EAI Endorsed Transactions
on Collaborative Computing

Research Article ICST.ORG

Automated Dimension Determination for NMF-based Incremental Collaborative Filtering

Xiwei Wang[1,*], Jun Zhang[2], and Ruxin Dai[3]

[1]Department of Computer Science, Northeastern Illinois University, Chicago, Illinois 60625, USA
[2]Department of Computer Science, University of Kentucky, Lexington, Kentucky 40506-0633, USA
[3]Department of Computer Science and Information Systems, University of Wisconsin River Falls, River Falls, Wisconsin 54022, USA

Abstract

The nonnegative matrix factorization (NMF) based collaborative filtering t e chniques h a ve a c hieved great success in product recommendations. It is well known that in NMF, the dimensions of the factor matrices have to be determined in advance. Moreover, data is growing fast; thus in some cases, the dimensions need to be changed to reduce the approximation error. The recommender systems should be capable of updating new data in a timely manner without sacrificing the prediction accuracy.

In this paper, we propose an NMF based data update approach with automated dimension determination for collaborative filtering purposes. The approach can determine the dimensions of the factor matrices and update them automatically. It exploits the nearest neighborhood based clustering algorithm to cluster users and items according to their auxiliary information, and uses the clusters as the constraints in NMF. The dimensions of the factor matrices are associated with the cluster quantities. When new data becomes available, the incremental clustering algorithm determines whether to increase the number of clusters or merge the existing clusters. Experiments on three different datasets (MovieLens, Sushi, and LibimSeTi) were conducted to examine the proposed approach. The results show that our approach can update the data quickly and provide encouraging prediction accuracy.

Received on 27 December 2014; accepted on 30 June 2015; published on 17 December 2015

Keywords: auxiliary information, incremental clustering, data growth, collaborative Filtering, NMF

doi:10.4108/eai.17-12-2015.150804

1. Introduction

The advent of the Internet has generated exponential growth of various kinds of data. For average people, easy-to-use tools are highly desired to retrieve useful information that is beneficial to their daily life. In eCommerce, a large number of online shopping websites employ recommender systems to make personalized product recommendations. In general, a recommender system is a program that utilizes algorithms to predict users' preferences by profiling their shopping patterns. With the help of recommender systems, online merchants could better sell their products to the users who have visited their websites in the past. Collaborative filtering (CF) techniques are one of the most popular fundamental algorithms in recommender systems. CF aims at predicting users' preferences based on their transaction history and/or their feedback on products. There are different types of CF techniques, e.g., item/user correlation based CF's [16], singular value decomposition (SVD) based latent factor CF's [17], and nonnegative matrix factorization (NMF) based CF's [3][23].

One of the critical components of a recommender system is the user data. CF techniques require at least users' rating data, which reflects their preferences over products, to make the predictions. In addition, the auxiliary information associated with users and items is also taken into account by some CF models. Chen et al. [4] proposed a toolkit for the feature

*Corresponding author. Email: xwang9@neiu.edu

based collaborative filtering, named SVDFeature. They presented an example of using the toolkit. In their example, the album information and the temporal information are treated as auxiliary information in their SVDFeature for better prediction. Gu et al. [6] incorporated user and item graphs into an NMF based CF algorithm to improve the prediction accuracy. It is known that in some datasets, e.g., the MovieLens dataset [17], the Sushi preference dataset [10], and the LibimSeTi dating agency dataset [2], auxiliary information such as users' demographic data and items' category data, are also provided. This information, if properly used, can improve the recommendation accuracy, especially when the original rating matrix is extremely incomplete.

Furthermore, CF algorithms must be able to handle the fast data growth efficiently. In general, data grows in two aspects: new items/users with their transaction or rating data and the accompanying auxiliary information. The algorithms need to update the data and provide recommendations in a timely manner. Additionally, the matrix factorization based collaborative filtering algorithms require the dimensions of the factor matrices to be set in advance. When new data becomes available, the dimensions need to be updated.

In this paper, we propose an NMF based data update approach with automated dimension determination for collaborative filtering purposes. The approach, named iCluster-NMF, is based on the incremental clustering algorithm and the incremental nonnegative matrix tri-factorization (NMTF) [5]. It can determine the dimensions of the factor matrices and update them automatically. It exploits the nearest neighborhood based clustering algorithm to cluster users and items according to their auxiliary information, and uses the clusters as the constraints in NMF. The dimensions of the factor matrices are associated with the cluster quantities. When new data arrives, the incremental clustering algorithm determines whether to increase the number of clusters or merge the existing clusters. We examine our approach on previously mentioned three datasets in three aspects: (1) the correctness of the approximated rating matrix, (2) the time cost of the algorithms, and (3) the number of clusters produced by the approach. The results show that our approach can update the data quickly and provide satisfactory prediction accuracy.

The contributions of this paper are twofold: (1) utilizing auxiliary information as the constraints in NMTF for data approximation; (2) incorporating the incremental clustering technique into NMTF to automatically determine the dimensions of the factor matrices.

The remainder of this paper is organized as follows. Section 2 presents the related work. Section 3 defines the problem and related notations. Sections 4 and 5 describe the main idea of the proposed approach as well as a comparison model. Section 6 studies the experiments and discusses the results. Some concluding remarks and future work are given in 7.

2. Related Work

With the increasing popularity of online applications, the problem of managing fast growing data has become one of the major research topics in data science. The emergence of eCommerce has greatly facilitated people's life and expedited the daily purchases. With a large amount of new data arriving, the recommender systems employed by online merchants have to update and process the data efficiently. In [1], Brand proposed update rules for adding data to a "thin" SVD data model, which is used to update new data into the lightweight recommender systems. Wang and Zhang [20] incorporated the missing value imputation and the randomization based perturbation into incremental SVD for privacy preserving collaborative filtering data update. Wang et al. [21] proposed a swarm intelligence based recommendation algorithm, named Ant Collaborative Filtering, to capture the evolution of user preference over time. By doing so, the new data can be dynamically updated online.

Our proposed approach uses NMF as the fundamental technique for the data update. In [23], Zhang et al. applied NMF to collaborative filtering to learn the missing values in the rating matrix. They treated NMF as a solution to the expectation maximization (EM) problems. Chen et al. [3] proposed an orthogonal nonnegative matrix tri-factorization (ONMTF) [5] based collaborative filtering algorithm. Their algorithm also takes into account the user similarity and item similarity. Nirmal et al. [19] proposed explicit incorporation of the additional constraint, called the "clustering constraint", into NMF in order to suppress the data patterns in the process of performing the matrix factorization. Their work is based on the idea that one of the factor matrices in NMF contains cluster membership indicators. The clustering constraint is another indicator matrix with altered class membership in it. This constraint then guides NMF in updating factor matrices. Based on this idea, the proposed model applies the user and item cluster membership indicators to nonnegative matrix tri-factorization (NMTF), which results in better imputation of the missing values.

With regard to the clustering algorithms, K-Means [14] is a popular and well studied approach that is easy to implement and is widely used in many domains. As the name of the algorithm indicates, K-Means needs the definition of "mean" prior to clustering. It minimizes a cost function by calculating the means of clusters. This makes K-Means most suitable for continuous numerical data. When given categorical data such as users'

demographic data and movies' genre information, K-Means needs a pre-processing phase to make the data suitable for clustering. Huang [7] proposed a K-Modes clustering algorithm to extend the K-Means paradigm to categorical domains. Their algorithm introduces new dissimilarity measures to handle categorical objects and replaces means of clusters with modes. Additionally, a frequency based method is used to update modes in the clustering process so that the clustering cost function is minimized. In 2005, Huang et al. [8] further applied a new dissimilarity measure to the K-Modes clustering algorithm to improve its clustering accuracy.

