Partition-based Collaborative Tensor Factorization for POI Recommendation

Wenjing Luan, *Student Member*, *IEEE*, Guanjun Liu, *Member*, *IEEE*, Changjun Jiang, and Liang Qi, *Student Member*, *IEEE*

***Abstract*—The rapid development of location-based social networks (LBSNs) provides people with an opportunity of better understanding their mobility behavior which enables them to decide their next location. For example, it can help travelers to choose where to go next, or recommend salesmen the most potential places to deliver advertisements or sell products. In this paper, a method for recommending points of interest (POIs) is proposed based on a collaborative tensor factorization (CTF) technique. Firstly, a generalized objective function is constructed for collaboratively factorizing a tensor with several feature matrices. Secondly, a 3-mode tensor is used to model all users' check-in behaviors, and three feature matrices are extracted to characterize the time distribution, category distribution and POI correlation, respectively. Thirdly, each user's preference to a POI at a specific time can be estimated by using CTF. In order to further improve the recommendation accuracy, PCTF (Partition-based CTF) is proposed to fill the missing entries of a tensor after clustering its every mode. Experiments on a real check-in database show that the proposed method can provide more accurate location recommendation.**

***Index Terms*—Clustering, context, feature extraction, point of interest (POI) recommendation, tensor factorization.**

I. Introduction

RECENTLY, location-aware devices such as mobile phones and GPS navigation systems greatly promote the development and wide application of location-based services. Navigational assistance and location sharing become very popular with the prevalence of location-based social networks (LBSNs) [1] such as Foursquare, Gowalla, GeoLife and Weibo. As shown in Fig. 1, an LBSN usually contains two parts: a map (physical-world) and a social network (cyber-space). There are lots of points of interest (POIs) in the physical-world. When a user arrives at a POI, a record named check-in can be generated and released into the cyber-space. Thus, through these LBSNs, people can easily record or share their real-time location as well as the real-life experiences. These LBSNs can closely connect the real world with the virtual internet. By June 2016, Foursquare has recorded over 8 billion check-ins made by more than 50 million people worldwide[1]. Weibo is currently the most popular LBSN in China and has 212 million active users till February 2015, and its users continuously grow every day. Such a huge number of geo-spatially tagged data produced from mobile devices in LBSNs provide researchers a great opportunity to mine, understand and analyze users' interests and behaviors, and thus service the users better.

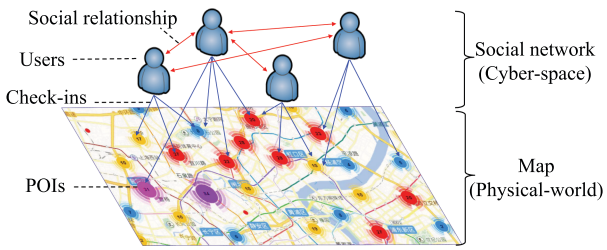

Fig. 1. A typical example of LBSN.

A personalized POI recommendation method generally yields a list of POIs according to several explicit or implicit evaluation scores on POIs [1]. In reality, check-ins made by a user in the cyber-space are not too many although POIs visited by her/him in the physical-world are plentiful. Thus, it is a big challenge for a recommendation method to make an accurate POI recommendation based on these sparse check-in data. Many researches are trying to solve this problem. For example, some recommendation methods have been proposed based on collaborative filtering (CF) and/or matrix factorization (MF) [2]−[8]. These methods can provide recommendation with high prediction accuracy and scalability. However, they ignore some other important factors such as time which is necessary to further enhance the accuracy of personalized POI recommendation. Figs. 2 and 3 show the check-in frequency during different time-slots in one day (one hour is treated as a time-slot), which is the ratio between the number of check-ins in a time-slot and that of all check-ins. We can see that users' check-in behaviors are quite different in different time-slots. For example, according to the check-in data derived from Weibo, it is obvious that people create more check-ins during daytime and before midnight

Manuscript received April 17, 2016; accepted December 20, 2016. This work was supported in part by the National Nature Science Foundation of China (91218301, 61572360), the Basic Research Projects of People's Public Security University of China (2016JKF01316), and in part by Shanghai Shuguang Program (15SG18). Recommended by Associate Editor Tadahiko Murata. *(Corresponding authors: Guanjun Liu and Changjun Jiang.)*

Citation: W. J. Luan, G. J. Liu, C. J. Jiang, and L. Qi, "Partition-based collaborative tensor factorization for POI recommendation," *IEEE/CAA J. of Autom. Sinica*, vol. 4, no. 3, pp. 437−446, Jul. 2017.

W. J. Luan, G. J. Liu, and L. Qi are with the Department of Computer Science and Technology, Tongji University, Shanghai 200092, China (e-mail: wenjingmengjing@163.com; liugj1116@163.com; 1210513@tongji.edu.cn).

C. J. Jiang is with the Key Laboratory of Embedded System and Service Computing, Tongji University, Shanghai 200092, China (e-mail: cjjiang@tongji.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JAS.2017.7510538

[1] https://foursquare.com/about/

than other time-slots. In addition, different POIs have their special time when to be frequently visited. Fig. 3 shows users' check-in frequencies for different categories including Art & Entertainment, Hotel, Shop, and Living Service. We can see that most check-ins take place in a particular time interval. For example, Art & Entertainment are often visited at nearly noon, Shops are welcomed in the afternoon, and Hotel and Living Service are hot places at night. It carries no meaning if a POI is recommended to a user at a wrong time [9]. Notice that Figs. 2 and 3 are from the statistics of our real check-in dataset on Weibo. This work presents a personalized POI recommendation method by fully considering the effect of different time spans on user's check-ins.

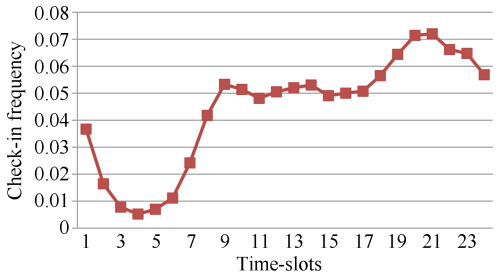

Fig. 2. Users' check-in frequencies in different time-slots.

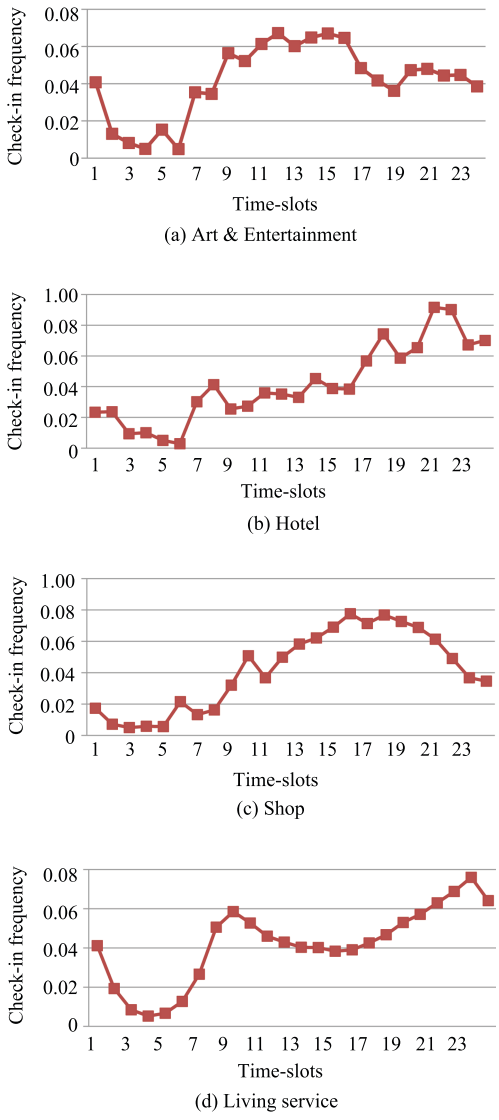

Fig. 3. Check-in frequency in different time-slots in different categories.

