Long-Short Term Spatiotemporal Tensor Prediction for Passenger Flow Profile

Ziyue Li, Hao Yan, Chen Zhang, and Fugee Tsung

***Abstract*—Spatiotemporal data are very common in many applications, such as manufacturing systems and transportation systems. Given the intrinsic complex spatial and temporal correlations of such data, short-term and long-term prediction for spatiotemporal data is often very challenging. Most of the traditional statistical models fail to preserve innate features in data alongside their complex correlations. In this paper, we focus on a tensor-based prediction method and propose several practical techniques to improve both long-term and short-term prediction accuracy. For long-term prediction, we propose the "tensor decomposition + 2-Dimensional Auto-Regressive Moving Average (2D-ARMA)" model, and an effective way to update prediction in real-time; For short-term prediction, we propose to conduct tensor completion based on tensor clustering to avoid oversimplification and ensure accuracy. A case study based on the metro passenger flow data is conducted to demonstrate the improved performance.**

***Index Terms*—Intelligent transportation system, probability and statistical methods, big data in robotics and automation.**

I. Introduction

PASSENGER flow data of an Urban Rapid Transit (URT) system are characterized as typical spatiotemporal data. Predicting passenger flow of URT systems has significant commercial value, such as automatic warning in advance when the system failure occurs. Based on the length of the prediction horizon, the task can be classified as short-term prediction (for several hours) and long-term prediction (for several days). Currently, the complicated spatial and temporal correlation structure hinders accurate prediction and further analysis [1].

The goal of spatiotemporal analysis is to capture various implicit spatial and temporal correlations. In our URT case, for spatial correlation, the first aspect is the Law of Geography, where the passenger flow of a station is usually affected by its spatially adjacent neighbors. Fig. 1(a) plots the correlation structure of passenger flow data in stations along a specific URT line on a specific day. We can observe that neighboring stations are strongly correlated. The second aspect is the contextual similarity, where two stations sharing similar functionalities (business center, residential area or school, etc.) are more likely to have similar passenger flows. For temporal correlation, the future prediction is often correlated with historical observations in two different temporal scales such as weekly correlation and daily correlation as shown in Fig. 1(b, c). Denote $\mathbf{y}_{(d,w)}$ as the inflow profile of day d on week w, by daily correlation we assume the profile $\mathbf{y}_{(d,w)}$ is correlated with previous days of the same week: $\mathbf{y}_{(d,w)} \sim f(\mathbf{y}_{(d-1,w)}, \mathbf{y}_{(d-2,w)})$; By weekly correlation the profile is believed to be correlated with the same day of previous weeks: $\mathbf{y}_{(d,w)} \sim f(\mathbf{y}_{(d,w-1)}, \mathbf{y}_{(d,w-2)})$.

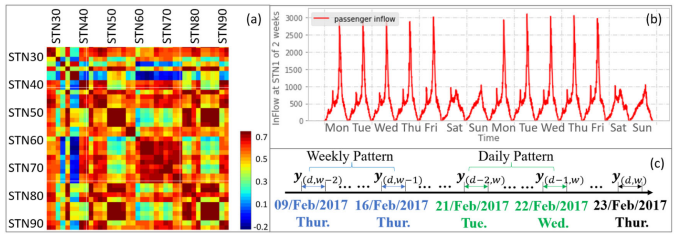

Fig. 1. (a) Spatial Correlation Matrix of Passenger One-Day Inflow Profile of Selected Stations in Hong Kong. (b) Passenger Inflow Profile of 2 Weeks for STN1 (Station names are desensitized as station code 'STN#'). (c) Illustration of Daily (in Green) and Weekly (in Blue) Temporal Correlation Observed in Data. Daily correlation: The passenger profile of today is related to the pattern yesterday. Weekly correlation: The passenger profile of today is correlated with the same day of previous weeks.

Existing spatiotemporal forecasting methods are mainly based on geostatistical models with regularization techniques. Different locations sharing similar features display common spatial correlations: Zhao *et al.* used $l_{2,1}$-norm to achieve the relatedness of locations sharing the same keyword pool in social media analysis [2]; Zhang *et al.* also utilized $l_{2,1}$-norm to encourage all locations to select common features in citywide passenger flow prediction [3].