The fast data growth requires the clustering algorithms to update the clusters constantly. The number of clusters might be increased or decreased. Su et al. [18] proposed a fast incremental clustering algorithm by changing the radius threshold value dynamically. Their algorithm restricts the number of the final clusters and reads the original dataset only once. It also considers the frequency information of the attribute values in the inter-cluster dissimilarity measure. Our approach adopts their clustering algorithm with some modifications. As stated previously, it is known that the NMF based collaborative filtering algorithms need to determine the dimensions of the factor matrices and update them when necessary. It is not convenient for people to manually specify these values and the automated decision making is highly desired. To this purpose, the proposed method determines the number of clusters by an incremental clustering algorithm and uses them as the dimensions in NMF.

3. Problem Description

Assume the data owner has three matrices: an incomplete user-item rating matrix $R \in \mathbb{R}^{m \times n}$, a user feature matrix $F_U \in \mathbb{R}^{m \times k_U}$, and an item feature matrix $F_I \in \mathbb{R}^{n \times k_I}$, where there are m users, n items, k_U user features, and k_I item features. An entry r_{ij} in R represents the rating left on item j by user i. The approximated matrix, denoted by $R_r \in \mathbb{R}^{m \times n}$ is the one that has all unknown values predicted in it.

When new users' ratings arrive, the new rows, denoted by $T \in \mathbb{R}^{p \times n}$, should be appended to the original matrix R. Meanwhile, their auxiliary information is also available, and thus the feature matrix is updated as well, i.e.,

$$\begin{bmatrix} R \\ T \end{bmatrix} \rightarrow R', \qquad \begin{bmatrix} F_U \\ \Delta F_U \end{bmatrix} \rightarrow F'_U \tag{1}$$

where $\Delta F_U \in \mathbb{R}^{p \times k_U}$.

Similarly, when new items become available, the new columns, denoted by $G \in \mathbb{R}^{m \times q}$, should be appended to the original matrix R, so should the item feature matrix, i.e.,

$$\begin{bmatrix} R & G \end{bmatrix} \rightarrow R'', \qquad \begin{bmatrix} F_I \\ \Delta F_I \end{bmatrix} \rightarrow F'_I \tag{2}$$

where $\Delta F_I \in \mathbb{R}^{q \times k_I}$.

4. iCluster-NMF Data Update

In this section, we will introduce the iCluster-NMF algorithm and its application in collaborative filtering data update.

4.1. Cluster-NMF

While iCluster-NMF handles the incremental data update, it is necessary to present the non-incremental version, named Cluster-NMF beforehand. This section is organized as follows: developing the objective function, deriving the update formulas, and the detailed algorithms.

Objective Function. Nonnegative matrix factorization (NMF) [13] is a widely used dimension reduction method in many applications such as clustering [5][11], text mining [22][15], data distortion based privacy preservation [9][19], etc. NMF is also applied in collaborative filtering to make product recommendations [23][3]. However, in CF data, a single user may have rated only a few items and one item may get only a small number of ratings. Therefore, the rating matrix is typically incomplete and NMF cannot directly work on it. In [23], Zhang et al. proposed the weighted NMF (WNMF) to work with incomplete matrices without a separate imputation procedure.

Given a rating matrix R and the associated weight matrix $W \in \mathbb{R}^{m \times n}$ that indicates the existence of values in R, the objective function of WNMF is

$$min_{U \geq 0, V \geq 0} f(R, W, U, V) = \|W \circ (R - UV^T)\|_F^2 \tag{3}$$

where U and V are two orthogonal nonnegative matrices, and $\circ$ denotes the element-wise multiplication.

$$w_{ij} = \begin{cases} 1 & if \quad r_{ij} \neq 0 \\ 0 & if \quad r_{ij} = 0 \end{cases} \quad (w_{ij} \in W, r_{ij} \in R) \tag{4}$$

When WNMF converges, $\tilde{R} = UV^T$ is the matrix with all missing entries filled. This process can be treated as either missing value imputation or unknown rating prediction.

Because of NMF's intrinsic property, when given a matrix R with objects as rows and attributes as columns, matrices U and V contain the clustering information of the objects. With that being said, in some cases, the data matrix R can represent relationships between two types of objects, e.g., user-item rating matrices in collaborating filtering applications and term-document matrices in text mining applications.

It is expected that both row (user/term) clusters and column (item/document) clusters can be obtained by performing NMF on R. With conventional NMF, it is very difficult to find two matrices U and V that represent user clusters and item clusters respectively at the same time. Hence, an extra factor matrix is needed to absorb the different scales of R, U, and V for simultaneous row clustering and column clustering [5]. Eq. (5) gives the objective function of the nonnegative matrix tri-factorization (NMTF).

$$min_{U\geq 0,S\geq 0,V\geq 0} f(R,U,S,V) = \|R - USV^T\|_F^2 \quad (5)$$

where $U \in \mathbb{R}_+^{m\times k}$, $S \in \mathbb{R}_+^{k\times l}$, and $V \in \mathbb{R}_+^{n\times l}$.

The use of S brings in a large scale of freedom for U and V so that they can focus on row and column clustering. In this scheme, both U and V are cluster membership indicator matrices while S is the coefficient matrix. Note that objects corresponding to rows in R are clustered into k groups and objects corresponding to columns are clustered into l groups.

With the auxiliary information of users and items, we can convert NMTF to a supervised learning procedure by applying cluster constraints to the objective function (5), giving the equation

$$\begin{aligned} &min_{U\geq 0,S\geq 0,V\geq 0} f(R,U,S,V,C_U,C_I) = \\ &\quad \alpha\cdot\|R - USV^T\|_F^2 + \beta\cdot\|U - C_U\|_F^2 + \gamma\cdot\|V - C_I\|_F^2 \end{aligned} \quad (6)$$

where α, β, and γ are coefficients that control the weight of each part. C_U and C_I are user cluster matrix and item cluster matrix, respectively. They are obtained by running clustering algorithms on user feature matrix F_U and item feature matrix F_I as mentioned in Section 3.

Combining Eqs. (3) and (6), we develop the objective function for the weighted and constrained nonnegative matrix tri-factorization, i.e.,

$$\begin{aligned} &min_{U\geq 0,S\geq 0,V\geq 0} f(R,W,U,S,V,C_U,C_I) = \\ &\alpha\cdot\|W\circ(R - USV^T)\|_F^2 + \beta\cdot\|U - C_U\|_F^2 + \gamma\cdot\|V - C_I\|_F^2. \end{aligned} \quad (7)$$

We name this matrix factorization the Cluster-NMF.

Update Formulas. In this section, we illustrate the derivation of the update formulas for Cluster-NMF.

Let $L = f(R,W,U,S,V,C_U,C_I)$, $X = \|W\circ(R - USV^T)\|_F^2$, $Y = \|U - C_U\|_F^2$, and $Z = \|V - C_I\|_F^2$. Take derivatives of X with respect to U, S, and V:

$$\frac{\partial X}{\partial U} = -2(W\circ R)VS^T + 2W\circ(USV^T)VS^T \quad (8)$$

$$\frac{\partial X}{\partial S} = -2U^T(W\circ R)V + 2U^T[W\circ(USV^T)]V \quad (9)$$

$$\frac{\partial X}{\partial V} = -2(W\circ R)^TUS + 2[W\circ(USV^T)]^TUS \quad (10)$$

Take derivatives of Y with respect to U, S, and V:

$$\frac{\partial Y}{\partial U} = 2U - 2C_U,\quad \frac{\partial Y}{\partial S} = \frac{\partial Y}{\partial V} = 0 \quad (11)$$

Take derivatives of Z with respect to U, S, and V:

$$\frac{\partial Z}{\partial U} = \frac{\partial Z}{\partial S} = 0,\quad \frac{\partial Z}{\partial V} = 2V - 2C_I \quad (12)$$