Note that the check-in data is usually very rare and sparse. During the period from December 1st, 2012 to September 25th, 2014, over 95 % of users made less than 20 check-ins in total and over 93 % of POIs were checked in less than 10 times. Thus, it is difficult to provide users appropriate POI recommendations by only considering their historical records. In this paper, we focus on a personalized POI recommendation based on a collaborative tensor factorization (CTF) approach [10]. It can be used to recover the missing entries (i.e., without any check-in records). It considers not only the relation among user, POI and time-slots, but also category distribution, time distribution, and the correlations between two POIs. After that, a value of an entry is used as a POI rating score given by a user in each time-slot, and then an accurate POI recommendation can be made.

In practice, there are a large number of users and POIs. A huge and sparse tensor can be constructed by using the whole data. It is difficult and unnecessary to deal with the whole tensor because some users or POIs or time-slots have the same features. Thus, using the similar objects' information to recommend POIs is enough to some extent. This is also the idea of collaborative filtering. Based on this idea, we partition a tensor through a clustering algorithm and then propose a personalized POI recommendation method.

The major contributions of this paper are summarized as follows:

1) We present a generalization of collaborative tensor factorization in order to deal with the sparse data. A framework of simultaneously factorizing an n-mode tensor with m features is presented that can be used for multi-relationship prediction and context-aware recommendation.

2) We use a three-mode tensor to model the correlation among users, POIs and time-slots. The POI recommendation is modeled as a problem of filling the missing entries of a tensor.

3) We propose a partition-based collaborative tensor factorization (PCTF) method to recommend POIs. Firstly, we partition a tensor by a clustering algorithm. Secondly, we recover the tensor by using the CTF over sub-tensors. Finally, a set of POIs are provided to users.

The rest of the paper is organized as follows: Section II reviews the related definitions and a generalization of collaborative tensor factorization. Section III proposes a partition-based collaborative tensor factorization method for POI recommendation method. Experiment design and result evaluations are given in Section IV. Section V introduces the related work. Section VI draws our conclusions and presents future work.

II. Preliminaries

This section recalls POIs, user check-ins, and the POI recommendation problem [11]. Then, an extended collaborative tensor factorization is presented.

A. Problem Definition

Definition 1 (POI): A point of interest (POI) is denoted as $P = [P_{id}, L_1, L_2, C_1, C_2, \ldots, C_k]$, where

1) P_{id} is the unique identifier of the POI;

2) L_1 and L_2 denote the latitude and longitude of the POI, respectively; and

3) $C_1 - C_k$ denote k categories which the POI belongs to.

A POI is a significant venue/location in the physical world, like a shopping mall or a theatre. Usually, a POI belongs to one or more categories such as Entertainment & Arts, Vehicles, Education, Food & Dining, Government, Health & Beauty, Home & Family, Shopping, Sports & Recreation, and Travel.

Definition 2 (User check-in): A check-in is denoted by $ci = [UC_{id}, U_{id}, P_{id}, T]$, where

1) UC_{id}, U_{id}, and P_{id} are the unique identifiers of a check-in, a user, and a POI, respectively; and

2) T is a time stamp.

In LBSNs such as Foursquare and Weibo, a user can mark a POI while visiting it, and thus a check-in is generated.

Definition 3 (POI recommendation): Given a set of check-ins $CI = \{ci_1, ci_2, \ldots, ci_n\}$, a POI recommendation is to provide an ordered list of top-k POIs $LP = (P'_1, P'_2, \ldots, P'_k)$ via CI.

Note that the goal of POI recommendation is to provide POI lists to users according to their preference for different POIs during different time-slots. Thus we need to obtain a user's preferences to all POIs before recommendation. It is assumed that the check-in frequency can characterize a user's visiting preference, i.e., the higher the check-in frequency is, the more the POI is preferred by a user [12]. Based on this assumption, we estimate a user's visiting preference for a POI which is called POI rating by using the related check-in frequency.

B. Tensor Factorization

First, $\mathbb{R}$ is the set of real numbers, $\mathbb{R}^+$ is the set of positive real numbers; $\mathbb{N} = \{0, 1, 2, \ldots\}$ is the natural number set, $\mathbb{N}^+ = \mathbb{N}/\{0\}$, $\mathbb{N}_k = \{0, 1, 2, \ldots, k\}$, and $\mathbb{N}_k^+ = \{1, 2, \ldots, k\}$, $k \in \mathbb{N}^+$.

A tensor is a multi-dimensional array. Generally, a one-mode tensor is a 1-dimension array that is represented as a vector. A two-mode tensor is a 2-dimension array represented by a matrix. In the same way, an n-mode tensor is an n-dimensional array.

Fig. 4 shows an example of 3-mode tensor $X \in \mathbb{R}^{5\times4\times3}$. Its three modes respectively represent users, POIs and time-slots. This "user-POI-time" tensor can be flattened in three ways to obtain matrices comprising its modes. An entry in X is denoted by $x(i, j, k)$ and its value equals the check-in frequency of user u_i in POI p_j at time-slot t_k. Given X, the distribution of check-ins over different POIs visited by a user u_i during a time-slot t_k can be obtained by retrieving the vector $X(i, j, k)$, $j \in \mathbb{N}_4^+$. Users can also be ranked at t_k according to p_j based on $X(i, j, k)$, $i \in \mathbb{N}_5^+$. POIs can be ranked according to overall check-in $\sum_{i=1}^{5}\sum_{k=1}^{3} X_{ijk}$.

Given an n-mode tensor $X \in \mathbb{R}^{D_1\times D_2\times\cdots\times D_n}$, a common approach to recover the missing entries of X is to factorize X into the multiplication of a core tensor $G \in \mathbb{R}^{K_1\times K_2\times\cdots\times K_n}$ and a set of (low-rank) matrices $M_1 \in \mathbb{R}^{D_1\times K_1}$, $M_2 \in \mathbb{R}^{D_2\times K_2}$, $\ldots$, and $M_n \in \mathbb{R}^{D_n\times K_n}$ based on X's non-zero entries [7]. The objective function to control the error of the factorization is defined by the following formula:

$$\begin{aligned} &L(G, M_1, M_2, \ldots, M_n) \\ &\quad = \frac{1}{2}\|X - G \times_1 M_1 \times_2 M_2 \times_3 \cdots \times_n M_n\|^2 \\ &\qquad + \frac{\lambda}{2}\left(\|G\|^2 + \sum_{i=1}^{n}\|M_i\|^2\right) \end{aligned} \tag{1}$$

where $\|\cdot\|^2$ denotes the l_2 norm; $\times$ stands for the matrix multiplication; $\times_i$ denotes the tensor-matrix multiplication and the subscript i represents the ith mode of a tensor; $G \in \mathbb{R}^{K_1\times K_2\times\cdots\times K_n}$ is a tensor, $M_i \in \mathbb{R}^{D_i\times K_i}$ is a matrix, and their multiplication is $(G \times_i M_i) \in \mathbb{R}^{K_1\times\cdots\times K_{i-1}\times D_i\times K_{i+1}\times\cdots\times K_n}$ such that:

$$\begin{aligned} &(G \times_i M_i)_{k_1\times\cdots\times k_{i-1}\times d_i\times k_{i+1}\times\cdots\times k_n} \\ &\quad = \sum_{k_i=1}^{K_i} x_{k_1\times\cdots\times k_{i-1}\times k_i\times k_{i+1}\times\cdots\times k_n} m_{d_i k_i}. \end{aligned}$$

