Hindawi
Scientific Programming
Volume 2018, Article ID 3907804, 11 pages
https://doi.org/10.1155/2018/3907804

Research Article

Improving POI Recommendation via Dynamic Tensor Completion

Jinzhi Liao,[1] Jiuyang Tang,[1,2] Xiang Zhao ,[1,2] and Haichuan Shang[3,4]

[1]*Key Laboratory of Science and Technology on Information System Engineering, National University of Defense Technology, Changsha, China*
[2]*Collaborative Innovation Center of Geospatial Technology, Wuhan, China*
[3]*National Institute of Information and Communications Technology, Tokyo, Japan*
[4]*Institute of Industrial Science, The University of Tokyo, Tokyo, Japan*

Correspondence should be addressed to Xiang Zhao; xiangzhao@nudt.edu.cn

Received 22 May 2018; Revised 30 July 2018; Accepted 7 August 2018; Published 13 November 2018

Academic Editor: Autilia Vitiello

POI recommendation finds significant importance in various real-life applications, especially when meeting with location-based services, e.g., check-ins social networks. In this paper, we propose to solve POI recommendation through a novel model of dynamic tensor, which is among the first triumphs of its kind. In order to carry out timely recommendation, we predict POI by utilizing a completion algorithm based on fast low-rank tensor. Particularly, the dynamic tensor structure is complemented by the fast low-rank tensor completion algorithm so as to achieve prediction with better performance, where the parameter optimization is achieved by a pigeon-inspired heuristic algorithm. In short, our POI recommendation via the dynamic tensor method can take advantage of the intrinsic characteristics of check-ins data due to the multimode features such as current categories, subsequent categories, and temporal information as well as seasons variations are all integrated into the model. Extensive experiment results not only validate the superiority of our proposed method but also imply the application prospect in large-scale and real-time POI recommendation environment.

1. Introduction

The mobile recommendation system has played an indispensable role in people's daily life. For instance, people naturally put up mobile phone to find the restaurants to eat, stores to go, and amusement parks to spare the time. With the proliferation of global positioning system (GPS), location-based social networking services (LBSNs), e.g., Foursquare, Facebook Places, Google Places, and so on, tighten the relationships among people. Through these platforms, people can easily access the information of popular points of interest (POI) which might well cater to their preferences. Therefore, comprehensive analysis of POIs via online check-ins data can make recommendation more accurate and suitable for people's need.

Accurate and prompt recommendation is the key component of an ideal POI recommender system. For example, when a user just leaves the train station at summer night, hotels and restaurants, rather than bars and nightspots, should be recommended due to the common sense that searching for places to settle down is the priority even though he or she is a king or queen of nightclub. The situation can be briefly described in Figure 1. Subsequently, after a short rest, bars or nightspots will appear in the user's recommendation list when he/she tries for another search. Plus, if the user arrives at noon, the system should accordingly recommend dessert shop or cafe instead of nightspots. Also, different seasons may lead to different recommendation in the same situation, such as hot pot restaurants in the freezing winter and cold drinks shops in the scorching summer.

Credible POI recommendation heavily relies on user check-ins data. In order to meet the real-time nature of the task, conventional methods usually resort to matrix factorization to solve the sparsity of user-POI matrices [1, 2]. However, intrinsically speaking, check-ins data contain

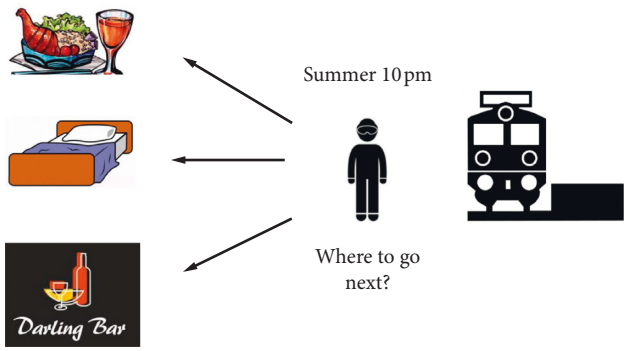

Figure 1: Sketch of POI recommendation problem.

multimode information, which is a feature overlooked by existing methods. To put it in other words, the methods based on matrix factorization suffers from the limitation that the information of check-ins data is not fully made use of and the usage mainly focuses on low-dimensional patterns, thereby sacrificing the high-dimensional patterns of data.

Intuitively, if a model handles data with higher dimensions, it can mine more information and the prediction result can accordingly become more accurate and reliable [3]. It has been shown in recent studies that tensor, a matrix with high-dimensional expansion, can better explain multimode data in comparison to other structures [4]. Consequently, the methods based on tensor are proposed to tackle POI recommendation problems. Since data are fixed in different dimensions, for instance, only users' current location and next location data are presented as a 2-way tensor, and the similarity are calculated between users or between locations. Based on the similarity, the model recommends the locations visited by other users. Owing to higher number of appended dimensions, the prediction performance of tensor-based methods overmatches the matrix-based methods [5].

Consequently, there are many methods based on tensor factorization put forward to improve the performance. Cheng et al. first added the successive time-stamp in the POI recommendation. They put forward the next personalized POI recommendation problem; a check-in tensor is constructed and factorizing personalized Markov chain (FBMC) is harnessed [6] to restrict the movement. To recommend to users the most possible successive POIs, a spatial-temporal latent ranking model, which highlights the significance of time, is put forward by Zhao et al. [7]. Nonetheless, previous methods neglect the personalized factor and fail to combine the location information with real-time situations. Li et al. [5] proposed time-aware factorizing personalized Markov chain (TA-FBMC), where they adopt a 4-dimension tensor to store spatial and temporal characteristics of check-ins data. The relevance between two successive check-ins is explored, and a time-decay factor is leveraged to weigh short-time interval and long-time interval check-ins data in prediction.

Despite the superiority of TA-FBMC, there are at least two shortcomings:

(i) *Model Complexity*. The whole model is constructed upon a static tensor, which means they put all check-ins data in each calculation. Although TA-FBMC contains the time-decay factor to decline the effect of primal data, the computing complexity is the same and the value of the factor is hard to decide.

(ii) *Redundancy Caused by User Dimension*. The user is regarded as an extra dimension, which not only artificially expands the scale of data but also diverts from the fact that people tend to seek for uniformity in the big data background. Additionally, when a new user enters the recommender system, the personalized feature fails to yield promising performance.

Consequently, a clear gap between existing research and potential application can be identified.