Using Eqs. (8) to (12), we get the derivatives of L:

$$\begin{aligned} \frac{\partial L}{\partial U} &= 2\alpha[W\circ(USV^T)]VS^T + 2\beta U \\ &\quad - 2\alpha(W\circ R)VS^T - 2\beta C_U \end{aligned} \quad (13)$$

$$\begin{aligned} \frac{\partial L}{\partial V} &= 2\alpha[W\circ(USV^T)]^TUS + 2\gamma V \\ &\quad - 2\alpha(W\circ R)^TUS - 2\gamma C_I \end{aligned} \quad (14)$$

$$\begin{aligned} \frac{\partial L}{\partial S} &= 2\alpha U^T[W\circ(USV^T)]V \\ &\quad - 2\alpha U^T(W\circ R)V \end{aligned} \quad (15)$$

To obtain update formulas, we apply the Karush-Kuhn-Tucker (KKT) complementary condition [12] to the nonnegativities of U, S, and V. We have

$$\begin{aligned} &\{2\alpha[W\circ(USV^T)]VS^T + 2\beta U \\ &\quad - 2\alpha(W\circ R)VS^T - 2\beta C_U\}_{ij}U_{ij} = 0 \end{aligned} \quad (16)$$

$$\begin{aligned} &\{2\alpha[W\circ(USV^T)]^TUS + 2\gamma V \\ &\quad - 2\alpha(W\circ R)^TUS - 2\gamma C_I\}_{ij}V_{ij} = 0 \end{aligned} \quad (17)$$

$$\{2\alpha U^T[W\circ(USV^T)]V - 2\alpha U^T(W\circ R)V\}_{ij}S_{ij} = 0 \quad (18)$$

They give rise to the corresponding update formulas:

$$U_{ij} = U_{ij}\cdot\frac{\{\alpha(W\circ R)VS^T + \beta C_U\}_{ij}}{\{\alpha[W\circ(USV^T)]VS^T + \beta U\}_{ij}} \quad (19)$$

$$V_{ij} = V_{ij}\cdot\frac{\{\alpha(W\circ R)^TUS + \gamma C_I\}_{ij}}{\{\alpha[W\circ(USV^T)]^TUS + \gamma V\}_{ij}} \quad (20)$$

$$S_{ij} = S_{ij}\cdot\frac{\{U^T(W\circ R)V\}_{ij}}{\{U^T[W\circ(USV^T)]V\}_{ij}} \quad (21)$$

Assume $k,l \ll \min(m,n)$, the time complexities of updating U, V, and S in each iteration are all $O(mn(k+l))$. Therefore, the time complexity of Cluster-NMF in each iteration is $O(mn(k+l))$.

The convergence analysis of the update formulas is presented in Appendix A.

Clustering the Auxiliary Information. In Eq. (7), the clustering membership indicator matrices are used as the constraints to perform the supervised learning. This requires the auxiliary information to be clustered beforehand. In [18], Su et al. proposed a nearest neighborhood based incremental clustering algorithm that can directly work on categorical data. We follow their algorithm and make some modifications so that it can be integrated into Cluster-NMF as the fundamental clustering technique.

Algorithm 1 depicts the steps to build the initial clusters for the existing feature matrices F_U and F_I. It is worth mentioning that since this algorithm takes categorical data as input, for each attribute, we store all possible values in one column. For example, a user vector (a row in F_U) contains 3 attributes (columns), gender, age, and occupation. Each column has a different number of possible values, e.g., gender has two possible values: male and female. Same format applies to F_I.

Detailed Algorithm. The whole process of performing Cluster-NMF, the non-incremental version of iCluster-NMF, on a rating matrix is illustrated in Algorithm 2 . In this algorithm, an extra stop criterion, the maximum iteration count, is set to terminate the program at a reasonable point. In collaborative filtering applications, this value varies from 10 to 100 and can generally produce satisfactory results.

4.2. iCluster-NMF

When new rows/columns are available, they are imputed by iCluster-NMF with the aid of U, S, V, C_U, and C_I generated by Algorithm 2.

Technically, iCluster-NMF is identical to Cluster-NMF, but focuses on a series of new rows or columns. Meanwhile, when new feature data ΔF_U and ΔF_I arrive, they need to be clustered into existing clusters, otherwise new clusters are created. Eq. (7) indicates the relationship between the dimensions of U and C_U, V and C_I. This means that once the clusters are updated, NMF must be completely recomputed.

In Eq. (1), we see that $T \in \mathbb{R}^{p\times n}$ is added to R as a few rows. This process is illustrated in Figure 1. Like Section 4.1, the objective function is developed by

$$min_{\Delta U\geq 0} f(T, W_T, \Delta U, S, V, \Delta C_U) = \alpha \cdot \|W_T \circ (T - \Delta U S V^T)\|_F^2 + \beta \cdot \|\Delta U - \Delta C_U\|_F^2. \quad (22)$$

Accordingly, the update formula for this objective function is obtained as follows

$$\Delta U_{ij} = \Delta U_{ij} \cdot \frac{\{\alpha(W_T \circ T)VS^T + \beta\Delta C_U\}_{ij}}{\{\alpha[W_T \circ (\Delta U S V^T)]VS^T + \beta\Delta U\}_{ij}}. \quad (23)$$

Since the row update only works on new rows, the time complexity of the algorithm in each iteration is $O(pn(l+k)+pkl)$. Assume $k, l \ll \min(p, n)$, the time complexity is then simplified to $O(pn(l+k))$.

Algorithm 1 Initial Cluster Builder

Input:
Object feature matrix: $D \in \mathbb{R}^{m\times f}$, where there are m objects and f attributes;
Maximum number of clusters: $maxK$;
Initial radius threshold: s;
Radius decreasing step: d_s;
Empty cluster collection: CS;
Initial cluster feature: CF.
Output:
Updated radius threshold: s';
Updated cluster collection: CS';
Updated cluster feature: CF';

1: Set CS' to empty and $maxScore$ to 0;
2: **for** $numK = 1$ *to* $maxK$ **do**
3: Reset D, s, CS, and CF;
4: **while** D is not empty **do**
5: Read a new object O from D;
6: **if** CS is empty **then**
7: Create a cluster with O and place it into CS;
8: **else**
9: Calculate the distance between O and each cluster in CS and find out the smallest distance $minDis_{oc}$;
10: **if** $minDis_{oc} < s$ **then**
11: Insert O into the nearest cluster and update CF;
12: **else**
13: Create a cluster with O and place it into CS;
14: **end if**
15: **end if**
16: **if** $|CS| > numK$ **then**
17: Calculate the distance between any two clusters and merge the two clusters with the minimum distance $minDis_{cc}$;
18: **if** $minDis_{cc} > s$ **then** $s = minDis_{cc}$;
19: **end if**
20: **end while**
21: **if** $|CS| < numK$ **then** $s = s - d_s$; Goto 3;
22: Calculate the inter-cluster distance and inner-cluster distance to obtain the clustering score $lScore$.
23: **if** $lScore > mScore$ **then** $mScore = lScore$; $CS' = CS$; $CF' = CF$; $s' = s$;
24: **end for**

The column update is almost identical to the row update. When the new data $G \in \mathbb{R}^{m\times q}$ arrives, it is updated according to Eq. (24). The time complexity for

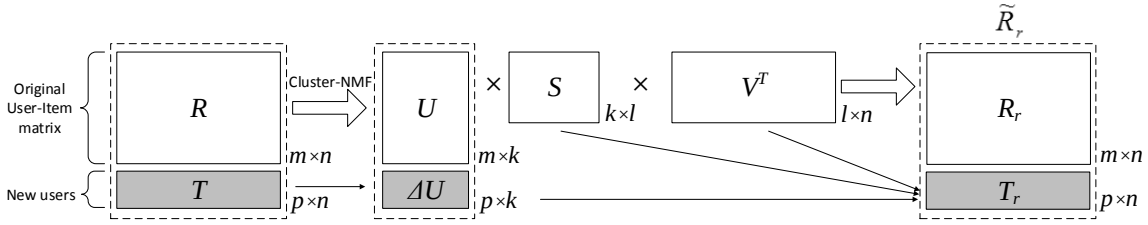

Figure 1. Updating new rows in iCluster-NMF

Algorithm 2 Cluster-NMF

Input:
User-Item rating matrix: $R \in \mathbb{R}^{m \times n}$;
User feature matrix: $F_U \in \mathbb{R}^{m \times k_U}$;
Item feature matrix: $F_I \in \mathbb{R}^{n \times k_I}$;
Coefficients in objective function: α, β, and γ;
Number of maximum iterations: $MaxIter$.