The first part of (1) is to control the factorization error and the second one is a regularization penalty to avoid overfitting. λ is a parameter which denotes the weight of the regularization penalty. Note that $K_1 - K_n$ are usually very small. By minimizing the objective function, we can get a set of optimized matrices $M_1 - M_n$. Afterwards, we can recover the missing entries of X by the following formula:

$$X_{\text{rec}} = G \times_1 M_1 \times_2 M_2 \times_3 \cdots \times_n M_n. \tag{2}$$

If the original tensor is very sparse, it is not accurate enough to fill the missing entries only by using those non-zero entries. In [13], a model of collective matrix factorization is proposed to improve the prediction accuracy by exploiting information from one relation while predicting another. Several matrices are simultaneously factorized, since parameters are shared among factors when an entity participates in multiple relations. Based on this idea, we can collaboratively factorize a tensor with several feature matrices to achieve a higher prediction accuracy of the missing entries in a tensor.

Given an n-mode tensor and several feature matrices, we collaboratively factorize tensor X with feature matrices $F_1 - F_m$. It is worth noting that at least one mode of feature matrix F_i should occur in tensor X.

1) If F_i shares one mode with X, $F_i \in \mathbb{R}^{D_i\times F_i}$ can be factorized into the multiplication of two matrices, i.e., $F_i = F_{i1} \times F_{i2}$, where $F_{i1} \in \mathbb{R}^{D_i\times K_i}$ and $F_{i2} \in \mathbb{R}^{K_i\times F_i}$ are low rank latent factors for F_i. Tensor X and matrix F_i share matrix F_{i1} or F_{i2}.

2) If the two modes of F_i both belong to X, $F_i \in \mathbb{R}^{D_i\times D_j}$ can be factorized into the multiplication of two matrices, i.e., $F_i = F_{i1} \times F_{i2}$, where $F_{i1} \in \mathbb{R}^{D_i\times K_i}$ and $F_{i2} \in \mathbb{R}^{K_i\times D_j}$ are low rank latent factors for F_i. Tensor X and matrix F_i share matrices F_{i1} and F_{i2}.

For the collaborative tensor factorization, the objective function can be defined as follows:

$$L(G, M_1, \ldots, M_n, F_{11}, \ldots, F_{m1}, F_{12}, \ldots, F_{m2})$$

$$= \frac{1}{2}\|X - G \times_1 M_1 \times_2 \cdots \times_n M_n\|^2 + \sum_{i=1}^{m} \frac{\lambda_i}{2}\|F_i - F_{i1} \times F_{i2}\|^2 + \frac{\lambda_0}{2}\left(\|G\|^2 + \sum_{i=1}^{n}\|M_i\|^2 + \sum_{i=1}^{m}\left(\|F_{i1}\|^2 + \|F_{i2}\|^2\right)\right) \tag{3}$$

where $\|X - G \times_1 M_1 \times_2 \cdots \times_n M_n\|^2$ and $\|F_i - F_{i1} \times F_{i2}\|^2$ are respectively used to control the errors of factorizing X and F_i; $\|G\|^2 + \sum_{i=1}^{n}\|M_i\|^2 + \sum_{i=1}^{m}(\|F_{i1}\|^2 + \|F_{i2}\|^2)$ is a regularization penalty to avoid over-fitting; and λ_i, $i \in \mathbb{N}_m$, are parameters denoting the weights of each part during the collaborative factorization. If $\lambda_i = 0$, our model degenerates to the original tucker decomposition method [14]−[16]. Since at least one mode of feature matrix F_i is shared with X, there exists an F_{ij}, $i \in \mathbb{N}_m^+$, $j \in \mathbb{N}_2^+$, that is identical with some M_k, $k \in \mathbb{N}_n^+$.

In order to solve the optimization problem, we use gradient descent algorithm to find a local optimization. Specifically, we use an element-wise optimization algorithm [7], which updates each entry in the tensor independently. After the factorization, we can recover X by (2).

C. Clustering Algorithm

Clustering algorithms are generally used in the unsupervised fashion. By clustering, a set of data instances can be grouped according to some notion of similarity. Such an algorithm needs only the set of object features as its input, but not all label information.

K-means clustering algorithm [17], [18] is a simple and popular method used to automatically partition a set of instances into k clusters. It proceeds by selecting k initial cluster centers and then iteratively refines them as follows:

1) Each instance I_i is assigned to its closest cluster center.

2) Each cluster center CC_i is updated to be the mean of its constituent instances.

3) The algorithm converges when there is no further change in assignment of instances to clusters.

Similarity calculation between instances is a key basis to a clustering algorithm. There are many methods to calculate object similarity, such as Euclidean distance, Manhattan distance, Cosine distance, Mahalanobis distance, and Jaccard coefficient. Cosine distance is a popular way to calculate the vector similarity. For vectors $x = [x_1, x_2, \ldots, x_n]$ and $y = [y_1, y_2, \ldots, y_n]$, their similarity can be calculated by the following equation.

$$\mathrm{Sim}(x, y) = \cos\theta = \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{\|x\| \cdot \|y\|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2} \cdot \sqrt{\sum_{i=1}^{n} y_i^2}}.$$

D. Tensor Partition

If a tensor is very huge and sparse, the correlation degree between entries is usually different. For example, in the "user-POI-time" tensor in Fig. 4, similarities between users (or time-slots or POIs) are different. According to the idea of collaborative filtering, users (or time-slots or POIs) with similar features should bring stronger influences to each other. Thus, in order to recover more accurate entries, a tensor can be partitioned into several small sub-tensors according to its modes before collaboratively factorizing these tensors. For example, if users are partitioned into two sets $(\{u_1, u_2\}, \{u_3, u_4, u_5\})$, time-slots into two sets $(\{t_1\}, \{t_2, t_3\})$, and POIs into two sets $(\{p_1, p_2\}, \{p_3, p_4\})$, then the "user-POI-time" tensor can be partitioned into $2 \times 2 \times 2 = 8$ sub-tensors, as shown in Fig. 5.

III. POI Recommendation Method Based on PCTF

This section proposes a partition-based collaborative tensor factorization (PCTF) method for POI recommendation. We model a "user-POI-time" tensor and extract three feature matrices, and then partition them into sub-tensors and sub-matrices by clustering every mode. Finally, we collaboratively factorize the corresponding sub-tensor with sub-matrices.

A. Framework

Fig. 6 shows the framework of our PCTF-based POI recommendation method, which consists of four layers: 1) data collection and pre-processing, 2) tensor construction and feature extraction, 3) tensor partition and factorization, and 4) personalized POI recommendation.

Firstly, we collect users' check-in data and POI information from Weibo and Dianping websites, and then extract data through processing and filtering. Secondly, we construct a tensor and three feature matrices by using the obtained data. Thirdly, we partition the tensor and matrices through clustering every mode and then recover the tensor by using CTF over sub-tensors and sub-matrices. Finally, we can recommend an ordered POI list to a user according to his/her preference ratings to different POIs in different time-slots which are queried from the recovered tensor. Since the middle layers are the key parts of our method, we especially introduce them in detail in the following sections.