In this article, we investigate to close the gap by developing a more accurate approach, namely, Prido (POI recommendation via dynamic tensor), which makes full advantage of multimode check-ins data so as to improve the performance of POI recommendation. Specifically, in accordance with the successful application of tensor in the field of recommendation, we conceive a dynamic tensor model for POI recommendation. On top of it, we also develop a heuristic tensor completion method, which is on the basis of low-rank tensor completion approach. In contrast to existing approaches, Prido considers more features—current category mode, next category mode, month mode, and temporal mode—of check-ins data, and leverages more advanced tensor completion algorithm with effective optimization strategy, for higher accuracy and efficiency. As to empirical assessment, we conduct extensive experiments on real-world data and compare the proposed method with a number of state-of-the-art methods. The empirical results prove that Prido remarkably improves the overall performance in comparison to the competitors.

1.1. Contributions. As a nutshell, the contributions of the article can be summarized to four ingredients:

(a) We propose to model POI recommendation with a dynamic tensor structure to exploit all available feature aspects, which is, to our best knowledge, among the first attempts

(b) A 4-dimension tensor is constructed to capture users' preference between two successive categories in different seasons and different hours

(c) The category information is recommended by leveraging a fast low-rank tensor completion method, equipped with pigeon-inspired algorithm to optimize the parameters and

(d) The proposed method is validated on real-world check-ins datasets and demonstrated to offer remarkable improvement over alternatives as far as efficiency and accuracy are concerned

1.2. Organization. Section 2 overviews related works on various recommendation methods in the domain of POI recommendation. After necessary background knowledge, the proposed method Prido is introduced in Section 3 including specific model and algorithms. Experimental studies are reported in Section 4, and the conclusion is elaborated in Section 5.

2. Related Work

As a special form of item recommendation, POI recommendation, which converts the online check-ins data into physical behavior suggestions, helps both academic and industrial development. Thus, a large number of methods are devoted to dealing with POI recommendation task.

Collaborative filtering (CF) was used to handle the POI recommendations problem in [8], which then became a widely used technique and was developed persistently. CF-based systems can be roughly divided into two parts, i.e., memory-based CF systems and model-based CF systems. In the former, there are also two subclasses, i.e., (1) user-based systems, which calculated the similarity between each pair of users [9] and (2) item-based systems, which computed the similarity between each pair of items [10]. As for model-based CF, it was developed in another direction, in which data mining techniques were added into the system so as to boost performance. Cheng et al. proposed a method which fused matrix factorization with geographical and social influence factors for POI recommendation [11]. Meanwhile, Gao et al. utilized the social network information to deal with the cold-start problems [12]. Noteworthy is that these CF models could not reveal the multifeatures of check-ins data. Therefore, the methods failed to deal with the data sparsity issue.

Recently, people pay more attention to the expansion of factors so as to enhance the explanatory power of the model. Based on Bayesian network, Park et al. introduced user profile to evaluate the matching between user profiles and restaurant profiles and output recommendation results in accordance to the matching scores [13]. Then, Ye et al. took advantage of a power-law distribution, including geographical influence, to achieve more accurate POI recommendation [14]. A combination of kernel density estimation and geographical influence was proposed by Chow to model the check-in behavior [15]. Zhang and Chow adopted a contrived gravity model to exploit the spatiotemporal sequential influence on location recommendations [16]. Yuan et al. incorporated the time factor when computing the similarity between two users in terms of the historical check-ins at the same time [17]. The significance of social impact on POI recommendation has also been presented in previous studies, through which the quality of long-term recommendation can somehow be improved [18]. With geographical correlations, social correlations, and categorical correlations among users and POIs taken into consideration, Zhang and Chow introduced a kernel estimation method to tackle POI recommendation [19]. Transforming the linear model into a nonlinear model, Zhang et al. proposed a matrix factorization method to learn the most potential interactions among two or more attributes [20]. However, previous models all transform the POI problems into a geographical feature in traditional item recommendation tasks, where the intrinsic relations of check-ins data are neglected.

Despite the boom of neural networks and their successful applications in many fields, there is little progress in the POI recommendation domain, for the heterogeneity nature of check-ins data, where the transformation of input may increase the complexity of the model. Among various neural networks, recurrent neural networks (RNN) have been proved to outperform other methods in modeling sequential data of arbitrary length with its recurrent calculation of hidden representation [21]. Chen et al. proposed to detect users' interests from their location-based tweet and then established the mapping between locations and user interests [22]. Liu et al. modeled users' check-ins data in a sequential manner and then utilized RNN to solve the recommendation problem [23]. In the mean time, Zhang et al. proposed nonlinear transformation to extract the user-based and POI-based spatial intents, respectively, to tackle the cold-start problem [24]. Among others, graph-based approaches [25] may also be applied to location-based social networks but not yet in POI recommendation.

Over the recent years, tensor, the high-dimensional expansion matrix, is proved to have prominent advantage when compared with other methods in terms of explaining data with multimode [3, 4], for instance, it has been successfully applied to traffic flow prediction [26]. Illuminated by traditional tensor completion methods, new solutions such as CP, Tucker, tensor train [27], and tensor network [28] were proposed to optimize the structure of tensor methods. Undoubtedly, it is hard for a solution to overmatch the others over all scenarios since different POI recommendation tasks require different models to capture the core of problem and the fittest models ought to deal with diverse realistic demands. In order to fully exploit the latent patterns of data, we proposed Prido in this paper, which is validated to achieve promising outcomes.

3. Methodology

In this section, the dynamic tensor model is elaborated, followed by the introduction of the tensor completion algorithm for personalized tensor, followed by the pigeon-inspired parameter optimization procedure.

3.1. Dynamic Tensor Model for Category Prediction. The fundamental knowledge of tensor is detailed at first, and the dynamic tensor model designed for personalized recommendation is then presented.

3.1.1. Tensor Basics. Tensor is a high-dimensional data representation, the expression of which can be 1-mode (vector) and 2-mode (matrix). A tensor with n-mode is denoted as $\mathbf{X} \in \mathcal{R}^{I_1 \times I_2 \times \ldots \times I_n}$, where I_n represents mode n

quantity, and the specific elements are $x_{(I_1,\ldots,I_k)}$, where $1 \le k \le n$. As for the *matriculating* operator, the target of which is to unfold a tensor into a matrix is denoted as $\text{unfold}(\mathbf{X}, n) = X_{(n)}$ and the elements of the tensor $(I_1, I_2, \ldots, I_n)$ is mapped to the matrix element (I_n, J), where

$$J = \prod_{m=1, m\neq n}^{k-1} I_m. \tag{1}$$

The reverse of matriculation is represented by $\text{fold}(X_{(n)}, n) = \mathbf{X}$ similarly.