Output:
Factor matrices: $U \in \mathbb{R}_+^{m \times k}$, $S \in \mathbb{R}_+^{k \times l}$, $V \in \mathbb{R}_+^{n \times l}$;
User cluster membership indicator matrix: $C_U \in \mathbb{R}^{m \times k}$;
Item cluster membership indicator matrix: $C_I \in \mathbb{R}^{n \times l}$;

Cluster users based on F_U by Algorithm 1 $\rightarrow C_U$;
Cluster items based on F_I by Algorithm 1 $\rightarrow C_I$;
Initialize U, S, and V with random values;
Build weight matrix W by Eq. (4);
Set $iteration = 1$ and $stop = false$;
while $(iteration <= MaxIter)$ *and* $(stop == false)$ **do**
  $U_{ij} \leftarrow U_{ij} \cdot \frac{\{\alpha(W \circ R)VS^T + \beta C_U\}_{ij}}{\{\alpha[W \circ (USV^T)]VS^T + \beta U\}_{ij}}$;
  $V_{ij} \leftarrow V_{ij} \cdot \frac{\{\alpha(W \circ R)^T US + \gamma C_I\}_{ij}}{\{\alpha[W \circ (USV^T)]^T US + \gamma V\}_{ij}}$;
  $S_{ij} \leftarrow S_{ij} \cdot \frac{\{U^T(W \circ R)V\}_{ij}}{\{U^T[W \circ (USV^T)]V\}_{ij}}$;
  $L \leftarrow \alpha \cdot \|W \circ (R - USV^T)\|_F^2 + \beta \cdot \|U - C_U\|_F^2 + \gamma \cdot \|V - C_I\|_F^2$;
  if (L increases in this iteration) **then**
    $stop = true$;
    Restore U, S, and V to their values in last iteration.
  end if
  $iteration = iteration + 1$;
end while

the column update is $O(qm(l+k))$.

$$\Delta V_{ij} = \Delta V_{ij} \cdot \frac{[\alpha(W_G \circ G)^T US + \gamma \Delta C_I]_{ij}}{\{\alpha[W_G \circ (US\Delta V^T)]^T US + \gamma \Delta V\}_{ij}} \quad (24)$$

Algorithm 3 Incremental clustering algorithm

Input:
Object feature matrix: $\Delta D \in \mathbb{R}^{m \times f}$, where there are m objects and f attributes;
Maximum number of clusters: $maxK$;
Radius threshold: s';
Cluster collection: CS';
Cluster feature: CF'.

Output:
Updated radius threshold: s'';
Updated cluster collection: CS'';
Updated cluster feature: CF'';

while ΔD is not empty **do**
  Read a new object O from ΔD;
  Calculate the distance between O and each cluster in CS' and find out the smallest distance $minDis_{oc}$;
  if $minDis_{oc} < s'$ **then**
    Insert O into the nearest cluster and update CF';
  else
    Create a cluster with O and place it into CS';
  end if
  if $|CS'| > maxK$ **then**
    Calculate the distance between any two clusters and merge the two clusters with the minimum distance $minDis_{cc}$;
    if $minDis_{cc} > s'$ **then** $s'' = minDis_{cc}$;
  end if
end while
$CF'' = CF'$; $CS'' = CS'$; $s'' = s'$;

5. A Comparison Model: kMeans-NMF

In the previous section, we introduced iCluster-NMF that utilizes the incremental clustering algorithm to obtain and update user clusters and item clusters. The cluster quantities can change when new data arrives. This would also affects the matrix dimensions in the NMF update. To study how this automated procedure performs differently from a user-controlled clustering based NMF data update, we propose a comparison model, named kMeans-NMF.

The model uses the K-Means algorithm instead of Algorithms 1 and 3 to cluster the users and items. It is shown in Eq. (7) that the dimensionality of U are equal to the dimensionality of C_U while V and C_I have the same dimensionalities. It requires the number of user clusters k and the number of item clusters l to be predetermined. To do so, the data owner has to run the K-Means algorithm multiple times and find out the best values for k and l. The cluster quantities do not change in the whole process. Therefore, the dimensions of the factor matrices remain the same during the update.

When new users' and items' feature data becomes available, K-Means calculates the distance between each new object and the existing cluster centroids so the closest cluster is identified. The new object is then added to this cluster and the centroid is updated.

In general, kMeans-NMF is identical to iCluster-NMF but has a different clustering algorithm as well as no recomputation for NMF.

6. Experimental Study

6.1. Data Description

In the experiments, we adopt the MovieLens [17], Sushi [10], and LibimSeTi [2] datasets as the test data. Table 1 collects the statistics of the datasets.

Table 1. Statistics of the data

Dataset	#users	#items	#ratings	Sparsity
MovieLens	943	1,682	100,000	93.7%
Sushi	5,000	100	50,000	90%
LibimSeTi	2,000	5,625	129,281	98.85%

The public MovieLens dataset has 3 subsets, 100K(100,000 ratings), 1M(1,000,000 ratings) and 10M(10,000,000 ratings). The first dataset, which is adopted in the experiments, has 943 users and 1,682 items. The 100,000 ratings, ranging from 1 to 5, were divided into two parts: the training set with 80,000 ratings and the test set with 20,000 ratings. In addition to rating data, user demographic information and item genre information are also available.

The Sushi dataset describes users' preferences on different kinds of sushi. There are 5,000 users and 100 sushi items. Each user has rated 10 items, with a rating ranging from 1 to 5. That is to say, there are 50,000 ratings in this dataset. To build the test set and the training set, for every user, 2 out of 10 ratings were randomly selected and were inserted into the test set (10,000 ratings) while the rest of ratings were used as the training set (40,000 ratings). Similar to MovieLens, the Sushi dataset comes with user demographic information as well as item group information and some attributes, e.g., the heaviness/oiliness in taste and how frequently the user eats the sushi.

The LibimSeTi dating dataset was gathered by LibimSeTi.cz, an online dating website. It contains 17,359,346 anonymous ratings of 168,791 profiles made by 135,359 users as dumped on April 4, 2006. However, only the user's gender is provided with the data. Later sections will show how to resolve this problem with the lack of item information. Confined to the memory limit of the test computer, the experiments only used 2,000 users and 5,625 items[1] with 108,281 ratings in the training set and 21,000 ratings in the test set. Ratings are on a $1 \sim 10$ scale where 10 is best.

6.2. Data Pre-processing

Because iCluster-NMF and kMeans-NMF require different feature data formats (numerical vs categorical), the data fed to them should be processed in different ways. In the MovieLens dataset, user demographic information includes user ID, age, gender, occupation, and zip code. Among them, we utilized age, gender, and occupation as features. For ages, the numbers were categorized into 7 groups: 1-17, 18-24, 25-34, 35-44, 45-49, 50-55, >=56. For gender, there are two possible values: male and female. According to the statistics, there are 21 occupations: administrator, artist, doctor, and so on. For iCluster-NMF, since it directly works on categorical data, we built the user feature matrix F_U with 3 attributes ($k_U = 3$). They correspond to gender (2 possible values), age (7 possible values), and occupation (21 possible values), respectively. In contrast, for kMeans-NMF, the categories were converted to numbers since K-Means algorithm only works on numerical data. The user feature matrix F_U was built with 30 attributes ($k_U = 30$); each user was represented as a row vector with 30 elements. An element will be set to 1 if the corresponding attribute value is true for this user and 0 otherwise. Similar with the user feature matrix, the item feature matrix was built in terms of their genres. Movies in this dataset were attributed to 19 genres and hence the item feature matrix F_I has 6 attributes for iCluster-NMF ($k_I = 6$ as a single movie could have up to 6 genres) and 19 attributes for kMeans-NMF ($k_I = 19$).