To capture the temporal pattern, many traditional time-series models have been developed for traffic prediction, such as Holt-Winters forecasting [4] and Auto-Regressive Integrated Moving Average (ARIMA) [5]. To consider temporal smoothness, it is often enforced by penalizing the difference between two consecutive timestamps $\|\boldsymbol{w}_t - \boldsymbol{w}_{t-1}\|$ [6]. However, the linear

Manuscript received March 16, 2020; accepted June 4, 2020. Date of publication June 25, 2020; date of current version July 3, 2020. This letter was recommended for publication by Associate Editor Ettore Lanzarone and Editor Jingang Yi upon evaluation of the reviewers' comments. This work was supported in part by Hong Kong RGC GRF under Grants 16203917 and 16201718, in part by NSFC under Grants 71931006, 71901131, and 71932006, in part by NSF CMMI under Grant 1922739, and in part by NSF under Grant DMS-1830363 and ITT/007/19GP. *(H. Yan and C. Zhang contributed equally to this work.) (Corresponding author: Ziyue Li.)*

Ziyue Li is with the Department of Industrial Engineering and Decision Analytics, Hong Kong University of Science and Technology, Hong Kong (e-mail: zlibn@connect.ust.hk).

Hao Yan is with the Assistant Professor in School of Computing, Informatics, and Decision Systems Engineering, Arizona State University, Tempe, AZ 85281 USA (e-mail: haoyan@asu.edu).

Chen Zhang is with the Assistant Professor in Industrial Engineering, Tsinghua University, Beijing 100084, China (e-mail: zhangchen01@tsinghua.edu.cn).

Fugee Tsung is with the Chair Professor in Department of Industrial Engineering and Decision Analytics, Hong Kong University of Science and Technology, Hong Kong (e-mail: season@ust.hk).

Digital Object Identifier 10.1109/LRA.2020.3004785

parametric form with few lags imposes strong assumptions on the temporal correlation, which may lead to under-fitting. Most importantly, a traditional ARIMA is powerless to present the cross-correlation of the spatial and temporal dimension, which is severely overlooked in most of the above research.

Another way to deal with the spatiotemporal prediction is the application of neural networks. Convolutional Neural Network (CNN) is a common way to capture spatial correlation by using a station's neighbors to predict its future behavior. Recurrent Neural Network (RNN) and its variants, e.g., Long Short Term Memory networks (LSTM), are sufficiently capable of modeling complex temporal correlation. Built upon CNN and LSTM, H. Yao *et al.* [7] developed a spatiotemporal network to predict taxi demand of different regions. It adds a further layer to consider the semantic similarity of locations, meaning that locations sharing a similar functionality may have similar demand patterns. Recently, X. Geng *et al.* [8] incorporated a multi-graph convolution network to interpret not only the Euclidean correlation among spatially adjacent regions but also involved non-Euclidean adjacency, including function similarity and connectivity, following by Gated Recurrent Neural Network (GRNN) to capture temporal correlation. However, it not only requires costly computational resources to train these deep learning models but also demands a very large sample size for training.

In this research, we aim to develop a computationally efficient and robust methodology to deal with the spatiotemporal prediction problem. To achieve this, we propose to use the tensor representation of the spatiotemporal data, which is proven as an efficient way to represent spatiotemporal data due to its sufficient capacity to preserve inter-correlations along multiple dimensions. In particular, there are two conventional ways to conduct spatiotemporal prediction by tensor, and we categorize them as 2-step tensor prediction and 1-step tensor prediction in the following.

a) 2-step Tensor Decomposition and Time Series Modeling for Spatiotemporal Prediction: These methods combine tensor decomposition and traditional time-series prediction models [9]–[11] and can be used for long-term prediction.

Tensor decomposition can be considered as a high-dimensional version of matrix singular value decomposition [12]. Two specific forms of tensor decomposition are usually adopted in tensor analysis. CANDECOMP/PARAFAC (CP) decomposes a tensor as a sum of rank-one tensors, and Tucker decomposition decomposes a tensor into a core tensor multiplied by each mode matrix.

Moreover, the 2-step tensor prediction initially conducts tensor decomposition to obtain the temporal mode matrix along time-dimension and then exploits the time-series model on the temporal mode. Holt-Winters forecasting [9], Auto-Regressive Integrated Moving Average (ARIMA), and Support Vector Regression (SVR) [11] have been exploited.