B. Tensor Construction

We represent the preference of a user to a location during a time-slot by a 3-mode tensor $X \in \mathbb{R}^{N \times M \times T}$, where N, M and T denote the counts of users, POIs and time-slots, respectively, as shown in Fig. 4. We give the formal models for users, POIs, and time-slots as follows:

Mode-1 (Users): $U = [u_1, u_2, \ldots, u_i, \ldots, u_N]$ denotes N different users;

Mode-2 (POIs): $P = [p_1, p_2, \ldots, p_j, \ldots, p_M]$ denotes M different POIs;

Mode-3 (Time-slots): $T = [t_1, t_2, \ldots, t_k, \ldots, t_T]$ denotes T different time spans.

We divide a day into 24 time-slots with equal length, and thus the number of slots in the time dimension is fixed. After

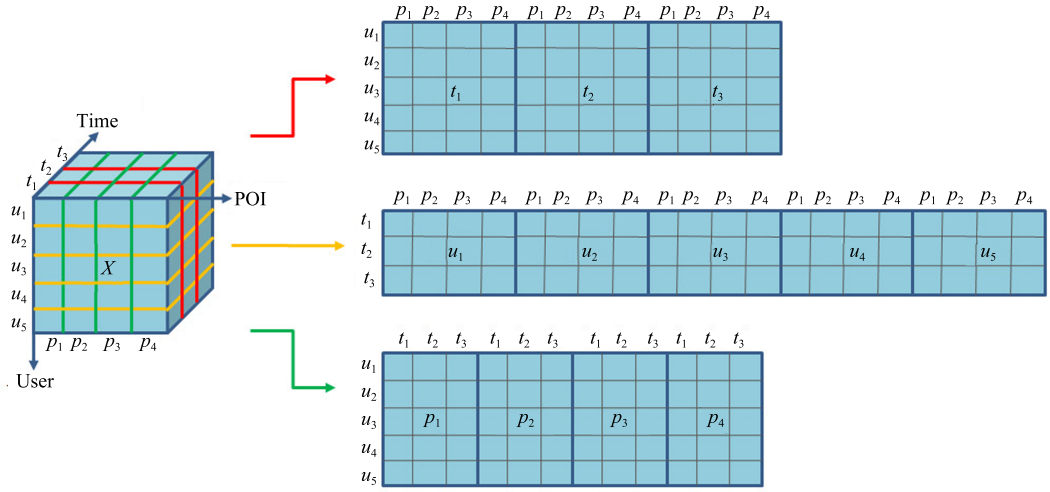

Fig. 4. An example of "user-POI-time" tensor.

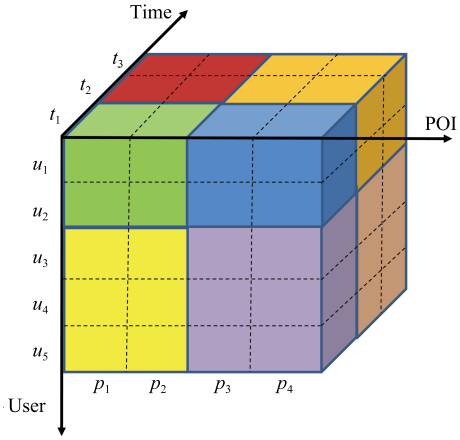

Fig. 5. An example of tensor partition.

projecting the check-ins over a long period into one day, we can calculate the check-in frequency for each time-slot of a day over the period. Thus, an entry $x(i, j, k)$ stores the check-in frequency (i.e., preference rating) of user u_i in p_j in t_k over a long period of time. The value of each entry in tensor X is normalized to [0, 1] for the convenience of factorization.

C. Feature Extraction

Since the constructed "user-POI-time" tensor X is too sparse, factorizing X solely based on its own non-zero entries cannot produce an accurate result. To deal with this problem, we extract and utilize three feature matrices that can be used as contexts in the factorization process. They reflect features of user, time, and POI correlation. The three matrices are denoted by $F_1 - F_3$ and describe the preference correlation between different users, the temporal correlation between different time-slots, and the co-occurrence correlation between different POIs, respectively.

1) User Feature: $F_1 \in \mathbb{R}^{N \times L}$ is the "user-POI category" matrix where $F_1(i, j)$ denotes the number of check-ins of user u_i in POI category c_j. Matrix F_1 gives the distribution of users over different categories, such as Hotel, Shop, and Entertainment & Arts. It incorporates the similarity between two users in terms of their category preferences. Intuitively, users with similar category preference features could have a similar POI check-in situation.

2) Time Feature: $F_2 \in \mathbb{R}^{T \times N}$ is the "time-user" matrix where $F_2(i, j)$ denotes the number of check-ins created in time-slot t_i by user u_j. Matrix F_2 reveals the correlation between different time-slots in terms of the distribution of check-ins of different users. Two time-slots sharing a similar user distribution could have a similar check-in situation.

3) POI Correlation Feature: $F_3 \in \mathbb{R}^{M \times M}$ is the "POI-POI" matrix where $F_3(i, j)$ describes the co-occurrence correlation between POIs p_i and p_j. It is calculated by $F_3(i, j) = |U^i \cap U^j|$. Notice that U^i (resp., U^j) denotes the set of users who have checked in p_i (resp., p_j) and $|U|$ is the number of elements in the set U. Then the value of each entry in matrix F_3 is normalized to [0, 1]. Once the correlation is determined, we can infer the visit probability of other POIs for a user through the user's check-in history.

D. Collaborative Tensor Factorization

Based on the tensor construction and feature extraction methods introduced above, we can obtain the "user-POI-time" tensor $X \in \mathbb{R}^{N \times M \times T}$ and three feature matrices: "user-POI category" matrix $F_1 \in \mathbb{R}^{N \times L}$, "time-user" matrix $F_2 \in \mathbb{R}^{T \times N}$, and "POI-POI" matrix $F_3 \in \mathbb{R}^{M \times M}$.

As shown in Fig. 7, X and $F_1 - F_3$ are simultaneously factorized. The three feature matrices have correlations with different modes of the tensor, respectively. F_1 shares a factor M_1 with X, F_2 shares M_2 with X, and F_3 shares M_2 and M_3 with X. By using the collaborative tensor factorization method described in Section II-B, the objective function of our model is defined as follows:

$$\begin{aligned} &L(G, M_1, M_2, M_3, F_{11}, F_{12}, F_{21}, F_{22}, F_{31}, F_{32}) \\ &\quad = \frac{1}{2}\|X - G \times_1 M_1 \times_2 M_2 \times_3 M_3\|^2 \\ &\qquad + \sum_{i=1}^{3} \frac{\lambda_i}{2}\|F_i - F_{i1} \times F_{i2}\|^2 \\ &\qquad + \frac{\lambda_0}{2}\left(\|G\|^2 + \sum_{i=1}^{3}\|M_i\|^2 + \sum_{i=1}^{3}(\|F_{i1}\|^2 + \|F_{i2}\|^2)\right). \end{aligned} \tag{4}$$

Since $F_1 - F_3$ share factors with X in the factorization, we have that $F_{11} = M_1$, $F_{12} \in \mathbb{R}^{K_1 \times L}$, $F_{21} = M_2$, $F_{22} = M_2^T$,