Specifically, the primary task of tensor unfolding is dimension reduction, turning it into a matrix. Instead of sampling eigenvalues simply from one order after another, tensor unfolding samples them from different orders in an alternating way to realize the transmission and the blending among eigenvalues from different orders in a tensor. For example, unfolding a tensor of $4 \times 3 \times 2$ in the first dimension will get a 4×6 matrix, of which the six columns are constituted from the second and the third orders' eigenvalues alternately. As regards to tensor folding, it is the inverse operation of unfolding.

The product of two tensors with the same size $\mathbf{A}, \mathbf{B} \in \mathscr{R}^{(I_1 \times I_2 \times \cdots \times I_N)}$ is defined as the sum of the products of their entries,

$$(\mathbf{A}, \mathbf{B}) = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_N} a_{(i_1, i_2, \ldots, i_n)} b_{(i_1, i_2, \ldots, i_n)}. \tag{2}$$

With regard to any $1 \le n \le N$, the product between matrix $M \in \mathscr{R}^{J \times I_n}$ and tensor $\mathbf{A} \in \mathscr{R}^{(I_1 \times I_2 \times \cdots \times I_N)}$ is denoted as $\mathbf{A} \times_n M$, which is further converted to the product of matrices,

$$\mathbf{Y} = \mathbf{A} \times_n M \Longleftrightarrow Y_{(n)} = M A_{(n)}. \tag{3}$$

The Frobenius norm of a tensor is denoted by $||\mathbf{X}||_F = \sqrt{(\mathbf{X}, \mathbf{X})}$. Evidently, $\mathbf{X}_F = ||X_{(k)}||$.

3.1.2. Tensor Stream. Tensor stream is represented by a series of tensors, denoted by $(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_T)$, where each $\mathbf{X}_t \in \mathscr{R}^{(I_1 \times I_2 \times \cdots \times I_m)}$, with $1 \le t \le N$. The series is indexed by time, which is presented in Figure 2. Dynamic tensor blocks $\mathbf{D}(t) = (\mathbf{X}_{\mathbf{T}_0}, \ldots, \mathbf{X}_{\mathbf{T}_t})$ with each $\mathbf{X}_{\mathbf{T}_t} \in \mathscr{R}^{I_1 \times I_2 \times \cdots \times I_{m+1}}$ are denoted by means of the combination of tensors in tensor stream from the initial one, as Figure 3 shows.

Since the Foursquare datasets possess temporal traits, evidently the adjacent historical data are crucial for improving overall performance. As a result, the prediction of category recommendation can be transformed into tensor structure completion task.

3.1.3. Dynamic Tensor. Our goal is to offer specific user POI recommendations during the selected time period, on the basis of the existing data. Considering that the category dataset merely contains check-ins data, by comparing FBMC [6], FBMC-LR [29], and TA-FBMC [5], we choose the probability formula in TA-FBMC to perform data transformation. Adopting the average of locations probability, which reduces repetitive computation of categories,

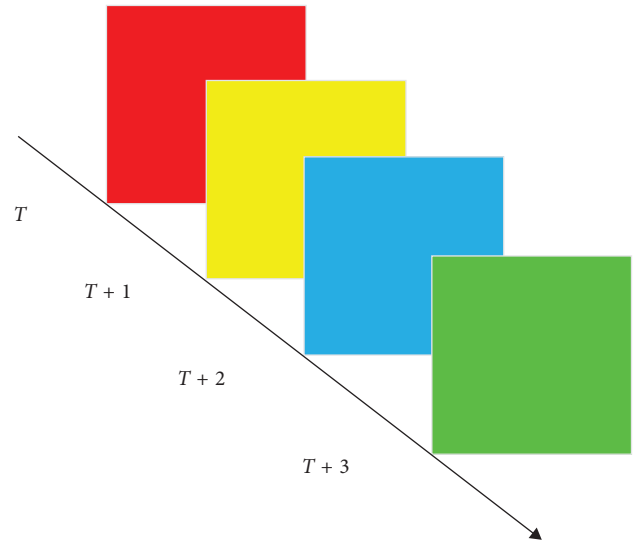

Figure 2: Sketch of tensor stream.

TA-FBMC sharply saves the time cost and the equation is formalized as

$$P\left(c_{(l_{t+1})} \middle| C_{(l_t)}\right) = \frac{1}{C_{(l_t)}} \sum_{c_{(l_t)} \in C_{l_t}} P\left(c_{(l_{t+1})} \middle| c_{(l_t)}\right). \tag{4}$$

The soundness was proved in [5].

$P(c_{(l_{t+1})} | c_{(l_t)})$ represents the probability of all users leaving current category $c_{(l_t)}$ and entering next category $c_{(l_{t+1})}$ at T_b. $L = [l_1, l_2, \ldots, l_n]$ denotes the set of locations, and C denotes the set of location categories, where $C_{(l_t)}$ represents the set of categories l_t might belong to. $c_{(l_t)} \in C_{(l_t)}$ means the category of current location l_t, and $c_{(l_{t+1})} \in C_{(l_{t+1})}$ represents the category of current location l_{t+1}. According to real-life schedules, we divided 24 hours of a day into 6 intervals, before dawn ($T_1 = [2, 3, 4, 5]$), forenoon ($T_2 = [6, 7, 8, 9]$), noon ($T_3 = [10, 11, 12, 13]$), afternoon ($T_4 = [14, 15, 16, 17]$), evening ($T_5 = [18, 19, 20, 21]$), and night ($T_4 = [22, 23, 24, 1]$).

A tensor with 4-way is utilized to construct the category data, including current categories, next categories, months, and time periods. The whole structure accordingly is converted to $\mathbf{B}_t \in \mathscr{R}^{\text{cc} \times \text{nc} \times t \times m}$, where cc and nc represent the current categories and next categories, t denotes the day intervals, and m is the number of historical months. The structure is further elaborated in Figure 4.

Furthermore, we elaborate the dynamic tensor as follows. Existing category dataset with missing data constitutes $\mathbf{B}_t^{\overline{\mathbf{T}}_t}$, which is used for predicting $\mathbf{B}_t^{\mathbf{T}_t}$. The transformation can be expressed as

$$\mathbf{B}_t^{\mathbf{T}_t} = f\left(\mathbf{B}_t^{\overline{\mathbf{T}}_t}\right). \tag{5}$$

In the forecasting phase, the dynamic tensor updates itself by refreshing $\mathbf{B}_t^{\overline{\mathbf{T}}_t}$ with the prediction result of $\mathbf{B}_t^{\mathbf{T}_t}$. The new $\mathbf{B}_t^{\overline{\mathbf{T}}_t}$ is defined as $\mathbf{B}_t^{\overline{\mathbf{T}}_{t+1}}$, and the size of $\mathbf{B}_t^{\overline{\mathbf{T}}_{t+1}}$ is fixed in $(\text{cc}, \text{nc}, t, m)$. In other words, prediction data take the place of the primary data.