However, existing 2-step tensor prediction methods are not capable of capturing all the weekly and daily patterns mentioned above. Take ARIMA or AR model for example. If the weekly pattern is desired, then the time-lag should be set large to include the past few weeks at least, which highly complicates the model. To address this, we propose to first reshape the daily profile into a matrix (in the form of $\mathbb{R}^{day\times week}$) and then apply a 2-Dimensional Auto-Regressive Moving Average (2D-ARMA) model to capture all those daily and weekly patterns. We name it as "2-step 2D-ARMA" tensor prediction. Finally, real-time prediction update when new data arrive is another challenge. To this end, we also propose a lean dynamic updating method.

b) 1-step Tensor Prediction based on Tensor Completion: This is based on tensor completion [13]–[15], and it is suitable for short-term prediction (prediction horizon as several hours ahead). Tensor completion is originally designated for tensor random missing data imputation. H. Tan *et al.* used tensor completion first time for traffic volume prediction [13], [16], which treated future data as missing data to be estimated.

For tensor completion, Low-Rank Tensor-Completion (LRTC) method has prevailed. One of the most common techniques is adding a nuclear-norm on the rank of tensor [17]–[19]; Q. Shi *et al.* tried l_1-norm on CP weight vector [20]; Q. Zhao *et al.* also proposed a CP-based Bayesian hierarchical probabilistic model, assuming that all mode matrices were generated from higher-level latent distributions, with sparsity-inducing prior to low-rank [21].

However, 1-step tensor prediction based on LRTC is prone to oversimplify the model, which results in the loss of prediction accuracy. To solve this problem, we propose to first cluster spatiotemporal data and then conduct LRTC within data from the same cluster. To this end, tensor clustering method will also be studied.

In this paper, we will focus on improving both 1-step and 2-step state-of-the-art tensor prediction methods, and provide practical and effective techniques to improve the prediction performance correspondingly. In summary, this paper makes the following contributions:

- For long-term prediction, to our best knowledge, we are the first to propose a 2-step tensor prediction model based on 2D-ARMA, capturing the daily and weekly correlations in a better way.
- For 2-step tensor prediction, we propose a novel lean dynamic updating method for tensor decomposition to update the previous prediction in a real-time manner when new data come. This is also ignored by most current research.
- For short-term prediction, we improve the LRTC by conducting it together with a tensor cluster algorithm.
- Furthermore, we applied our model to real passenger flow data in Hong Kong URT system, and it provides quite accurate predictions compared with the existing methods.

II. Preliminaries

In this section, we will review the preliminaries and backgrounds of tensor decomposition and tensor completion methods.

A. Notations and Operations

Throughout this exposition, we denote scalars in italics, e.g., n; vectors by lowercase italic letters in boldface, e.g., $\boldsymbol{u}$; matrices

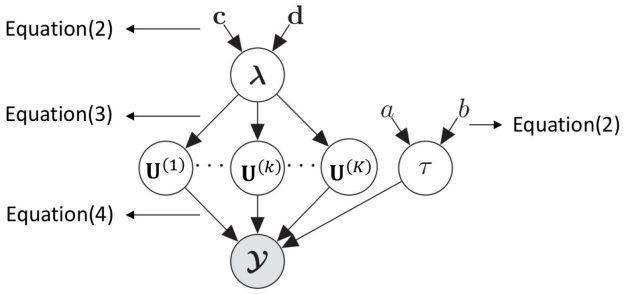

Fig. 2. Bayesian Probabilistic Structure

by uppercase boldface letters, e.g., $\mathbf{U}$; and high dimensional data and tensor by boldface script capital, e.g., $\boldsymbol{\mathcal{X}}$.

B. Tensor Decomposition and Completion

In this subsection, we will introduce the basic knowledge of CP decomposition and tensor completion.

a) CP Decomposition: A tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_K}$ is represented as the weighted summation of a set of rank-one tensors [12]:

$$\begin{aligned} \boldsymbol{\mathcal{X}} &= \sum_{r=1}^{R} \lambda_r \boldsymbol{u}_r{}^{(1)} \circ \boldsymbol{u}_r{}^{(2)} \circ \cdots \circ \boldsymbol{u}_r{}^{(K)} \\ &= [\![\boldsymbol{\lambda}; \mathbf{U}^{(1)}, \mathbf{U}^{(2)}, ..., \mathbf{U}^{(K)}]\!], \end{aligned} \tag{1}$$