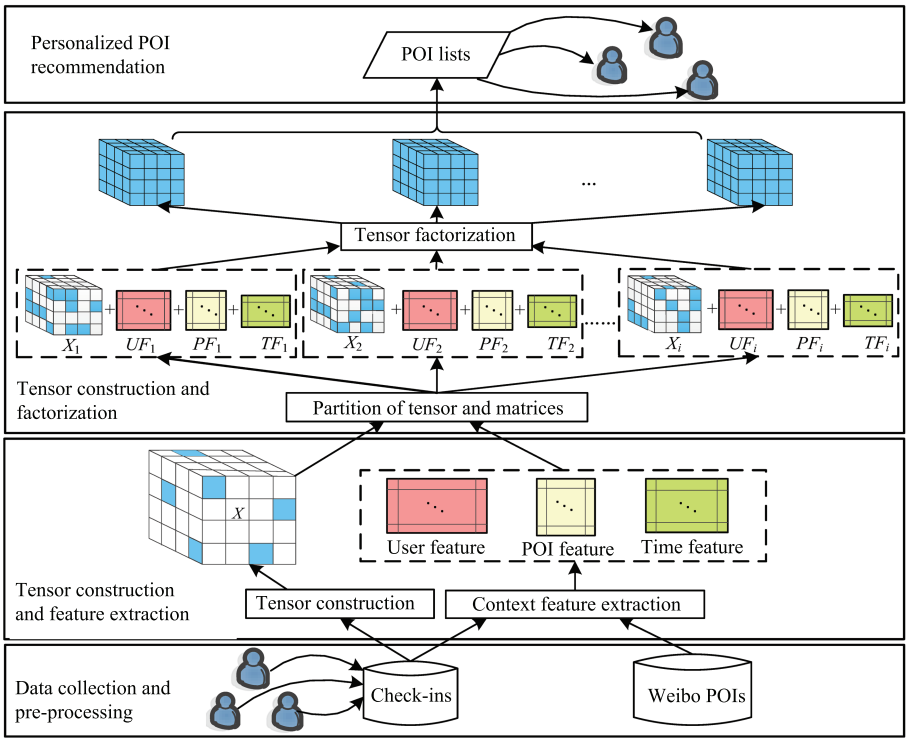

Fig. 6. The framework of PCTF-based POI recommendation (X_i, UF_i, PF_i, and TF_i denote the ith sub-tensor, the ith sub-matrices of user feature matrix, POI correlation matrix and time feature, respectively).

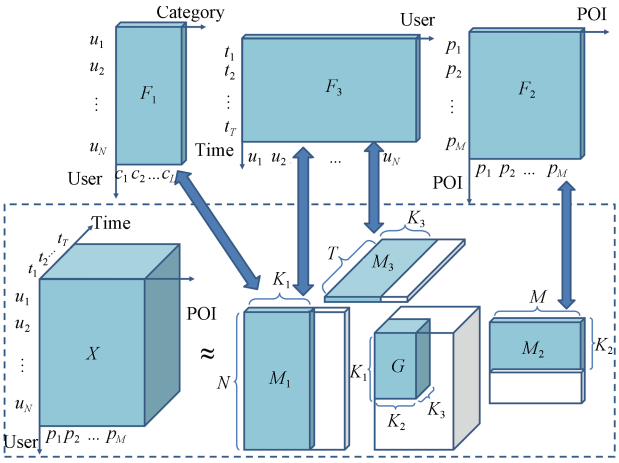

Fig. 7. Collaborative tensor factorization.

$F_{31} = M_3$, and $F_{32} = M_1^T$. Thus the objective function is:

$$\begin{aligned} L(G, M_1, M_2, M_3, F_{12}) &= \frac{1}{2}\|X - G \times_1 M_1 \times_2 M_2 \times_3 M_3\|^2 \\ &+ \frac{\lambda_0}{2}\left(\|G\|^2 + \sum_{i=1}^{3}\|M_i\|^2 + \|F_{12}\|^2\right) \\ &+ \frac{\lambda_1}{2}\|F_1 - M_1 \times F_{12}\|^2 \\ &+ \frac{\lambda_2}{2}\|F_2 - M_2 \times M_2^T\|^2 \\ &+ \frac{\lambda_3}{2}\|F_3 - M_3 \times M_1^T\|^2. \end{aligned} \tag{5}$$

Then, we use an element-wise gradient descent optimization algorithm [7] to solve this optimization problem.

Finally, we can recover X by $X_{\text{rec}} = G \times_1 M_1 \times_2 M_2 \times_3 M_3$. Every user's preferences to different POIs in different time-slots (i.e., the recovered values) can be obtained, and thus an ordered POI list can be recommended to a user according to his/her preferences (i.e., the recovered tensor).

E. Partition-based POI Recommendation

The result of performing CTF on one big tensor is not accurate due to the fact that correlation degree between users (or time-slots or POIs) are not considered in CTF. Therefore, we partition the tensor into sub-tensors according to the related similarities and perform CTF on the corresponding sub-tensors and sub-matrices. Algorithm 1 describes our PCTF algorithm where X is the "user-POI-time" tensor and F_1, F_2, and F_3 are user feature matrix, time feature matrix, and POI correlation feature matrix, respectively. Here, one object is denoted by a feature vector, and we use cosine distance to compute their similarities.

In what follows, we analyze the complexity of our algorithm. During Step 2, we use the K-means clustering algorithm on $F_1 \in \mathbb{R}^{N\times L}$, $F_2 \in \mathbb{R}^{T\times N}$, $F_3 \in \mathbb{R}^{M\times M}$, respectively, and the algorithm complexity is $\mathrm{O}(I_1 N_u N L) + \mathrm{O}(I_2 N_t T N) + \mathrm{O}(I_3 N_p M M)$, where I_i $(i = 1, 2, 3)$ denotes the number of iterations, and N_u, N_t, and N_p are respectively the number of clusters of users, time-slots and POIs. According to [7], if we use CTF on the whole tensor and matrices, the complexity is $\mathrm{O}(\mathbb{K} K_1 K_2 K_3 r_u)$, where $\mathbb{K}$ is the number of ratings and K_i $(i = 1, 2, 3)$ and r_u are the dimensionalities of the factors M_i $(i = 1, 2, 3)$ and F_1, respec-

Algorithm 1 Algorithm of partition-based collaborative tensor factorization

Input:

$X \in \mathbb{R}^{N\times M\times T}$, $F_1 \in \mathbb{R}^{N\times L}$, $F_2 \in \mathbb{R}^{T\times N}$, $F_3 \in \mathbb{R}^{M\times M}$;

Output:

$X_{\mathrm{rec}} \in \mathbb{R}^{N\times M\times T}$;

Step 1: Compute user, time, and POI similarities by cosine distance over F_1, F_2 and F_3, respectively;

Step 2: Cluster users into N_u clusters, time-slots into N_t clusters, and POI into N_p clusters by using K-means algorithm, the user, time-slot and POI similarities are used as the inputs of K-means, respectively;

Step 3: Partition X into $N_u \times N_t \times N_p$ sub-tensors, denoted by S_i, $i \in [1, N_u \times N_t \times N_p]$;

Step 4: Partition F_1, F_2, F_3 into N_u, N_t, N_p sub-matrices, respectively;

Step 5: For each sub-tensor S_i with the corresponding sub-matrices, factorize it using the CTF method;

Step 6: Recover each sub-tensor to obtain a new tensor $S_{i_{\mathrm{rec}}}$;

Step 7: Merge $S_{i_{\mathrm{rec}}}$ $(i \in [1, N_u \times N_t \times N_p])$ and obtain the whole tensor X_{rec};

Step 8: Return X_{rec}.

tively. However, we have partitioned the tensor and matrices into several non-over- lapping sub-tensors (correspondingly, some non-overlapping sub-matrices). Therefore, it is easy to parallelize by performing CTF on them independently that can greatly reduce the execution time.