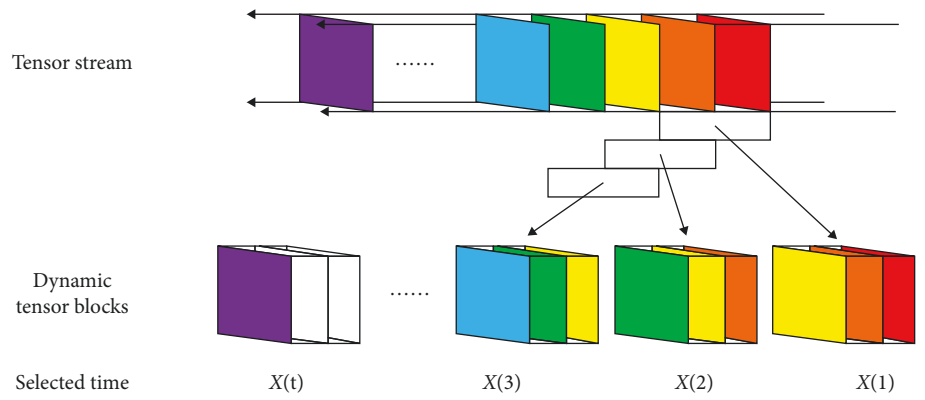

Figure 3: Sketch of dynamic tensor blocks.

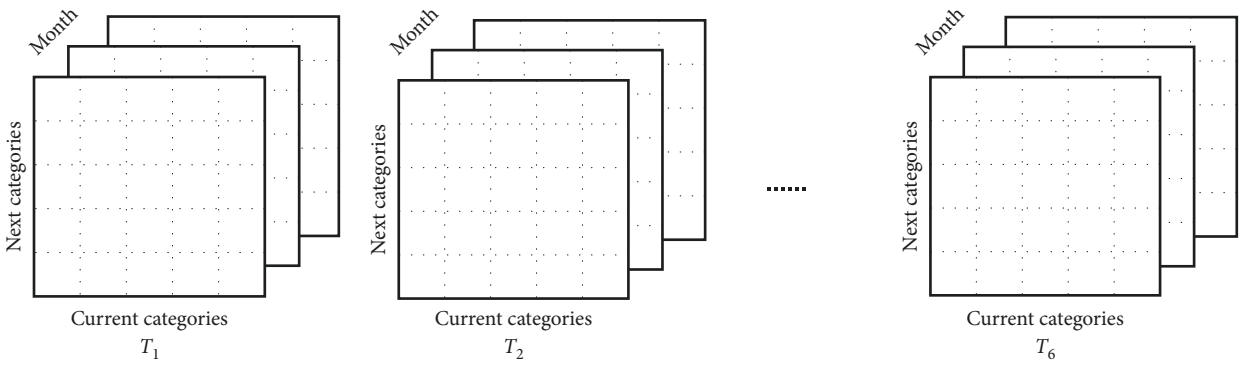

Figure 4: Sketch of 4-dimensions tensor.

After predicting the missing data, the current category data integrity will be improved; the number and the length of utilized data will also get enhanced.

In a nutshell, the overall algorithm concerning the structure of dynamic tensor completion is encapsulated in Algorithm 1.

Firstly, we input the existing sets $\overline{\mathbf{T}}_t$ and target time t^* into the model as the initial data and expected targets. Dynamic tensor $\mathbf{B}_t$, prediction sets $\mathbf{T}_t$, and $a_t = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ are combined to ensure the processing flow. Then, with a_0 initialized as $(1/n, 1/n, \ldots, 1/n)$, the procedure of Prido starts. According to existing data $\mathbf{B}_t^{\overline{\mathbf{T}}_t}$ and former parameters a_{t-1}, PIO can determine the current optimal parameter a_t. Taking a_t as input, FALRTC completes $\mathbf{B}_t^{\overline{\mathbf{T}}_t}$ and outputs current result $\mathbf{B}_t^{\mathbf{T}_t}$. In accordance with the nature of dynamic tensor, we refresh the original data $\mathbf{B}_t^{\overline{\mathbf{T}}_t}$ with the predicted result $\mathbf{B}_t^{\mathbf{T}_t}$ as existing data $\overline{\mathbf{T}}_{t+1}$ for the next round. Then, time t proceeds to $t+1$. Prido will repeat the operation until reaching the target time t^*. Finally, the prediction in t^*, $\mathbf{T}_{t^*}$, can be obtained.

Thus far, we have not explained the intrinsic characteristics of PIO and FALRTC, which will be covered in the following subsections.

3.2. Fast Incremental Tensor Completion. A tensor completion algorithm to realize fast dynamic tensor completion is proposed so as to cope with POI recommendation task.

3.2.1. Optimization Formulation. Current heuristic methods such as Tucker decomposition [4] and CP decomposition [30] aim at transforming tensor into other kinds of data structures. Especially, in CP decomposition, tensor $\mathbf{A} \in \mathscr{R}^{n_1 \times n_2 \times \ldots n_d}$ is represented by a larger r as the linear combination of r tensors (vectors) with rank-1:

$$\begin{aligned} \mathbf{A} &= \sum_{i=1}^{r} \lambda_i \alpha_i^1 \otimes \alpha_i^2 \otimes \ldots \alpha_i^d, \\ \min_{\mathbf{X}, \alpha_1 \ldots \alpha_n} &: \quad \lambda_i \left\| \mathbf{X} - \alpha_i^1 \otimes \ldots \alpha_i^n \right\|_F^2, \\ s.t. &\quad \mathbf{X}_\Omega = \mathbf{D}_\Omega. \end{aligned} \tag{6}$$

As far as Tucker decomposition is concerned, tensor $\mathbf{A} \in \mathscr{R}^{n_1 \times n_2 \times \ldots n_d}$ is decomposed into matrices $U_{(m)} \in \mathscr{R}^{I_m \times J_m} (1 \le m \le d)$ and one small core tensor $\mathbf{G} \in \mathscr{R}^{J_1 \times J_2 \times \ldots \times J_d}$:

$$\begin{aligned} \mathbf{A} &= \mathbf{G} \times_1 U_{(1)} \times_2 U_{(2)} \times \ldots \times_d U_{(d)}, \\ \min_{\mathbf{X}, \mathbf{G}, U_{(1)} \ldots U_{(n)}} &: \quad \frac{1}{2} \left\| \mathbf{X} - \mathbf{G} \times_1 U_{(1)} \times \ldots \times_n U_{(n)} \right\|_F^2, \\ s.t. &\quad \mathbf{X}_\Omega = \mathbf{D}_\Omega. \end{aligned} \tag{7}$$

These methods require data structure transformation. But as this process goes, the error gradually accumulates because of the accompanied inevitable distortion of the original dataset.