where each $\boldsymbol{u}_r{}^{(k)} (k = 1, \ldots, K)$ is a unit vector, and $\circ$ is the outer product, and $\boldsymbol{u}_r{}^{(1)} \circ \boldsymbol{u}_r{}^{(2)} \circ \cdots \circ \boldsymbol{u}_r{}^{(K)}$ is a rank-one tensor. $\mathbf{U}^{(k)} = [\boldsymbol{u}_1{}^{(k)}, \boldsymbol{u}_2{}^{(k)}, \ldots, \boldsymbol{u}_R{}^{(k)}] \in \mathbb{R}^{I_K \times R} (k = 1, \ldots, K)$ is the mode matrix of dimension-k and R is the rank of CP decomposition. $\boldsymbol{\lambda} = [\lambda_1, \ldots, \lambda_R]$ is the score vector.

b) Bayesian Low-rank Tensor Decomposition: Consider $\boldsymbol{\mathcal{Y}}^{I_1 \times I_2 \times \cdots \times I_K}$ is a noisy observation of tensor $\boldsymbol{\mathcal{X}}$, i.e., $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{X}} + \boldsymbol{\epsilon}$, and the noise is assumed to be an i.i.d Gaussian distribution $\boldsymbol{\epsilon} \sim \prod_{i_1, \ldots, i_K} \mathcal{N}(0, \tau^{-1})$. $\boldsymbol{\mathcal{X}}$ is generated by CP model, with weight absorbed inside of mode matrices. Mode-k factor matrix $\mathbf{U}^{(k)}$ can be denoted by row-wise or column-wise vectors $\mathbf{U}^{(k)} = [\boldsymbol{u}_{i_k}^{(k)}, \ldots, \boldsymbol{u}_{i_k}^{(k)}, \ldots, \boldsymbol{u}_{i_k}^{(k)}]^T = [\boldsymbol{u}_1{}^{(k)}, \ldots, \boldsymbol{u}_r{}^{(k)}, \ldots, \boldsymbol{u}_R{}^{(k)}]$.

The generative model based on Bayesian probabilistic structure [21] is shown in Fig. 2, and is specified as:

- $\boldsymbol{\lambda}$ and τ are generated by:

$$\begin{aligned} P(\boldsymbol{\lambda}) &= \prod_{r=1}^{R} Ga(\lambda_r | c_0^r, d_0^r), \\ P(\tau) &= Ga(\tau | a_0, b_0). \end{aligned} \tag{2}$$

  where $Ga(x|a, b)$ denotes a Gamma distribution.
- $\mathbf{U}^{(k)}$ (given $\boldsymbol{\lambda}$) is generated by:

$$P(\mathbf{U}^{(k)} | \boldsymbol{\lambda}) = \prod_{i_k=1}^{I_k} \mathcal{N}(\boldsymbol{u}_{i_k}^{(k)} | 0,\ \Lambda^{-1}), \Lambda = diag(\boldsymbol{\lambda}), \tag{3}$$

  where $\mathcal{N}(\cdot)$ denotes a Gaussian distribution, and $diag(\cdot)$ is a diagonal matrix.
- $\boldsymbol{\mathcal{Y}}_{\boldsymbol{\Omega}}$ (given $\{\mathbf{U}^{(k)}\}_{k=1}^{K}, \tau$) is generated by:

$$P(\boldsymbol{\mathcal{Y}}_{\boldsymbol{\Omega}} | \{\mathbf{U}^{(k)}\}_{k=1}^{K}, \tau) = \prod_{i_1=1}^{I_1} \cdots \prod_{i_k=1}^{I_k} \mathcal{N}\left(\mathcal{Y}_{i_1 i_2 \ldots i_K} | \langle \boldsymbol{u}_{i_1}^{(1)}, \boldsymbol{u}_{i_2}^{(2)}, \cdots, \boldsymbol{u}_{i_K}^{(K)} \rangle, \tau^{-1}\right)^{\boldsymbol{O}_{i_1 i_2 \ldots i_K}}. \tag{4}$$

where $\langle \boldsymbol{u}_{i_1}^{(1)}, \boldsymbol{u}_{i_2}^{(2)}, \ldots, \boldsymbol{u}_{i_K}^{(K)} \rangle$ denotes a generalized inner-product of K vectors.