IV. Experiments

A. Data Collection and Analysis

This section introduces our datasets. They are crawled from two publicly accessed websites: Weibo and DianPing. We collect users' check-in data from Weibo which is the largest social network website in China. In total, we crawl 694 million check-ins created by 390 000 users in 90 000 POIs of Shanghai city from December 1st, 2012 to September 25th, 2014. For each POI, we crawl its description including identifier, geographic coordinates and categories. For the POIs without category labels, we supply the category information according to that from DianPing website. For each check-in, its description includes not only identifiers of the check-in, user, and POI, but also a timestamp.

Since every user's check-in behaviors usually happen with randomness and uncertainty, and the check-in data is too large and sparse, a small dataset is extracted from the raw data to evaluate our method. First, we select all such users whose total check-in number is more than 500. Secondly, from the selected data, we delete the check-ins which are created in the POIs that are in total visited less than 50 times. After preprocessing and filtering, a collection of 17 469 check-ins created by 131 unique users in 106 different POIs are eventually obtained.

We set one hour as a time-slot in this paper. By using the extracted dataset, we construct a "user-POI-time" tensor and its size is $131\times106\times24$. Even though the tensor is constructed based on this preprocessed dataset, it is still very sparse, i.e., only 0.98 % of entries of X have non-zero values.

B. Experiment Design and Evaluation

Some experiments are conducted to evaluate our proposed POI recommendation model by using the extracted dataset. We compare our proposed PCTF method with the matrix factorization approach with time slicing (TMF) [19] and collaborative tensor factorization (CTF). The three methods aim to obtain every user's ratings to every POIs in each time-slot.

1) TMF method is realized as follows: a) project all the check-ins into one day, b) divide all the check-ins into 24 subsets according to their timestamps, e.g., if the timestamp of a check-in is 8:08, then it belongs to the subset in time-slot "8:00−9:00", c) construct 24 "user-POI" matrices of the check-ins in each time-slot, and 4) fill the missing entries by matrix factorization for each "user-POI" matrix.

2) CTF method simultaneously factorizes the "user-POI-time" tensor with other three feature matrices by using the method presented in Section III.

3) PCTF method partitions the "user-POI-time" tensor into a group of sub-tensors, and each feature matrix is partitioned into a group of sub-matrices. For each sub-tensor, it is factorized with corresponding sub-matrices by using the CTF. Finally, the whole recovered tensor is obtained by merging the recovered sub-tensors (see Algorithm 1).

In our experiments, two metrics are used to evaluate the performance of the methods. They are mean absolute error (MAE) and root mean square error (RMSE) [11], where y_{rec} is a recovered value, y is the ground truth, and n is the number of entries. MAE and RMSE can be calculated by (6) and (7), respectively.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - (y_{\mathrm{rec}})_i|}{n} \tag{6}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n} (y_i - (y_{\mathrm{rec}})_i)^2}{n}}. \tag{7}$$

In our experiments, we realize TMF with SVD [19] ($k = 10$) and solve the CTF optimization problem by using element-wise gradient descent optimization algorithm proposed in [7]. In PCTF, we set the numbers of user clusters, time-slot clusters and POI clusters to be 5, 5 and 2, respectively. In the following, we study the effect from the number of clusters through experiments. We set $\varepsilon = 0.02$ and conduct ten groups of experiments. Each group contains ten sub-experiments. In each sub-experiment, we randomly select 70 % of entries as training data, and the remaining 30 % are used as validation data. After completing all the sub-experiments, the result of each group is recorded by computing the average MAE and RMSE of

its ten sub-experiments. Fig. 8 presents the performance of the three approaches. We can see that both CTF and PCTF outperform TMF. The reason is that the TMF method only uses the factorization of "user-POI" matrix which neglects the correlations among different time-slots. In addition, not only the time correlations in tensor but also the features of users, POIs and time-slots are leveraged in CTF-based methods.

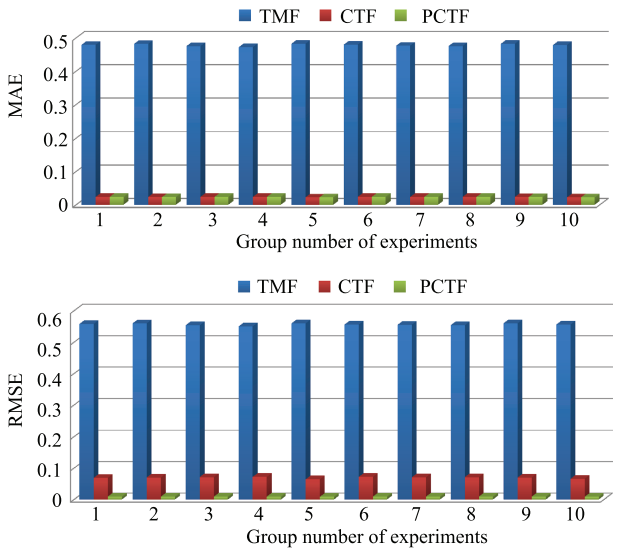

Fig. 8. Performance comparison of TMF, CTF and PCTF.

In order to study the influence from the number of clusters to the recommendation results, we conduct the following experiments. When the numbers of clusters are set differently for users (or time-slots or POIs), the MAE and RMSE change accordingly, as shown in Figs. 9 and 10. In Figs. 9 and 10, $N_u \times N_p \times N_t$ denotes the numbers of user-clusters, POI-clusters and time-slot-clusters are N_u, N_p, N_t, respectively. They indicate that the number of sub-tensors should be decided according to different datasets. However, PCTF always outperforms TMF and CTF.

V. Related Work

A. Applications of Tensor Factorization

In urban computing [20], tensor factorization method has been widely used. For example, in order to realize different kinds of recommendations, some context-aware tensor factorization methods have been proposed by leveraging additional information, such as the activity-activity correlation and geographical features of a location [21]−[23]. Urban refueling behavior has been inferred together with POI data, traffic features, and gas stations' contextual features [24], [25]. Travel time of a road segment without being traversed by trajectories in current time-slot has also been estimated through a context-aware tensor factorization approach [21]. Based on the tensor factorization method, the fine-grained noise situation of different times of a day for each region of NYC has been inferred by using the 311 compliant data together with social media, road network data, and POIs [11]. To the best of our knowledge, the personalized POI recommendation method of collaborative tensor factorization has not been presented yet.

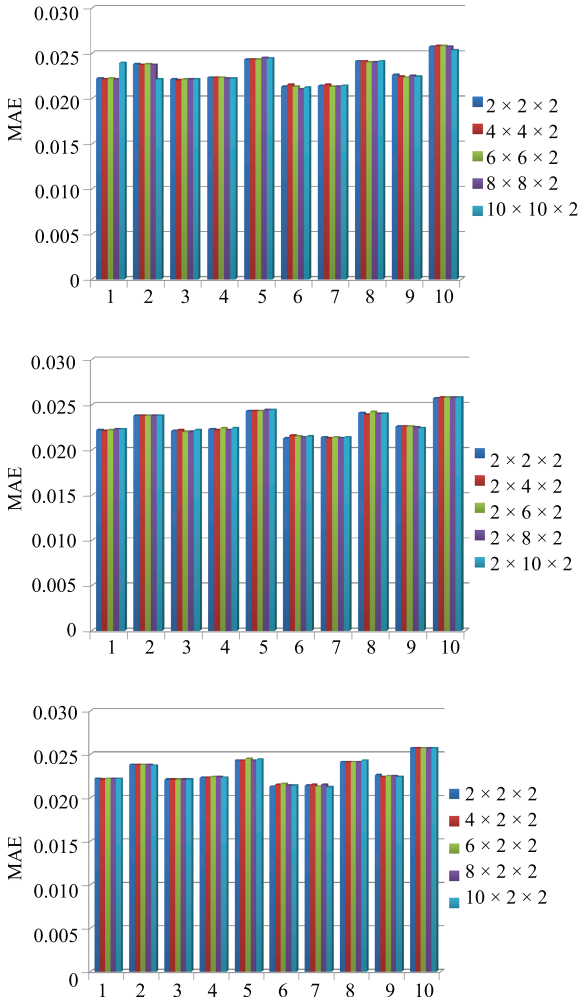

Fig. 9. MAE under different numbers of sub-tensors.