Unlike traditional methods, we focus on dynamic tensor completion, in which low computational costs, fast convergence, and high accuracy are required. Consequently, the fast low-rank tensor completion algorithm, namely, FALRTC [31], is harnessed, which proves to be more efficient compared with other approaches.

3.2.2. Algorithmic Solution. In order to improve convergence speed and tackle the tensor trance norm minimization problem, FALRTC is put forward.

With regard to POI recommendation, $\mathbf{D}_t \in \mathscr{R}^{J_1 \times J_2 \times J_3 \times J_4}$ for current category, next category, and seasons and interval modes is considered as the basic element for computing. The specific task is to work out the optimization problem, presented as follows:

$$\begin{aligned} \min_{X_{(t)}} : \quad & f(\mathbf{X}) \coloneqq \sum_{i=1}^{4} \alpha_i \left\| X_{(i)} \right\|_*, \\ s.t. \quad & \mathbf{X}_\Omega = \mathbf{D}_\Omega, \end{aligned} \tag{8}$$

where α_i is the constant satisfying $\alpha_i \geq 0$ and $\sum_{i=1}^{n} \alpha_i = 1$. The difficulty of efficiently solving the optimization problem is mainly caused by the nonsmooth terms in the equation.

The convergence rate can be reduced to $O(K^{-1/2})$ by substituting gradient information with subgradient information, and K refers to the number of iterations [32]. However, this value for minimizing general functions is $O(K^{-2})$ [32]. In order to solve a nonsmooth optimization problem [33], FALRTC aims to (1) transform the original check-ins data into smooth data and (2) tackle the smooth problem, and the solution is utilized to cope with the original problem. Liu et al. [31] provides detailed introduction.

3.3. Parameter Optimization Procedure. The optimization algorithm applied to each dynamic completion parameter optimization is developed in this section.

We mainly utilize the pigeon-inspired optimization (PIO) to optimize the parameters of the dynamic tensor completion method and pinpoint the patterns of each step. As introduced in [34], PIO, which is a population-based swarm intelligence algorithm, imitates pigeons' navigation homing behavior. In the algorithm, Duan and Qiao adopted two operators to describe two stages of the homing phenomenon.

3.3.1. Map and Compass Operators. In the first stage, pigeons can briefly picture the topographic map in the head by means of magnetic sense. They take the height of the sun as compass to modify flight path. After approaching the destination, the dependence on the sun decreases.

3.3.2. Landmark Operator. In the second stage, when approaching the destination, the pigeons take more attention on the landmark. When spotting the familiar building, they will fly straightly to the goal. Otherwise, they will follow leaders that are familiar with the landmark.

Similarly, in the beginning, PIO sets initial location $X_i = [x_{i1}, x_{i2}, \ldots x_{in}]$ and velocity $V_i = [v_{i1}, v_{i2}, \ldots v_{in}]$ for these pigeons. Then, the new location and velocity of each pigeon is updated accordingly as follows:

$$\begin{aligned} V_i^t &= V_i^{t-1} e^{-R \times t} + \text{rand}\left(X_{\text{gbest}} - X_i^{t-1}\right), \\ X_i^t &= X_i^{t-1} + V_i^t, \end{aligned} \tag{9}$$

where $R \in [0, 1]$ denotes the map compass operator, rand represents the random number values in $[0, 1]$, t denotes the current iterations, and X_{gbest} is the global optimum in $t-1$ iterations. The first stage operation will repeat until T_1 iterations.

In the second stage, with landmark operator utilized, pigeons compare the operator with destination. If marching well, the pigeons fly straightly to the goal. After each iteration, half of the pigeons which are far away from the destination might be weeded out. X_{center}, which is the central location of the remainder, will be set as the new landmark. The second stage operation will repeat until T_2 iterations. The combined system is defined as follows:

$$\begin{aligned} X_{\text{center}}^{t-1} &= \frac{\sum X_i^{t-1} \cdot F\left(X_i^{t-1}\right)}{N_p^{t-1} \sum F\left(X_i^{t-1}\right)}, \\ N_p^t &= \frac{N_p^{t-1}}{2}, \\ X_i &= X_i^{t-1} + rand\left(X_{\text{center}}^{t-1} - X_i^{t-1}\right), \end{aligned} \tag{10}$$

where $F()$ is the quality of the pigeon individual and defined as follows:

$$F\left(X_i^{t-1}\right) = \begin{cases} \dfrac{1}{\text{fitness}\left(X_i^{t-1}\right) + \varepsilon}, & \text{for maximization,} \\ \text{fitness}\left(X_i^{t-1}\right), & \text{for minimization,} \end{cases} \tag{11}$$

The whole two-stage operation is depicted in Algorithm 2.

With PIO applied to optimize the parameters of each step in dynamic tensor, each block owns more explanatory and typical power for the specific sampling month interval. For example, when the model is utilized to recommend the July location category, the parameters trained by PIO in the block from Feb to July evidently outperform the one from Jan to Jun.

4. Experiments and Results

In this section, the experimental results are reported, followed by in-depth analysis.

4.1. Experiment Settings. We utilized Foursquare, https://foursquare.com, the most widely used public check-ins datasets, for POI recommendation.

Input: dynamic tensor $\mathbf{B}_t$, existing sets $\overline{\mathbf{T}}_t$, prediction sets $\mathbf{T}_t$, target time t^*, parameter $a_t = (\alpha_1, \alpha_2, \ldots, \alpha_n)$;
Output: prediction $\mathbf{T}_{t^*}$.

```
initialize a_0 = (1/n, 1/n, ..., 1/n)
while t ≠ t* do
    a_t = PIO(B_t^{T̄_t}, a_{t-1})
    B_t^{T_t} = FALRTC(B_t^{T̄_t}, a_t)
    T̄_{t+1} = refresh(T_t, T̄_t)
    t = t + 1
return T_{t*};
```

Algorithm 1: Prido.