In the Sushi dataset, eight of the user demographic attributes were used: gender, age, city in which the user has lived the longest until age 15 (plus region and east/west). Additionally, the city (plus region and east/west) in which the user currently lives was also used. In this case, users' age was categorized into six groups by the data provider: 15-19, 20-29, 30-39, 40-49, 50-59, >=60. User gender consists of male and female, which is the same as MovieLens. There are 48 cities (Tokyo, Hiroshima, Osaka, etc.), 12 regions (Hokkaido,

[1] User profiles are considered as items for this dataset

Tohoku, Hokuriku, etc.) and 2 possible east/west values (either the eastern or western part of Japan). Thus, the user feature matrix for iCluster-NMF on this dataset has 5,000 rows and 8 columns. Nevertheless, since there are too many possible values ($2 + 6 + (48 + 12 + 2) \times 2 = 132$ values) for all attributes, only gender and age were used to build the user feature matrix for kMeans-NMF. This makes the matrix have 5,000 rows and 8 columns (2 genders plus 6 age groups). The item feature matrix, on the other hand, has 100 rows and 3 columns for iCluster-NMF (16 columns for kMeans-NMF) since there are 2 styles, 2 major groups, and 12 minor groups.

Since the LibimSeTi dataset only provides the user gender information, it was simply used as the user cluster indicator matrix C_U. Note that in this dataset, there are three possible gender values: male, female, and unknown. To be consistent, the number of user clusters is set to 1 for iCluster-NMF and 3 for kMeans-NMF.

6.3. Evaluation Strategy

To evaluate the algorithms, the error of unknown value prediction and the time cost were measured. Besides iCluster-NMF and kMeans-NMF, a naive Cluster-NMF was exploited as the benchmark in the experiments for comparisons. Two SVD based collaborative filtering algorithms were studied as well.

Naive Cluster-NMF: The Benchmark Model. The general idea of the naive Cluster-NMF is quite close to iCluster-NMF. The only difference is the way of updating the clusters. In iCluster-NMF, we use Alogrthim 1 to build the initial clusters which are then updated by Algorithm 3. In contrast, the naive Cluster-NMF does not use incremental clustering but simply uses the idea of Alogrthim 1 to cluster the existing objects to the fixed number of clusters and re-cluster them (to the fixed number of clusters as well) when new data is available. In other words, F'_U in Eq. (1) and F'_I in Eq. (2) are re-clustered every time there is an update on the data. This will significantly lower the performance of the algorithm but it theoretically produces the most accurate result among all.

The SVD Based Comparison Models. In order to demonstrate how much improvement our algorithms have achieved, they were compared to two SVD based collaborative filtering algorithms and the performance was evaluated. In [1], Brand proposed a recommender system that leveraged the probabilistic imputation to fill the missing values in the incomplete rating matrix and then used the incremental SVD to update the imputed rating matrix. This makes SVD work seamlessly for CF purposes. We denote this algorithm as iSVD. The SVD based method that was proposed by Wang and Zhang [20] is similar to [1] but has additional processing steps to ensure privacy protection. Additionally, it uses mean value imputation instead of the probabilistic imputation to remove missing values. We denote this algorithm as pSVD. It is worth mentioning that neither of them considers auxiliary information so only the rating matrix is used.

Evaluation Measures and Experiment Procedure. The experiments measured the prediction error and the time cost on three proposed algorithms as well as iSVD and pSVD. The prediction error was measured by calculating the difference between the actual ratings in the test set and the predicted ratings. A common and popular criterion is the mean absolute error (MAE), which can be calculated as follows:

$$MAE = \frac{1}{|TestSet|} \sum_{r_{ij} \in TestSet} |r_{ij} - p_{ij}| \qquad (25)$$

where p_{ij} is the predicted rating.

When building the starting matrix R, the split ratio was used to decide how many ratings would to be removed from the whole training data. For example, there are 1,000 users and 500 items with their ratings in the training data. If the split ratio is 40% and a row update will be done, we use the first 400 rows as the starting matrix ($R \in \mathbb{R}^{400 \times 500}$). The remaining 600 rows of the training matrix will be added to R in several rounds. Similarly, if a column update will be performed, we use the first 200 columns as the starting matrix ($R \in \mathbb{R}^{1000 \times 200}$) while the remaining 300 columns will be added to R in several rounds.

In each round, 100 rows/columns were added to the starting matrix. If the number of the rows/columns of new data is not divisible by 100, the last round will update the rest. Therefore, in this example, the remaining 600 rows will be added to R in 6 rounds with 100 rows each. Note that the Sushi data set only has 100 items in total but we still want to test the column update on it so 10 items were added instead of 100 in each round.

The basic procedure of the experiments is as follows:

1. Perform Algorithm 1 and Algorithm 2 on R;
2. Append the new data to R by iCluster-NMF, kMeans-NMF, and naive Cluster-NMF (nCluster-NMF for short), yielding the updated rating matrix $\tilde{R}_r$;
3. Measure the prediction errors and the time costs of the updates;
4. Compare and study the results.

The machine we used was equipped with Intel® Core™ i5-2405S processor, 8GB RAM and was installed with UNIX operating system. We wrote and ran the code in MATLAB.

6.4. Results and Discussion

Parameter Setup. The parameters that have to be determined by kMeans-NMF are listed in Table 2, where k is the column dimension of matrix U and l is the column dimension of V.

Table 2. Parameter setup for kMeans-NMF

Dataset	α	β	γ	k	l	$MaxIter$
MovieLens	0.2	0	0.8	7	7	10
Sushi	0.4	0.6	0	7	5	10
LibimSeTi	1	0	0	3	10	10

For the MovieLens dataset, we set $\alpha = 0.2$, $\beta = 0$, and $\gamma = 0.8$, which means that the prediction relied mostly on the item cluster matrix, and then the rating matrix, whereas eliminated the user cluster matrix. This combination was selected after probing many possible cases. Both k and l are set to 7 because K-Means was prone to generate empty clusters with greater k and l, especially on the data with very few users or items. It is worth mentioning that if β or γ is a non-zero value, the user or item cluster matrix will be used and k or l is equal to the number of user clusters or item clusters. As long as β or γ is zero, the algorithm will eliminate the corresponding cluster matrix and k or l will be unrelated to the number of user clusters or item clusters.

For the Sushi dataset, we set $\alpha = 0.4$, $\beta = 0.6$, and $\gamma = 0$. The parameters indicate that the user cluster matrix played the most critical role during the update process. In contrast, rating matrix was the second important factor as it indicates the user preference on items. The item cluster matrix seems trivial so it did not participate in the computation. We set k to 7 and l to 5 based on the same reason as mentioned in the previous paragraph.

For the LibimSeTi dataset, full weight was given to the rating matrix. The user and item cluster matrices received zero weight since they did not contribute anything to the positive results. As mentioned in the data description, users' auxiliary information only includes the gender with three possible values. So k was set to 3. In this case, l only denotes the column dimension of V and was set to 10.

Table 3. Parameter setup for iCluster-NMF and nCluster-NMF

Dataset	$maxK$	s	d_s
MovieLens	10	1	0.1
Sushi	10	1	0.1
LibimSeTi	3/1	1	0.1

The iCluster-NMF and nCluster-NMF are in general the same as kMeans-NMF but with different clustering approaches and NMF re-computation strategies. In Algorithm 1, the maximum number of clusters $maxK$, the initial radius threshold s, and the radius decreasing step d_s must be determined in advance. Table 3 gives the parameter setup for iCluster-NMF and nCluster-NMF. Note that the LibimSeTi dataset has $maxK = 3$ for user clusters and $maxK = 1$ for item clusters.