B. Context-aware POI Recommendation

Recently, many contexts have been considered in the model and many context-ware recommendation methods have been proposed to improve the effectiveness of personalized POI recommendation. For example, Hsieh *et al*. [9] consider time factor into route recommendation. Ye *et al*. [26], [27] use a linear fusion framework to integrate the social and geographical characteristics of users and locations to recommend the next location. Zheng *et al*. [28] use GPS data and comments at various locations to discover interesting locations and possible activities that can be performed for recommendation. Cheng *et al*. [29] fuse matrix factorization with geographical and social influence for POI recommendation. However, there is no general framework that has the capacity to model different contexts.

C. Block-based Tensor Factorization

In practice, two toolboxes are widely used for tensor manipulation: one is the Tensor Toolbox for MATLAB [30] (for sparse tensors) and another is an N-way Toolbox for MATLAB [31] (for dense tensors). Besides, various block-based algorithms [32]−[35] and systems have been developed due to the significant cost of tensor factorization. Various parallelization strategies, such as map-reduce-based implementation [33], and

sampling-based method [34] are proposed to perform tensor factorization [35]. In [36], a personalized tensor decomposition mechanism for considering the user's focus is proposed to improve the accuracy and reduce the overall decomposition time. Based on these works, we partition the tensor by clustering its every mode, factorize each sub-tensor with corresponding sub-matrices by using collaborative tensor factorization method, and then recover the whole tensor.

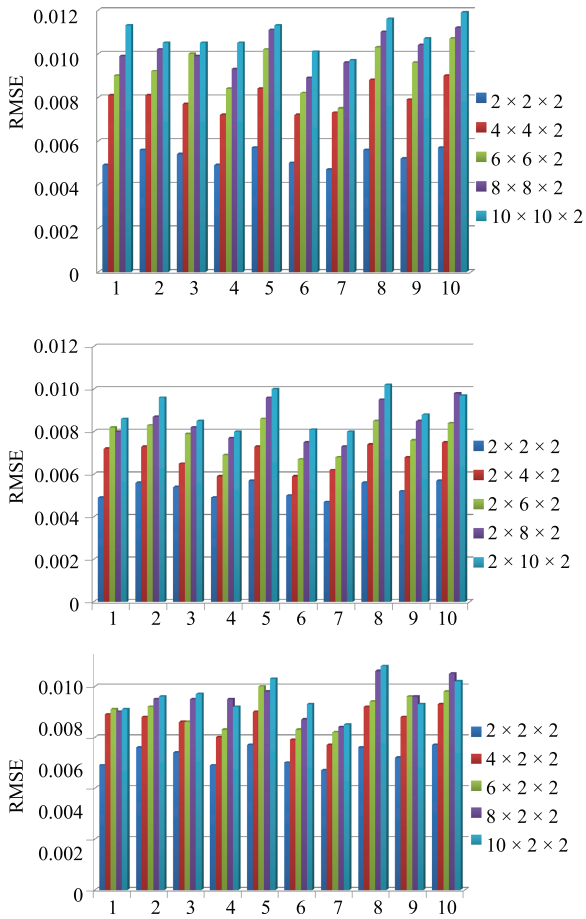

Fig. 10. RMSE under different numbers of sub-tensors.

VI. Conclusions

In this paper, a generalization model of collaborative tensor factorization is presented. In order to realize POI recommendation, we model all users' check-in behaviors as a 3-mode "user-POI-time" tensor and construct three feature matrices from different perspectives. Then, we recover users' POI preferences by using the partition-based collaborative tensor factorization method. Compared with the method using "user-POI" matrix factorization with time slicing, CTF-based methods can provide more accurate recommendation.

Note that this paper is an extended version of our conference version [10]. Based on the work in [10], the tensor-partition module is proposed and then the partition-based CTF algorithm is developed. Experiments illustrate that PCTF can obtain better results in comparison with TMF and CTF.

Our future work focuses on the impacts of context features on the recommendation results.

References

[1] Y. Zheng, X. Xie, and W. Y. Ma, "GeoLife: a collaborative social networking service among user, location and trajectory," *IEEE Data Eng. Bull.*, vol. 33, no. 2, pp. 32−40, 2010.

[2] L. Y. Lv, M. Medo, C. H. Yeung, Y. C. Zhang, Z. K. Zhang, and T. Zhou, "Recommender systems," *Phys. Rep.*, vol. 519, no. 1, pp. 1−49, Oct. 2012.

[3] X. Luo, M. C. Zhou, Y. N. Xia, and Q. S. Zhu, "An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems," *IEEE Trans. Ind. Inform.*, vol. 10, no. 2, pp. 1273−1284, May 2014.

[4] D. D. Lee and H. S. Seung, "Unsupervised learning by convex and conic coding," in *Proc. 9th Int. Conf. Neural Information Processing Systems*, Cambridge, MA, USA, 1997, pp. 515−521.

[5] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, no. 6755, pp. 788−791, May 1999.

[6] N. Guan, D. Tao, Z. Luo, and B. Yuan, "Online nonnegative matrix factorization with robust stochastic approximation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1087−1099, Jul. 2012.

[7] A. Karatzoglou, X. Amatriain, L. Baltrunas, and N. Oliver, "Multiverse recommendation: N-dimensional tensor factorization for context-aware collaborative filtering," in *Proc. 4th ACM Conf. Recommender Systems*, New York, NY, USA, 2010, pp. 79−86.

[8] J. Li, G. J. Liu, C. J. Jiang, and C. G. Yan, "A hybrid method of recommending POIs based on context and personal preference confidence," in *Proc. 3rd IEEE/ACM Int. Conf. Big Data Computing, Applications and Technologies*, Shanghai, China, 2016, pp. 287−292.

[9] H. P. Hsieh, C. T. Li, and S. D. Lin, "Measuring and recommending time-sensitive routes from location-based data," *ACM Trans. Int. Syst. Technol.*, vol. 5, no. 3, Article No. 45, Jun. 2014.

[10] W. J. Luan, G. J. Liu, and C. J. Jiang, "Collaborative tensor factorization and its application in POI recommendation," in *Proc. 13th Int. Networking, Sensing, and Control*, Mexico City, Mexico, 2016, pp. 28−30.

[11] Y. Zheng, T. Liu, Y. L. Wang, Y. M. Liu, Y. M. Zhu, and E. Chang, "Diagnosing New York City's noises with ubiquitous data," in *Proc. 2014 ACM Int. Joint Conf. Pervasive and Ubiquitous Computing*, Seattle, WA, USA, 2014, pp. 715−725.