Input: pigeon location $X_i = [x_{i1}, x_{i2}, \ldots x_{in}]$, velocity $V_i = [v_{i1}, v_{i2}, \ldots v_{in}]$, map compass operator R, max iterations T_1, T_2, landmark N_p^t, existing data $\overline{\mathbf{T}}_t$, quality function $F()$;
Output: optimal location a_t^*;

```
initialize X_i, V_i, X_gbest, N_p^t
while reach T_1 do
    Calculate the velocity of each pigeon
    Update the location of each pigeon
    t = t + 1
while reach T_2 do
    Calculate landmark X_center^t
    Update N_p^t
    Calculate the location
    t = t + 1
a_t^* = X_t(x_t1, x_t2, ..., x_tn)
return a_t^*;
```

Algorithm 2: The procedure of PIO.

Table 1: The data statistics.

City	User	Location	Category	Tip
New York	2,581	206,416	249	166,530
Los Angeles	1,604	215,614	249	109,526

4.1.1. Data Sources. We utilized the check-ins data in New York and Los Angeles from January 2010 to June 2011, which contain the information of users, locations, categories, and tips. The statistics of the data are depicted in Table 1.

4.1.2. Evaluation Indices. Our target is to recommend a suitable category to the user in need, and a list of Top-N recommended categories is provided. Once the user selects at least one item in the suggested list, the recommendation will be reckoned as successful. Specifically, if our recommended categories intersect with users' real Top-N lists, the prediction is deemed to be correct:

$$P@N = \frac{\text{the counts of correct predictions}}{\text{the total number of recommendation rounds}}. \quad (12)$$

4.2. Experiment Results. We experimentally compare our tensor completion-based method with matrix factorization models MF, PMF, and FBMC and tensor factorization models CD, TD TA-FBMC, and TAD-FMPC, and the details are shown in Table 2. Furthermore, to validate the merit of the structure based on dynamic tensor, the original completion method, static-Prido, is applied in the experiment as well. In short, eight methods are assessed in our work.

All experiments were implemented in MATLAB 2013a, and all tests were performed on a PC with Intel Core 2 2.67 GHz and 4 GB RAM. Tables 3 and 4 provide the overall results.

As Tables 3 and 4 and Figure 5 show, Prido outperforms other methods, except TD in $P@100$. For the similar distribution of the result in two cities, we take New York City into analysis.

Overall speaking, the various traditional matrix factorization models perform relatively worse compared with tensor factorization models. In terms of accuracy, even the best method utilizing matrix factorization (FBMC) achieves half the value of the worst one harnessing tensor factorization (TA-FBMC). The reason can be attributed to the fact that the matrix is 2-dimension tensor in essence, which means at least two dimension information cannot

Table 2: Models for comparison.

Model	Scale	Description
Matrix factorization (MF)	$\|U\| \times \|C\|$	MF is widely used in CF and usually set as baseline.
Probabilistic matrix factorization (PMF)	$\|U\| \times \|C\| \times \|C\|$	PMF is a conventional model in recommendation domain.
Factorized personalized Markov chain (FBMC)	$\|U\| \times \|C\| \times \|C\|$	FBMC formalizes the user's preference as a personalized Markov chain.
Tucker decomposition (TD)	$\|U\| \times \|T\| \times \|C\| \times \|C\|$	TD transforms the high-dimension tensor into a core tensor with a relative matrix in each dimension.
Canonical polyadic decomposition (CD)	$\|U\| \times \|T\| \times \|C\| \times \|C\|$	CD transforms the high-dimension tensor into a multiple equation of linear complexity.
Time-aware FBMC (TA-FBMC)	$\|U\| \times \|T\| \times \|C\| \times \|C\|$	TA-FBMC equips the time factor with the FBMC.
Time-aware decay FBMC (TAD-FMPC)	$\|U\| \times \|T\| \times \|C\| \times \|C\|$	TAD-FMPC adds decay of the probability over time in TA-FBMC.
Static prido (s-Prido)	$\|U\| \times \|T\| \times \|C\| \times \|C\|$	s-Prido removes the dynamic tensor structure from Prido.

Table 3: Result comparison in New York.

	Matrix factorization			Tensor factorization					
Metrics	MF	PMF	FBMC	CD	TD	TA-FBMC	TAD-FMPC	s-Prido	Prido
P@1	0.0016	0.0060	0.0310	0.0767	0.0921	0.0747	0.1230	0.1022	0.1310
P@5	0.0197	0.0283	0.1063	0.2221	0.2642	0.2298	0.2996	0.2454	0.3201
P@10	0.0444	0.0571	0.1700	0.3249	0.3863	0.3397	0.4136	0.3768	0.4234
P@20	0.0822	0.1160	0.2699	0.4829	0.5357	0.4801	0.5615	0.5387	0.5911
P@50	0.2744	0.2843	0.4893	0.7195	0.7552	0.7130	0.7699	0.7488	0.7824
P@100	0.5053	0.5127	0.7280	0.8887	0.9040	0.8812	0.8965	0.8876	0.9010

Table 4: Result comparison in Los Angeles.

	Matrix factorization			Tensor factorization					
Metrics	MF	PMF	FBMC	CD	TD	TA-FBMC	TAD-FMPC	s-Prido	Prido
P@1	0.0433	0.0057	0.0477	0.0677	0.0964	0.0928	0.1519	0.1252	0.1672
P@5	0.1142	0.0336	0.1351	0.2270	0.2695	0.2580	0.3250	0.2742	0.3425
P@10	0.1734	0.0666	0.1992	0.3216	0.3957	0.3610	0.4382	0.3923	0.4594
P@20	0.2863	0.1305	0.3023	0.4920	0.5477	0.4974	0.5756	0.5523	0.6092
P@50	0.4486	0.2949	0.5088	0.7242	0.7588	0.7262	0.7753	0.7598	0.7973
P@100	0.5834	0.5389	0.7412	0.8920	0.9027	0.8839	0.8971	0.8856	0.8993

be involved in the structure, and no matter which factorization methods is utilized, missing dimensions inevitably will restrict the latent accuracy. Thus, the further development of POI recommendation has to rely on tensor factorization.

As for the comparisons among tensor factorization-based methods, CD and TD, as the earliest solutions, their practical arithmetical operations cost much more running time and they also perform badly in the fine-grained prediction task. Although TD obtains the best performance in P@100 recommendation with the precision at 90.4%, the expense of time reaches almost 2 days, whereas the last four methods solely cost at most 3 hours. In terms of balancing running time with prediction precision, these two conventional methods are not the ideal choices.