As far as iSVD and pSVD, the only parameter involved is the rank of the singular matrix. To determine this value, both algorithms were run for multiple times with different ranks. We selected the numbers that achieved the optimal outcomes. The best ranks for the MovieLens, the Sushi, and the LibimSeti datasets are 13, 7, and 10, respectively.

Experimental Results. Figure 2 shows the time cost for updating new rows and columns by kMeans-NMF, iCluster-NMF, nCluster-NMF, as well as iSVD and pSVD. In most cases, nCluster-NMF and pSVD took significantly longer time than others. This is because nCluster-NMF was used to probe all possible cluster quantities to find out the choices that achieve the best MAE's. That is to say, it tries to cluster users into k groups and items into l groups, where $k, l = \{1, 2, ..., 10\}$, which results in 100 combinations. In addition, nCluster-NMF needs to re-cluster the whole data every time the new portion arrives. This requires even more time. As for pSVD, since it uses the mean value of each column to impute all missing values in that column, when a large amount of data is involved in the update (e.g. the row update on MovieLens and the column update on Sushi), the time cost can be high. The performance of iSVD is not as sensitive as pSVD to the data size but it also suffers from high matrix dimensionality, as shown in Figure 2(e).

Comparing kMeans-NMF and iCluster-NMF, it can be seen that their time costs were close in the process, though the former was slightly faster than the latter. This is because iCluster-NMF not only updates the clusters' content as kMeans-NMF does, but also combines existing clusters or creates new clusters when necessary. The cluster update itself does not cost more time but since the number of clusters changes in some cases, the NMF has to be recomputed, which requires additional time.

As a reference, Table 4 lists the optimal number of clusters on the Sushi dataset. Note that the split ratio determines how many rows or columns should be present in the starting matrix. iCluster-NMF first runs Algorithm 1 on R to find the optimal number of clusters for users and items. Then they will be updated when new data is added to R. The numbers shown in this table are the final cluster quantities. When the rows were being updated, the model kept the columns unchanged and vice versa. This is why the number of item clusters remained the same when performing the row update and the number of user clusters remained the same when performing the column update. From the table,

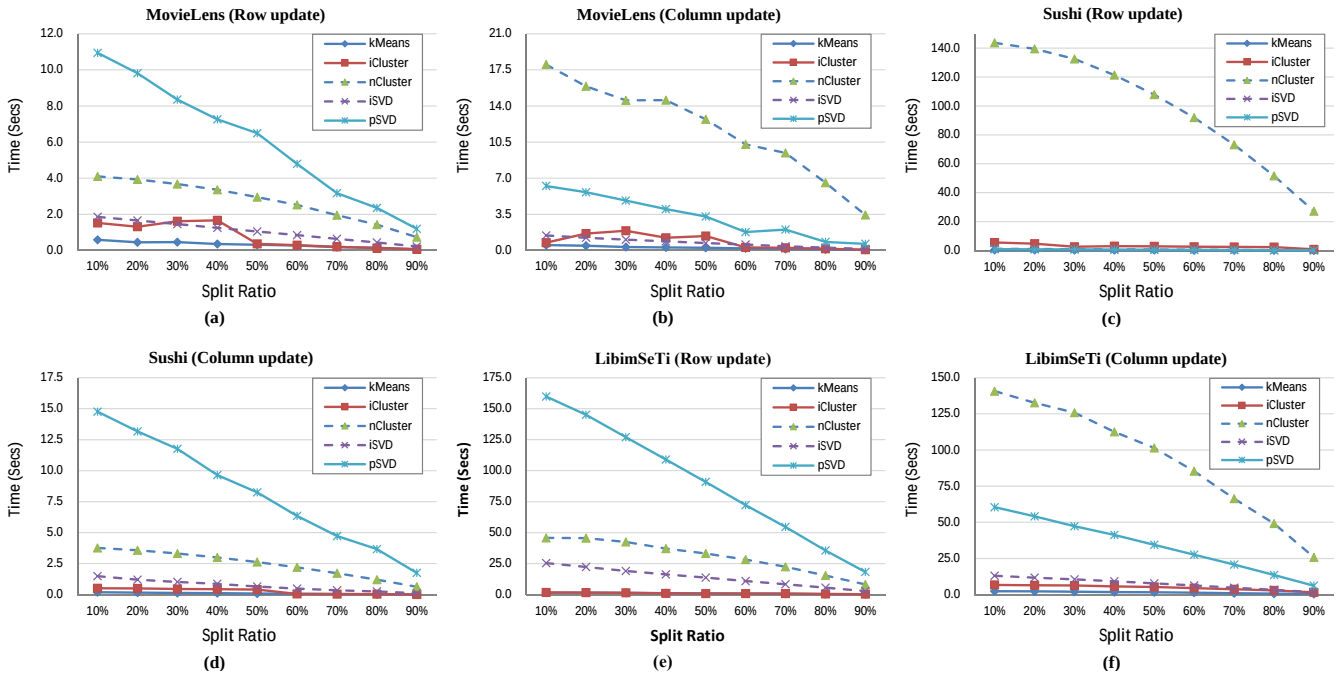

Figure 2. Time cost variation with split ratio

Table 4. Optimal number of clusters on the Sushi dataset

Split Ratio		10%	20%	30%	40%	50%	60%	70%	80%	90%
iCluster-NMF (Row)	#UserClusters	10	10	10	10	10	10	9	10	8
	#ItemClusters	10	10	10	10	10	10	10	10	10
iCluster-NMF (Column)	#UserClusters	5	5	5	5	5	5	5	5	5
	#ItemClusters	5	5	5	5	4	4	7	9	10
nCluster-NMF (Row)	#UserClusters	7	7	7	7	7	7	7	7	7
	#ItemClusters	7	7	7	7	7	7	7	7	7
nCluster-NMF (Column)	#UserClusters	6	6	6	6	6	6	6	6	6
	#ItemClusters	10	10	10	10	10	10	10	10	10

one can see that the best combinations obtained by nClsuter-NMF were 7 user clusters / 7 item clusters for the row update and 6 user clusters / 10 item clusters for the column update. Although the numbers are different from the ones obtained by iCluster-NMF, their MAE's are nearly the same.

The mean absolute errors of the prediction are plotted in Figure 3. iSVD performed worst on all datasets while nCluster-NMF reached the best results in most cases. Due to the way that nCluster-NMF works, the MAE's were consistently at the same level. They did not change significantly with varying split ratios. The only exception was the row update on the Sushi dataset, where iCluster-NMF achieved lower MAE than nCluster-NMF when the split ratio became higher. This to some extent means that updating the number of clusters in iCluster-NMF benefited the lower global prediction error. The figures show that iCluster-NMF outperformed kMeans-NMF on all three datasets. It is interesting to look at the errors of pSVD, which were very close to iCluster-NMF on LibimSeTi but were worse on other datasets. Remember that we mentioned in Section 6.1, LibimSeTi only provides user gender information. In other words, our proposed models did not really receive any extra helpful information from this dataset. Thus, its prediction accuracy was almost identical to pSVD's, which does not utilize auxiliary information at all.