c) Tensor Completion: Tensor completion is usually designed for missing data imputation, like pixel imputations in image data [21]. The basic tensor completion is formulated as following:

$$\min_{\boldsymbol{\mathcal{Y}}} \ \|\boldsymbol{\mathcal{X}}_{\boldsymbol{\Omega}} - \boldsymbol{\mathcal{Y}}_{\boldsymbol{\Omega}}\| + \alpha \|\boldsymbol{\mathcal{Y}}\|_*, \tag{5}$$

where $\boldsymbol{\mathcal{X}}$ is the incomplete input tensor, $\boldsymbol{\mathcal{Y}}$ is the completed output tensor, $\|\cdot\|_*$ is the nuclear norm, defined as the (weighted) sum of the nuclear norms of the matricizations of $\boldsymbol{\mathcal{Y}}$, to achieve low rank [17], and $\boldsymbol{\Omega}$ is sampling set which denotes the indices of the observed elements of a tensor.

III. Proposed Tensor Prediction Framework

In this section, we aim to propose a spatiotemporal prediction framework based on the tensor decomposition and tensor completion methods in Section II. For the short-term prediction, the effective prediction horizon is 2-hour ahead, which is the response time needed for URT systems to take corresponding actions when abnormal passenger flow patterns happen. In the case of URT passenger flow, data can be represented as: $\boldsymbol{\mathcal{X}}^{L \times T \times P}$. L is location standing for 120 stations in our dataset, T is the time scope we are looking at (here is from 1^{st} Jan 2017 to 28^{th} Feb 2017), P is the 5-minute interval observations per day with 247 sensing points. In this study, the size of our data is $\boldsymbol{\mathcal{X}}^{120 \times 59 \times 247}$.

Our prediction problem can be formulated as:

$$\boldsymbol{\mathcal{X}}^{L \times (T+\tau) \times P} = f(\boldsymbol{\mathcal{X}}^{L \times T \times P}), \tag{6}$$

where τ is the prediction horizon, and $f(\cdot)$ is the prediction model.

A. 2-Step Tensor Long-Term Prediction: Combine Tensor Decomposition and 2D-ARMA

2-step tensor prediction becomes popular in the past few years and the mechanism (shown as Fig. 3(a)) behind is designed as:

- Formulate data in a tensor form. In our case, it is $\boldsymbol{\mathcal{X}}^{L \times T \times P}$. After the tensor decomposition, the temporal mode matrix $\mathbf{U}_T$ can be obtained.
- Use traditional time-series model $f(\cdot)$ to predict incoming τ time's temporal model matrix $\mathbf{U}_{T+\tau} = f(\mathbf{U}_T)$.
- Then, the predicted tensor $\hat{\boldsymbol{\mathcal{X}}}^{L \times (T+\tau) \times P}$ can be computed as: $\hat{\boldsymbol{\mathcal{X}}}^{L \times (T+\tau) \times P} = [\![\boldsymbol{\lambda}; \mathbf{U}_L, \mathbf{U}_{T+\tau}, \mathbf{U}_P]\!]$.

For the time-series model, options include Holt-Winters method, ARIMA, etc. However, as mentioned before, they are unable to capture the temporal pattern, as shown in Fig. 1(c), especially the weekly pattern. This is because, to take the same day of the past two weeks, at least 14 days time-lags should