Moreover, s-Prido outperforms TA-FBMC by approximately 3%, which reveals that our 4-dimensions tensor structure is effective. Despite the fact that the user factor matters in personalized recommendation, people prefer to do popular things and consequently, the introduction of users' check-ins data is more reasonable than separate analysis. With the month dimension added, the influence of different seasons can be expressed in the model. Nevertheless, TAD-FMPC defeats s-Prido across all metrics since static structure cannot expose the weight of diverse time, in which the premier data have the same influence as recent data for recommendation. To tackle this issue, TAD-FMPC adopts three different types of decay of the probability over the time factor. Nevertheless, this method neglects the variety of different seasons. Due to the adjacency of seasons, the user's behavior in autumn is similar to summer, whereas it has little correlation with spring. Therefore, the check-ins data in different seasons should not be simply reflected in the same equation. Based on the consideration, we propose Prido, in which adjacent seasons data are put into one dynamic tensor block. As can be seen from the table, Prido improves the result by 2% when compared with TAD-FMPC.

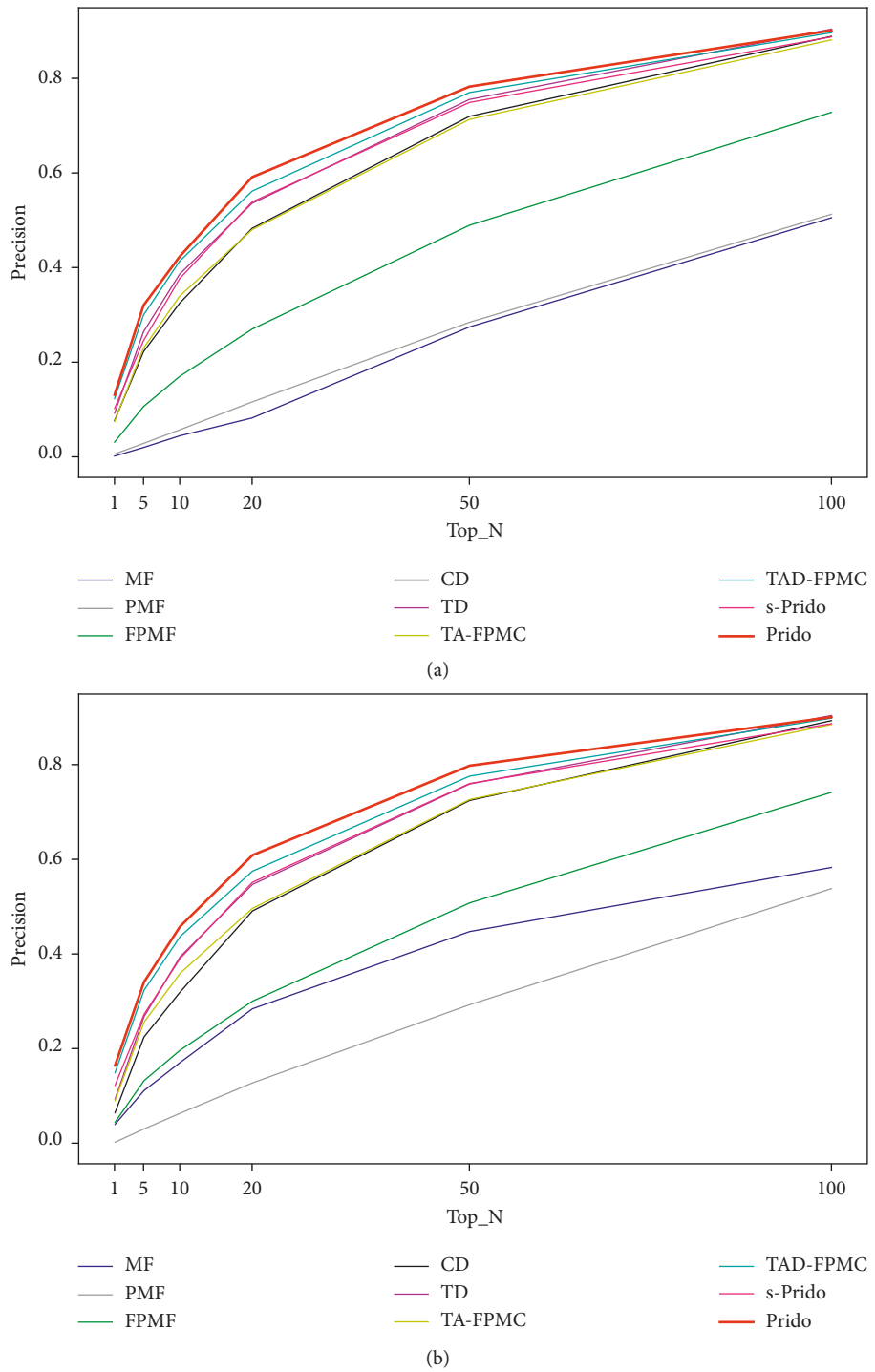

Figure 5: Sketch of category prediction. (a) New York. (b) Los Angeles.

5. Conclusion

In this paper, a novel dynamic heuristic tensor completion approach on the basis of fast low-rank tensor completion algorithm is proposed to solve the POI recommendation task. Fast low-rank tensor completion (FALRTC) is leveraged as a supplement to the original dynamic tensor structure, so as to improve the performance prediction. Prido is able to capture the inner features of check-ins data due to the multimode characteristics such as current categories, next categories, and temporal information as well as seasons variations are all integrated in the model. Experiment results not only validate the superiority of our proposed method but also suggest that there is potential application opportunity as for POI recommendation environment in a large and dynamic scope.

Data Availability

The data used to support this study are available at https://foursquare.com/, which can be publicly accessed without any charge.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was partially supported by NSFC under Grant nos. 61872446, 71690233, and 71331008.

References

[1] Y. Hu, Y. Koren, and C. Volinsky, "Collaborative filtering for implicit feedback datasets," in *Proceedings of 8th IEEE International Conference on Data Mining (ICDM 2008)*, pp. 263–272, Pisa, Italy, December 2008.

[2] R. Pan, Y. Zhou, B. Cao et al., "One-class collaborative filtering," in *Proceedings of 8th IEEE International Conference on Data Mining (ICDM 2008)*, pp. 502–511, Pisa, Italy, December 2008.

[3] A. Cichocki, R. Zdunek, A. H. Phan, and S. Amari, *Nonnegative Matrix and Tensor Factorizations-Applications to Exploratory Multi-Way Data Analysis and Blind Source Separation*, Wiley, Hoboken, NJ, USA, 2009.