We attribute the promising results to not only the incremental clustering but also the recomputation of NMF. On one hand, clusters are updated when the new data comes in. This strategy ensures that the cluster membership indicator matrices C_U and C_I in Eq. (7) always maintain up-to-date relationships between either rows or columns. This, in turn, benefits the NMF update. On the other hand, due to the accumulated error in the incremental update, NMF needs to be recomputed to maintain the accuracy. It

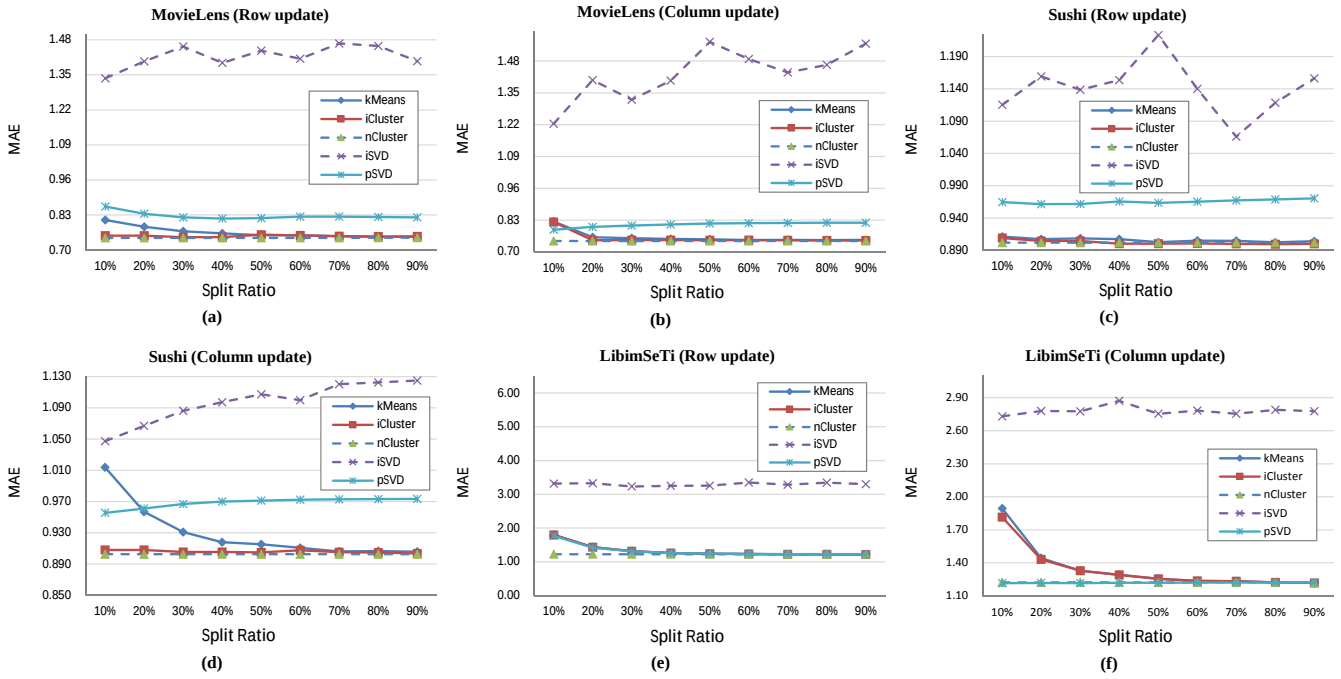

Figure 3. MAE variation with split ratio

is not convenient for the data owner to determine when to perform the recomputation and update the dimensions of the factor matrices. In this situation, iCluster-NMF recomputes NMF when the number of clusters change. It also explains why the MAE's of kMeans-NMF and iCluster-NMF tend to be close when the split ratios become higher —since kMeans-NMF does not recompute NMF, the more data it starts with, the less accumulated update error it has. Nevertheless, with more data available, the error will inevitably become larger.

As a summary, the iCluster-NMF data update algorithm produced higher prediction accuracy while costing just a little more time, if not the same as kMeans-NMF did. More importantly, the former does not need the data owner to determine the number of user and item clusters and can recompute the NMF when necessary. Once useful auxiliary information became available, both algorithms outperformed the incremental SVD based algorithms with respect to the prediction accuracy. The results are encouraging.

7. Conclusion and Future Work

In this paper, we propose an NMF based data update approach with automated dimension determination for collaborative filtering purposes. It integrates the incremental clustering technique into the NMF based data update algorithm. This approach utilizes the auxiliary information to build the cluster membership indicator matrices of users and items. These matrices are regarded as the constraints in updating the weighted nonnegative matrix tri-factorization. The proposed approach, named iCluster-NMF, does not require the data owner to determine when to recompute the NMF and the dimensions of the factor matrices. Instead, it sets the dimensions of the factor matrices according to the clustering result on users and items and updates it automatically. Experiments conducted on three different datasets demonstrate the high accuracy and performance of iCluster-NMF.

In the real world, when people are shopping online, the factors that affect their decisions are not quite unique. In collaborative filtering research, most literatures focus on the correlations between users and items. This is apparently one of the most consequential factors but there are also some others. In future work, we will take into account more related auxiliary information, such as social networks, to achieve better prediction accuracy. We will also make use of the group preference to provide privacy preserving product recommendations.

Appendix A. Convergence Analysis for Cluster-NMF Update Formulas

We follow [13] to prove that the objective function $L = f(R, W, U, S, V, C_U, C_I)$ is nonincreasing under the update formulas (19), (20), and (21).

Definition 1. $H(u, u')$ is an auxiliary function for $F(u)$ if the conditions

$$H(u, u') \geq F(u), \quad H(u, u) = F(u) \tag{A.1}$$

are satisfied.

Lemma 1. If H is an auxiliary function for F, then F is nonincreasing under the update

$$u^{t+1} = \underset{u}{\operatorname{argmin}} H(u, u^t) \tag{A.2}$$

Lemma 1 can be easily proved since we have $F(u^{t+1}) = H(u^{t+1}, u^{t+1}) \leq H(u^{t+1}, u^t) \leq H(u^t, u^t) = F(u^t)$.

We will prove the convergences of the update formulas (19), (20), and (21) by showing that they are equivalent to Eq. (A.2), with proper auxiliary functions defined.

Let us rewrite the objective function L,

$$\begin{aligned} L = & tr\{\alpha(W \circ R)^T \cdot (W \circ R)\} \\ & + tr\{-2\alpha(W \circ R)^T \cdot [W \circ (USV^T)]\} \\ & + tr\{\alpha[W \circ (USV^T)]^T \cdot [W \circ (USV^T)]\} \\ & + tr(\beta U^T U) + tr(-2\beta U^T C_U) + tr(\beta C_U^T C_U) \\ & + tr(\gamma V^T V) + tr(-2\gamma V^T C_I) + tr(\gamma C_I^T C_I) \end{aligned} \tag{A.3}$$

where $tr(*)$ is the trace of a matrix.

Eliminating the irrelevant terms, we define the following functions that are only related to U, V, and S, respectively.

$$\begin{aligned} L(U) = & tr\{-2\alpha(W \circ R)^T \cdot [W \circ (USV^T)] \\ & + \alpha[W \circ (USV^T)]^T \cdot [W \circ (USV^T)] \\ & + \beta U^T U - 2\beta U^T C_U\} \\ = & tr\{-2[\alpha(W \circ R)VS^T + \beta C_U]U^T \\ & + U^T[\alpha W \circ (USV^T)VS^T] + U^T(\beta U)\} \end{aligned} \tag{A.4}$$

$$\begin{aligned} L(V) = & tr\{-2\alpha(W \circ R)^T \cdot [W \circ (USV^T)] \\ & + \alpha[W \circ (USV^T)]^T \cdot [W \circ (USV^T)] \\ & + \gamma V^T V - 2\gamma V^T C_I\} \\ = & tr\{-2[\alpha(W \circ R)^T US + \gamma C_I]V^T \\ & + V^T[\alpha(W \circ (USV^T))^T US] + V^T(\gamma V)\} \end{aligned} \tag{A.5}$$

$$\begin{aligned} L(S) = & tr\{-2\alpha(W \circ R)^T \cdot [W \circ (USV^T)] \\ & + \alpha[W \circ (USV^T)]^T \cdot [W \circ (USV^T)]\} \\ = & tr\{[-2\alpha U^T(W \circ R)V]S^T \\ & + [\alpha U^T(W \circ (USV^T))V]S^T\} \end{aligned} \tag{A.6}$$

Lemma 2. For any matrices $X \in \mathbb{R}_+^{n \times n}$, $Y \in \mathbb{R}_+^{k \times k}$, $F \in \mathbb{R}_+^{n \times k}$, $F' \in \mathbb{R}_+^{n \times k}$, and X, Y are symmetric, the following inequality holds

$$\sum_{i=1}^{n}\sum_{j=1}^{k} \frac{(XF'Y)_{ij}F_{ij}^2}{F'_{ij}} \geq tr(F^T XFY) \tag{A.7}$$

The proof of Lemma 2 is presented in [5]. We use this lemma to build an auxiliary function for $L(U)$. Since $L(V)$ and $L(S)$ are similar to $L(U)$, their convergences are not necessary to be discussed.