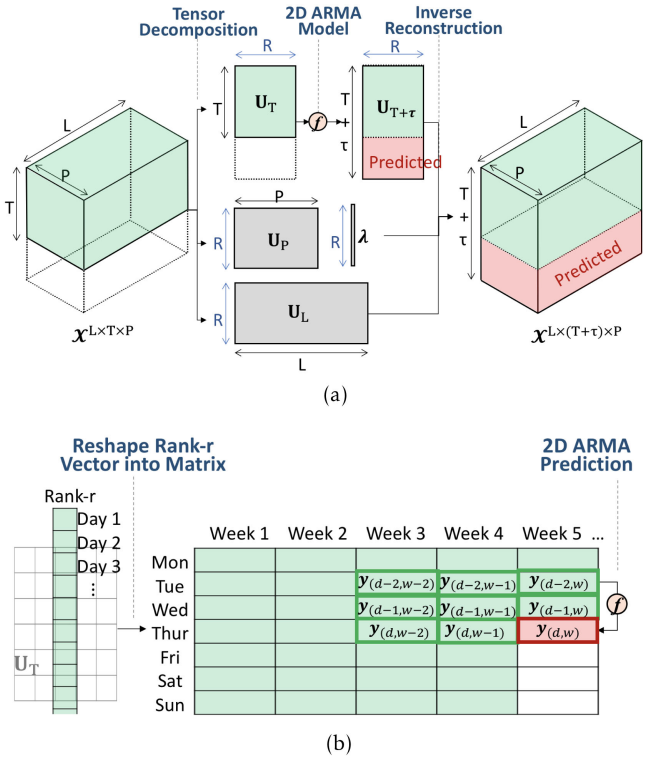

Fig. 3. 2-step 2D-ARMA Tensor Prediction. (a) Tensor Decomposition + 2D ARMA Model. (b) 2D ARMA Model on Rank-r Vector of $\mathbf{U}_T$.

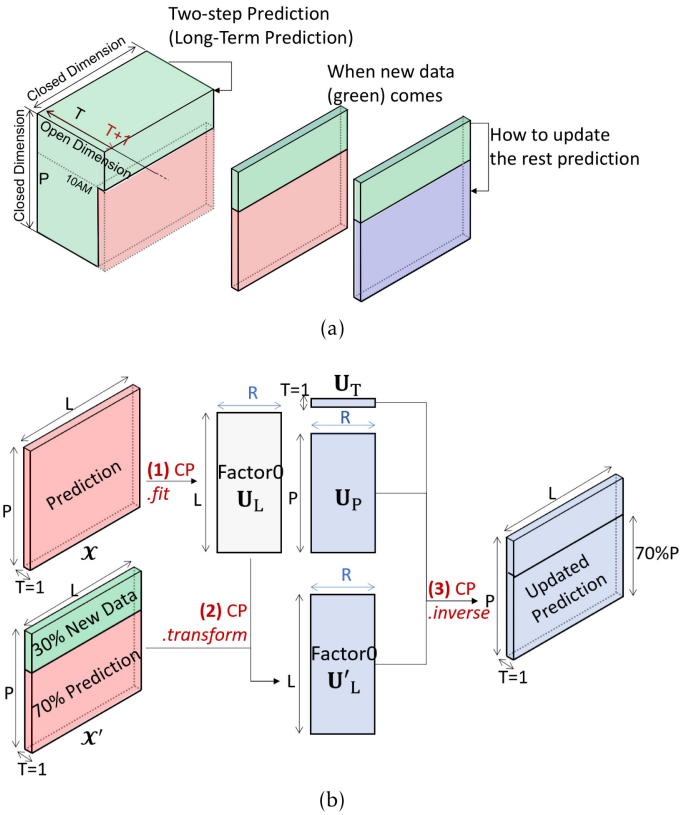

Fig. 4. Proposed method to Update Prediction. (a) 2-step Tensor Prediction Unable to Update Prediction. (b) Lean Dynamic Tensor Decomposition Updating: CP.*fit* is CP composition function, CP.*transform* will be shown in Eq.(8), CP.*inverse* is CP reconstruction function.

be involved in the model, which leads to too much model complexity.

Therefore, we proposed the adoption of 2D-ARMA model for the prediction step (shown in Fig. 3(b)), with following the additional steps as follows:

- For the rank-r vector of $\mathbf{U}_T, r = 1, 2, ..., R$, reshape it into 2D matrix, with each row representing one certain day of the week and each column representing the seven days in one week.
- Train the 2D-ARMA model using $\mathbf{U}_T$.
- After prediction, vectorize the matrix back to vector, and combine all the rank vectors to obtain $\mathbf{U}_{T+\tau}$.

The matrix from rank-r vector $\boldsymbol{u}_r$ of $\mathbf{U}_T$ is represented as a 2D random field $v[d, w], d \in \mathbb{R}^{D=7}, w \in \mathbb{R}^W$. The 2D ARMA$(p_1, p_2, q_1, q_2)$ model is defined for the $D \times W$ for the matrix $\mathbf{V} = \{v[d, w] : 0 \leqslant d \leqslant D-1, 0 \leqslant w \leqslant W-1\}$ by the following equation:

$$\begin{aligned} v[d,w] + \sum_{\substack{i=0 \\ (i,j)\neq(0,0)}}^{p_1} \sum_{j=0}^{p_2} a_{ij} v[d-p_1, w-p_2] \\ = \sum_{i=0}^{q_1} \sum_{j=0}^{q_2} b_{ij} \varepsilon[d-p_1, w-p_2]. \end{aligned} \tag{7}$$

$\{\varepsilon[d, w]\}$ is a stationary white noise with variance σ^2, and the coefficients of $\{a_{ij}\}, \{b_{ij}\}$ are the parameters of the model. (p_1, p_2) and (q_1, q_2) are the time lags of (d, w) for v and ε respectively. The parameter estimation is explained in [22].

