi-objective optimization method with GPU acceleration on the fermi architecture, *New Generat. Comput.*, vol. 29, no. 2, pp. 163–184, 2011.

[79] B. Recht and C. Ré, Parallel stochastic gradient algorithms for large-scale matrix completion, *Math. Program. Comput.*, vol. 5, no. 2, pp. 201–226, 2013.

[80] Y. Y. Xu, R. R. Hao, W. T. Yin, and Z. X. Su, Parallel matrix factorization for low-rank tensor completion, *Inverse Probl. Imaging*, vol. 9, no. 2, pp. 601–624, 2015.

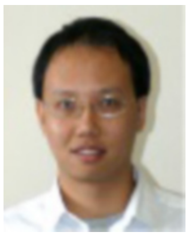
Andy Ramlatchan is a PhD student in Computer Science at Old Dominion University in Norfolk, VA, USA. He has worked for the US government for several years in multiple capacities where he leveraged big data and machine learning for various mission support programs. He is currently a senior computer scientist at NASA Langley Research Center in Hampton, VA, USA. His main research interests include matrix factorization and tensor completion for high dimensionality multi-modal sensor data.

Mengyun Yang received the BS degree in mathematics from Shaoyang University in 2007, and the MS degree in computational mathematics from Hunan Normal University in 2012. He is a lecturer in Shaoyang University and a PhD candidate at Central South University, Hunan, China. His current research interests include matrix completion and bioinformatics.

Jianxin Wang received the BEng and MEng degrees in computer engineering from Central South University, China, in 1992 and 1996, respectively, and the PhD degree in computer science from Central South University, China, in 2001. He is the vice dean and a professor in Department of Computer Science, Central South University, Changsha, China. His current research interests include algorithm analysis and optimization, parameterized algorithm, bioinformatics, and computer network. He is a member of the IEEE.

Quan Liu is a master student at Central South University. His main research interests include machine learning, recommender systems, and statistical association study.

Min Li received the PhD degree in computer science from Central South University, China, in 2008. She is currently a professor at the School of Information Science and Engineering, Central South University, Changsha, China. Her main research interests include bioinformatics and systems biology.



Yaohang Li received the BS degree from South China University of Technology in 1997, and the MS and PhD degrees in computer science from Florida State University, Tallahassee, FL, USA, in 2000 and 2003, respectively. He is an associate professor in computer science at Old Dominion University, Norfolk, VA, USA. His research interests are in protein structure modeling, computational biology, bioinformatics, Monte Carlo methods, big data algorithms, and parallel and distributive computing.