Lemma 3.

$$\begin{aligned} H(U, U') = & -2\sum_{ij}\{[\alpha(W \circ R)VS^T + \beta C_U]U^T\}_{ij} \\ & + \sum_{ij}\frac{\{\alpha W \circ (U'SV^T)VS^T + \beta U'\}_{ij}U_{ij}^2}{U'_{ij}} \end{aligned} \tag{A.8}$$

is an auxiliary function of $L(U)$ and the global minimum of $H(U, U')$ can be achieved by

$$U_{ij} = U'_{ij} \cdot \frac{\{\alpha(W \circ R)VS^T + \beta C_U\}_{ij}}{\{\alpha[W \circ (U'SV^T)]VS^T + \beta U'\}_{ij}} \tag{A.9}$$

Proof. We need to prove two conditions as specified in Definition 1. It is apparent that $H(U, U) = L(U)$. According to Lemma 2, we have

$$\begin{aligned} & \sum_{ij}\frac{\{\alpha W \circ (U'SV^T)VS^T + \beta U'\}_{ij}U_{ij}^2}{U'_{ij}} \\ & = \sum_{ij}\frac{\{\alpha W \circ (U'SV^T)VS^T\}_{ij}U_{ij}^2}{U'_{ij}} + \sum_{ij}\frac{\{\beta U'\}_{ij}U_{ij}^2}{U'_{ij}} \\ & \geq tr\{U^T[\alpha W \circ (USV^T)VS^T]\} + tr\{U^T(\beta U)\}. \end{aligned} \tag{A.10}$$

Therefore, $H(U, U') \geq L(U)$. Thus $H(U, U')$ is an auxiliary function of $L(U)$.

To find the global minimum of $H(U, U')$ with U' fixed, we take the derivative of $H(U, U')$ with respect to U_{ij} and let it be zero:

$$\begin{aligned} \frac{\partial H(U, U')}{\partial U_{ij}} = & \{-2[\alpha(W \circ R)VS^T + \beta C_U]\}_{ij} \\ & + 2\frac{\{\alpha W \circ (U'SV^T)VS^T + \beta U'\}_{ij}U_{ij}}{U'_{ij}} = 0 \end{aligned} \tag{A.11}$$

Solving for U_{ij}, we have

$$U_{ij} = U'_{ij} \cdot \frac{\{\alpha(W \circ R)VS^T + \beta C_U\}_{ij}}{\{\alpha[W \circ (U'SV^T)]VS^T + \beta U'\}_{ij}} \tag{A.12}$$

Since $F(U^0) = H(U^0, U^0) \geq H(U^1, U^0) \geq F(U^1) \geq ...$, $F(U)$ is monotonically decreasing and updating U by Eq. (A.12) can reach the global minimum. □

Similarly, the convergences of update formulas (20) and (21) can be proved as well.

References

[1] Brand, M. (2003) Fast Online SVD Revisions for Lightweight Recommender Systems. In *Proceedings of SIAM International Conference on Data Mining* (SIAM).

[2] Brozovsky, L. and Petricek, V. (2007) Recommender System for Online Dating Service. In *Proceedings of Znalosti 2007 Conference* (VSB).

[3] Chen, G., Wang, F. and Zhang, C. (2007) Collaborative Filtering Using Orthogonal Nonnegative Matrix Tri-factorization. In *Proceedings of the 7th IEEE International Conference on Data Mining Workshops* (IEEE): 303–308.

[4] Chen, T., Zhang, W., Lu, Q., Chen, K., Zheng, Z. and Yu, Y. (2012) SVDFeature: A Toolkit for Feature-based Collaborative Filtering. *Journal of Machine Learning Research* **13**: 3619–3622.

[5] Ding, C., Li, T., Peng, W. and Park, H. (2006) Orthogonal Nonnegative Matrix Tri-Factorizations for Clustering. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM): 126–135.

[6] Gu, Q., Zhou, J. and Ding, C. (2010) Collaborative Filtering: Weighted Nonnegative Matrix Factorization Incorporating User and Item Graphs. In *Proceedings of the 7th IEEE International Conference on Data Mining Workshops* (SIAM): 199–210.

[7] Huang, Z. (1997) A Fast Clustering Algorithm to Cluster Very Large Categorical Data Sets in Data Mining. In *Research Issues on Data Mining and Knowledge Discovery*: 1–8.

[8] Huang, Z., Deng, S. and Xu, X. (2005) Improving K-modes Algorithm Considering Frequencies of Attribute Values in Mode **3801**: 157–162.

[9] Kabir, S.M.A., Youssef, A.M. and Elhakeem, A.K. (2007) On Data Distortion for Privacy Preserving Data Mining. In *Proceedings of Canadian Conference on Electrical and Computer Engineering* (IEEE): 308 – 311.

[10] Kamishima, T. and Akaho, S. (2006) Efficient Clustering for Orders. In *Proceedings of the 2nd International Workshop on Mining Complex Data*: 274–278.

[11] Kim, J. and Park, H. (2008) *Sparse Nonnegative Matrix Factorization for Clustering*. Tech. rep., Georgia Institute of Technology.

[12] Kuhn, H. and Tucker, A. (1951) Nonlinear Programming. In *Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probability*: 481–492.

[13] Lee, D.D. and Seung, H.S. (2001) Algorithms for Non-negative Matrix Factorization. *Advances in Neural Information Processing Systems* **13**: 556–562.

[14] MacQueen, J.B. (1967) Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, **1**: 281–297.

[15] Pauca, V.P., Shahnaz, F., Berry, M.W. and Plemmons, R.J. (2004) Text Mining Using Nonnegative Matrix Factorizations. In *Proceedings of the 2004 SIAM Interational Conference on Data Mining* (SIAM), **54**: 452–456.

[16] Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P. and Riedl, J. (1994) GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work* (ACM): 175–186.

[17] Sarwar, B.M., Karypis, G., Konstan, J.A. and Riedl, J.T. (2000) Application of Dimensionality Reduction in Recommender Systems – A Case Study. In *Proceedings of ACM WebKDD Workshop* (ACM).

[18] Su, X., Lan, Y., Wan, R. and Qin, Y. (2009) A Fast Incremental Clustering Algorithm. In *Proceedings of the 2009 International Symposium on Information Processing*: 175–178.

[19] Thapa, N., Liu, L., Lin, P., Wang, J. and Zhang, J. (2011) Constrained Nonnegative Matrix Factorization for Data Privacy. In *Proceedings of the 7th International Conference on Data Mining*: 88–93.

[20] Wang, X. and Zhang, J. (2012) SVD-based Privacy Preserving Data Updating in Collaborative Filtering. In *Proceedings of the World Congress on Engineering 2012* (IAENG): 377–284.

[21] Wang, Y., Liao, X., Wu, H. and Wu, J. (2012) Incremental collaborative filtering considering temporal effects. *http://arxiv.org/abs/1203.5415* .

[22] Xu, W., Liu, X. and Gong, Y. (2003) Document Clustering Based on Non-negative Matrix Factorization. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval* (ACM): 267–273.

[23] Zhang, S., Wang, W., Ford, J. and Makedon, F. (2006) Learning from Incomplete Ratings Using Non-negative Matrix Factorization. In *Proceedings of the 6th SIAM International Conference on Data Mining* (SIAM): 548–552.