Algorithm 1: 2-Step 2D-ARMA Tensor Prediction based on CP Decomposition for Long-Term Prediction.

Input: $\mathcal{X}^{L\times T\times P}$, R, τ, (p_1, p_2, q_1, q_2).
Output: $\mathcal{X}^{L\times(T+\tau)\times P}$.

CP Decomposition: obtain $\mathbf{U}_T$ by Eq.(1)
2D-ARMA Prediction:
for $r = 1$ to R **do**
  Reshape $\boldsymbol{u}_r$: $\boldsymbol{u}_r \in \mathbb{R}^T$ to $\mathbf{V} \in \mathbb{R}^{D\times W}$, with $D \times W \geqslant R$
  for $i = 1$ to τ **do**
    $\mathbf{V}' = f_{2D-ARMA}(\mathbf{V})$ by Eq.(7)
    $\mathbf{V} = \mathbf{V}'$
  end for
  Reshape $\mathbf{V}'$: $\mathbf{V}'$ to $\boldsymbol{u}'_r \in \mathbb{R}^{T+\tau}$
end for
$\mathbf{U}_{T+\tau} = [\boldsymbol{u}'_1, \ldots, \boldsymbol{u}'_r, \ldots, \boldsymbol{u}'_R]$
CP Inverse Reconstruction: obtain $\mathcal{X}^{L\times(T+\tau)\times P}$ by Eq.(1)

The 2-step 2D-ARMA tensor prediction is summarized in Algorithm 1.

Another challenge is to instantly and efficiently update the prediction result when new data come, to which little research has yielded solutions. The problem is specified in Fig. 4(a). After

we obtain the whole predicted passenger flow for tomorrow, when we reach 10:00 AM tomorrow, for example, the new data will have arrived instead. Then how to update our original prediction dynamically and yield more accurate prediction for the rest of the day $T+1$, especially for the next two hours, is a problem. To address this, we propose a lean dynamic tensor decomposition updating method to recursively update the prediction over time. The method is demonstrated in Fig. 4(b). For the long-term prediction result of day $T+1$, $\boldsymbol{\mathcal{X}}^{L\times 1\times P}$, we propose the following procedure:

- Decompose the old long-term prediction result into $\mathbf{U}_P, \mathbf{U}_L$ and $\mathbf{U}_T$, with Factor 0 as $\mathbf{U}_L$, which needs to be updated when new data arrive. An illustration is shown in Fig. 4(b).
- Assume 30% new data of it have come, firstly splice the 30% new data and the rest 70% long-term prediction together, and update $\mathbf{U'}_L$ according to Eq.(8) [12].
- Finally, use the updated $\mathbf{U'}_L$, and the original $\mathbf{U}_T, \mathbf{U}_P$, to reconstruct the tensor.

$$\mathbf{U'}_{\mathrm{L}} = \boldsymbol{\mathcal{X}'}_{(0)}(\mathbf{U}_{\mathrm{T}} \odot \mathbf{U}_{\mathrm{P}})(\mathbf{U}_{\mathrm{T}}^T\mathbf{U}_{\mathrm{T}} * \mathbf{U}_{\mathrm{P}}^T\mathbf{U}_{\mathrm{P}})^{\dagger}, \tag{8}$$

where $\boldsymbol{\mathcal{X}'}_{(0)}$ is the mode-0 unfolding along L-dimension, $\odot$ is Khatri-Rao product and † is Khatri-Rao product pseudoinverse [12].

In practice, we can combine the weight $\boldsymbol{\lambda}$ and factor matrix 0 together and update them when new data arrive.

B. 1-Step Tensor Prediction for Short-Term: Tensor Completion

To adopt the tensor completion framework to the short-term prediction, we treat the historical data as the observation set (with observation indicator as 1), and the future horizon to be predicted as missing data (with observation indicator as 0).

One thing to be noted is that tensor completion is not designed for long-term prediction. In particular, here we define two conceptions: open dimension and closed dimension (as shown in Fig. 4(a)). Open dimension is the dimension that keeps increasing as data arrive, e.g. dimension T-day here; Closed dimension is the one which has fixed maximum length, e.g. dimension L-station, and P-time point. Tensor completion can be used for missing data imputation along a closed dimension (e.g. P and L), but not along an open dimension (T). So it can only achieve short-term prediction.