[12] X. T. Li, G. Cong, X. L. Li, T. A. N. Pham, and S. Krishnaswamy, "Rank-GeoFM: a ranking based geographical factorization method for point of interest recommendation," in *Proc. 38th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Santiago, Chile, 2015, pp. 433−442.

[13] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proc. 14th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Las Vegas, Nevada, USA, 2008, pp. 650−658.

[14] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455−500, Sep. 2009.

[15] L. R. Tucker, "Implications of factor analysis of three-way matrices for measurement of change," in *Problems in Measuring Change*, C. W. Harris, Ed. Madison, Wisconsin, USA: University of Wisconsin Press, 1963, pp. 122−137.

[16] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279−311, Sep. 1966.

[17] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Mathematical Statistics and Probability*, Berkeley, CA, USA, 1967, pp. 281−297.

[18] X. S. Lu and M. C. Zhou, "Analyzing the evolution of rare events via social media data and k-means clustering algorithm," in *Proc. 13th Int. Conf. Networking, Sensing, and Control*, Mexico City, Mexico, 2016.

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

[20] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: concepts, methodologies, and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 3, Article No. 38, Sep. 2014.

[21] Y. L. Wang, Y. Zheng, and Y. X. Xue, "Travel time estimation of a path using sparse trajectories," in *Proc. 20th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, New York, USA, 2014, pp. 25−34.

[22] V. W. Zheng, B. Cao, Y. Zheng, X. Xie, and Q. Yang, "Collaborative filtering meets mobile recommendation: a user-centered approach," in *Proc. 24th AAAI Conf. Artificial Intelligence*, Atlanta, GA, USA, 2010, pp. 236−241.

[23] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang, "Towards mobile intelligence: learning from GPS history data for collaborative recommendation," *Artif. Intell. J.*, vol. 184−185, pp. 17−37, Jun. 2012.

[24] F. Z. Zhang, D. Wilkie, Y. Zheng, and X. Xie, "Sensing the pulse of urban refueling behavior," in *Proc. 2013 ACM Int. Joint Conf. Pervasive and Ubiquitous Computing*, Zurich, Switzerland, 2013, pp. 13−22.

[25] J. B. Shang, Y. Zheng, W. Z. Tong, E. Chang, and Y. Yu, "Inferring gas consumption and pollution emission of vehicles throughout a city," in *Proc. 20th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, New York, USA, 2014, pp. 1027−1036.

[26] M. Ye, P. F. Yin, and W. C. Lee, "Location recommendation for location-based social networks," in *Proc. 18th SIGSPATIAL Int. Conf. Advances in Geographic Information Systems*, San Jose, CA, USA, 2010, pp. 458−461.

[27] M. Ye, P. F. Yin, W. C. Lee, and D. L. Lee, "Exploiting geographical influence for collaborative point-of-interest recommendation," in *Proc. 34th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Beijing, China, 2011, pp. 325−334.

[28] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang, "Collaborative location and activity recommendations with GPS history data," in *Proc. 19th Int. Conf. World Wide Web*, Raleigh, NC, USA, 2010, pp. 1029−1038.

[29] C. Cheng, H. Yang, I. King, and M. R. Lyu, "Fused matrix factorization with geographical and social influence in location-based social networks," in *Proc. 26th AAAI Conf. Artificial Intelligence*, Toronto, ON, Canada, 2012, pp. 17−23.

[30] B. W. Bader and T. G. Kolda, *MATLAB Tensor Toolbox Version 2.5* [Online]. Available: http://www.sandia.gov/~tgkolda/TensorToolbox, Accessed on: Jan., 2014.

[31] C. A. Andersson and R. Bro, "The *n*-way toolbox for MATLAB," *Chemom. Intell. Lab. Syst.*, vol. 52, no. 1, pp. 1−4, Aug. 2000.

[32] C. Chen, D. S. Li, Y. Y. Zhao, Q. Lv, and L. Shang, "WEMAREC: accurate and scalable recommendation through weighted and ensemble matrix approximation," in *Proc. 38th Int. ACM SIGIR Conf. Research and Development in Information Retrieval*, Santiago, Chile, 2015, pp. 303−312.

[33] U. Kang, E. Papalexakis, A. Harpale, and C. Faloutsos, "GigaTensor: scaling tensor analysis up by 100 times-algorithms and discoveries," in *Proc. 18th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, Beijing, China, 2012, pp. 316−324.

[34] E. E. Papalexakis, C. Faloutsos, and N. Sidiropoulos, "Parcube: sparse parallelizable tensor decompositions," in *Proc. 2012 European Conf. Machine Learning and Knowledge Discovery in Databases*, Bristol, UK, 2012, pp. 521−536.

[35] A. H. Phan and A. Cichocki, "Block decomposition for very large-scale nonnegative tensor factorization," in *Proc. 3rd IEEE Int. Workshop Computational Advances in Multi-Sensor Adaptive Processing*, Aruba, Dutch Antilles, 2009, pp. 316−319.

[36] X. S. Li, S. Y. Huang, K. S. Candan, and M. L. Sapino, "Focusing decomposition accuracy by personalizing tensor decomposition (PTD)," in *Proc. 23rd ACM Int. Conf. Information and Knowledge Management*, Shanghai, China, 2014, pp. 689−698.

Wenjing Luan (S'16) received the B.S. and M.S. degrees from Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively. She is now pursuing the Ph.D. degree in the Department of Computer Science and Technology, Tongji University, Shanghai, China. She received the Best Student Paper Award-Finalist in the 13th IEEE International Conference on Networking, Sensing and Control (ICNSC 2016) Conference. Her current research interests include location-based social networks, recommender system, and intelligent transportation systems.

Guanjun Liu (M'16) received the Ph.D. degree in computer software and theory from Tongji University, Shanghai, China, in 2011. He was a postdoctoral research fellow at Singapore University of Technology and Design, Singapore, from 2011 to 2013. He worked at Humboldt-University zu Berlin, Germany, from 2013 to 2014 as a postdoctoral research fellow supported by the Alexander von Humboldt Foundation. He is currently an associate professor with the Department of Computer Science and Technology, Tongji University. He has published 50+ papers. His research interests include Petri net theory, model checking, web service, workflow, discrete event systems, and information security.

Changjun Jiang received the Ph.D. degree from the Institute of Automation, Chinese Academy of Science, Beijing, China, in 1995. He is currently the leader of the Key Laboratory of the Ministry of Education for Embedded System and Service Computing, Tongji University, Shanghai, China. He is an IET Fellow and an Honorary Professor with Brunel University, London. He has published more than 300 papers in journals and conference proceedings, including *Chinese Science*, the *IEEE Transactions on Robotics and Automation*, and the *IEEE Transactions on Fuzzy Systems*. He has led over 30 projects supported by the National Natural Science Foundation of China, the National High Technology Research and Development Program of China, and the National Basic Research Developing Program of China. His research interests include concurrency theory, Petri nets, formal verification of software, cluster, grid technology, intelligent transportation systems, and service-oriented computing. Prof. Jiang has been the recipient of one international prize and seven domestic prizes in the field of science and technology.

Liang Qi (S'16) received the B.S. and M.S. degrees from Shandong University of Science and Technology, Qingdao, China, in 2009 and 2012, respectively. He is currently working toward the Ph.D. degree with the Department of Computer Science and Technology, Tongji University, Shanghai, China. He has been supported by a scholarship from the China Scholarship Council. From 2015 to 2016, he was a joint Ph.D. Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He has authored over a dozen technical papers in journals and conference proceedings. His research interests include Petri nets, recommender system, and intelligent transportation systems.