[4] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[5] X. Li, M. Jiang, H. Hong, and L. Liao, "A time-aware personalized point-of-interest recommendation via high-order tensor factorization," *ACM Transactions on Information Systems*, vol. 35, no. 4, pp. 1–23, 2017.

[6] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme, "Factorizing personalized markov chains for next-basket recommendation," in *Proceedings of 19th International Conference on World Wide Web, WWW 2010*, pp. 811–820, Raleigh, NC, USA, April 2010.

[7] S. Zhao, T. Zhao, H. Yang, M. R. Lyu, and I. King, "STELLAR: spatial-temporal latent ranking for successive point-of-interest recommendation," in *Proceedings of Thirtieth AAAI Conference on Artificial Intelligence*, pp. 315–322, Phoenix, AZ, USA, February 2016.

[8] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proceedings of Tenth International World Wide Web Conference, WWW 10*, pp. 285–295, Hong Kong, China, May 2001.

[9] J. L. Herlocker, J. A. Konstan, A. Borchers, and J. Riedl, "An algorithmic framework for performing collaborative filtering," in *Proceedings of 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 230–237, Berkeley, CA, USA, August 1999.

[10] C. Desrosiers and G. Karypis, *A comprehensive Survey of Neighborhood-Based Recommendation Methods, in Recommender Systems Handbook*, Springer, Berlin, Germany, 2011.

[11] C. Cheng, H. Yang, I. King, and M. R. Lyu, "Fused matrix factorization with geographical and social influence in location-based social networks," in *Proceedings of Twenty-Sixth AAAI Conference on Artificial Intelligence*, Toronto, Ontario, Canada, July 2012.

[12] H. Gao, J. Tang, and H. Liu, "gSCorr: modeling geo-social correlations for new check-ins on location-based social networks," in *Proceedings of 21st ACM International Conference on Information and Knowledge Management, CIKM'12*, pp. 1582–1586, Maui, HI, USA, October–November 2012.

[13] M. Park, J. Hong, and S. Cho, "Location-based recommendation system using bayesian user's preference model in mobile devices," in *Proceedings Ubiquitous Intelligence and Computing, 4th International Conference, UIC 2007*, pp. 1130–1139, Hong Kong, China, July 2007.

[14] M. Ye, P. Yin, W. Lee, and D. L. Lee, "Exploiting geographical influence for collaborative point-of-interest recommendation," in *Proceeding of 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011*, pp. 325–334, Beijing, China, July 2011.

[15] J. Zhang and C. Chow, "Ticrec: a probabilistic framework to utilize temporal influence correlations for time-aware location recommendations," *IEEE Trans. Services Computing*, vol. 9, no. 4, pp. 633–646, 2016.

[16] J. D. Zhang and C. Y. Chow, "Spatiotemporal sequential influence modeling for location recommendations: a gravity-based approach," *ACM Transactions on Intelligent Systems Technology*, vol. 7, no. 1, pp. 1–25, 2015.

[17] Q. Yuan, G. Cong, Z. Ma, A. Sun, and N. Magnenat-Thalmann, "Time-aware point-of-interest recommendation," in *Proceedings of 36th International ACM SIGIR conference on research and development in Information Retrieval, SIGIR'13*, pp. 363–372, Dublin, Ireland, July–August 2013.

[18] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: user movement in location-based social networks," in *Proceedings of 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1082–1090, San Diego, CA, USA, August 2011.

[19] J. D. Zhang and C. Y. Chow, "Point-of-interest recommendations in location-based social networks," *SIGSPATIAL Special*, vol. 7, no. 3, pp. 26–33, 2015.

[20] J. D. Zhang, C. Y. Chow, and J. Xu, "Enabling kernel-based attribute-aware matrix factorization for rating prediction," *IEEE Transactions on Knowledge Data Engineering*, vol. 29, no. 4, pp. 798–812, 2017.

[21] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of Interspeech 2010, 11th Annual Conference of the International Speech Communication Association*, pp. 1045–1048, Makuhari, Chiba, Japan, September 2010.

[22] Y. Chen, J. Zhao, X. Hu, X. Zhang, Z. Li, and T. Chua, "From interest to function: location estimation in social media," in *Proceedings of Twenty-Seventh AAAI Conference on Artificial Intelligence*, Bellevue, Washington, USA, July 2013.

[23] Q. Liu, S. Wu, L. Wang, and T. Tan, "Predicting the next location: a recurrent model with spatial and temporal contexts," in *Proceedings of Thirtieth AAAI Conference on Artificial Intelligence*, pp. 194–200, Phoenix, AZ, USA, February 2016.

[24] Z. Zhang, C. Li, Z. Wu, A. Sun, D. Ye, and X. Luo, "NEXT: a neural network framework for next POI recommendation," *CoRR*, 2017, http://arxiv.org/abs/1704.04576.

[25] X. Zhao, C. Xiao, X. Lin, W. Zhang, and Y. Wang, "Efficient structure similarity searches: a partition-based approach," *The VLDB Journal*, vol. 27, no. 1, pp. 53–78, 2018.

[26] J. Liao, J. Tang, W. Zeng, and X. Zhao, "Efficient and accurate traffic flow prediction via incremental tensor completion," *IEEE Access*, vol. 6, pp. 36897–36905, 2018.

[27] I. V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.
[28] O. R. A. Kazeev and C. Schwab, "Low-rank tensor structure of linear diffusion operators in the tt and qtt formats," *Linear Algebra and its Applications*, vol. 438, no. 11, pp. 4204–4221, 2013.
[29] C. Cheng, H. Yang, M. R. Lyu, and I. King, "Where you like to go next: successive point-of-interest recommendation," in *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, pp. 2605–2611, Beijing, China, August 2013.
[30] T. G. K. E. Acar, D. M. Dunlavy, and M. Mørup, "Scalable tensor factorizations for incomplete data," *Chemometrics and Intelligent Laboratory Systems*, vol. 106, no. 1, pp. 41–56, 2011.
[31] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208–220, 2013.
[32] Y. Nesterov, *Introductory Lectures on Convex Programming, Lecture Notes*, Springer, Berlin, Germany, 1998.
[33] Y. Nesterov, "Smooth minimization of non-smooth functions," *Mathemtaical Programming*, vol. 103, no. 1, pp. 127–152, 2005.
[34] H. Duan and P. Qiao, "Pigeon-inspired optimization: a new swarm intelligence optimizer for air robot path planning," *International Journal of Intelligent Computing and Cybernetics*, vol. 7, no. 1, pp. 24–37, 2014.