For tensor completion methods, in particular, we follow the Bayesian LRTC framework proposed by Zhao *et al.* [21].

Denote $\boldsymbol{\Theta} = \{\mathbf{U}^{(1)}, \ldots, \mathbf{U}^{(\mathrm{K})}, \boldsymbol{\lambda}, \tau\}$ from Eq.(2, 3, 4). After calculating the log-joint distribution, and the posterior distribution, the missing data can be estimated after getting $\boldsymbol{\Theta}$, by:

$$P(\boldsymbol{\mathcal{Y}}_{\boldsymbol{\Omega}^c}|\boldsymbol{\mathcal{Y}}_{\boldsymbol{\Omega}}) = \int P(\boldsymbol{\mathcal{Y}}_{\boldsymbol{\Omega}^c}|\boldsymbol{\Theta})P(\boldsymbol{\Theta}|\boldsymbol{\mathcal{Y}}_{\boldsymbol{\Omega}})\,\mathrm{d}\boldsymbol{\Theta}. \tag{9}$$

However, this method still demonstrates some drawbacks as noted in [23]. In particular, when the tensor data violate the innate low-rank structure, such as in our case that the spatial information of URT tensor data is diverse from stations to stations, the low-rank assumption along the spatial dimension cannot hold tenably. Consequently, it is far from enough to assume the spatial prior as low-rank. This most likely oversimplifies the original data structure.

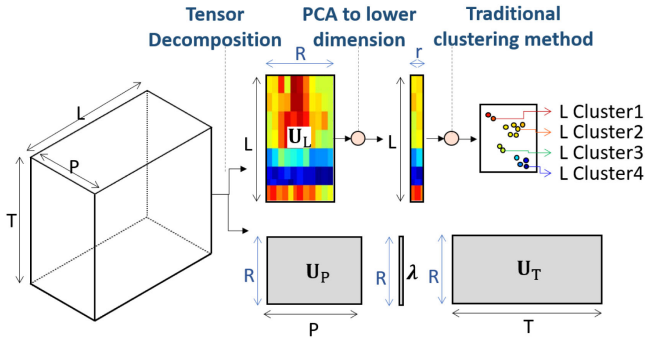

Fig. 5. Tensor Clustering based on Tensor Decomposition

To solve the problem, before the Bayesian LRTC, we proposed to use tensor clustering first [24], [25] to classify the tensor samples into several classes, with highly similar samples within a same cluster. Tensor clustering method is shown in Fig. 5 with the following steps:

- Conduct tensor decomposition to obtain location model matrix.
- Implement principal component analysis to further reduce the dimension.
- Cluster based on a particular clustering method, such as K-mean and hierarchical method.

Thus different URT stations can be divided into several clusters, and Bayesian LRTC can be conducted within each cluster since the homogeneity within-cluster can guarantee its performance.

IV. Experiments

According to what we have discussed in Session III, our URT passenger flow tensor data are $\boldsymbol{\mathcal{X}}^{120\times 59\times 247}$, representing 120 stations, over the past 59 days, with each day 247 sampling points. For long-term prediction, we use data from 5:00AM, 01-Jan-2017 to 01:00AM, 19-Feb-2017 as observed data (training set), data from 05:00AM, 19-Feb-2017 to 01:00AM, 22-Feb-2017 as validation set and data from 05:00AM, 22-Feb-2017 to 1:00AM, 28-Feb-2017 as missing data (testing set). For short-term prediction, we use data from 5:00AM, 01-Jan-2017 to 01:00AM, 19-Feb-2017 as observed data (training set), data from 05:00AM, 19-Feb-2017 to 11:10AM, 20-Feb-2017 as validation set and data from 11:10AM, 20-Feb-2017 to 1:00AM, 21-Feb-2017 as missing data (testing set). Besides, to validate the consistent improvement of our model, additional 7 days (until 7^{th} March) have been added, and all the experiments described above have been conducted by using moving-window, with every experiment moving one day forward. For example, in the first experiment, data from [day-1, day-59] have been used; in the second experiment, data from [day-2, day-60] have been used, with window moving one day forward. And 95% confidence interval has been calculated by the seven experiments.

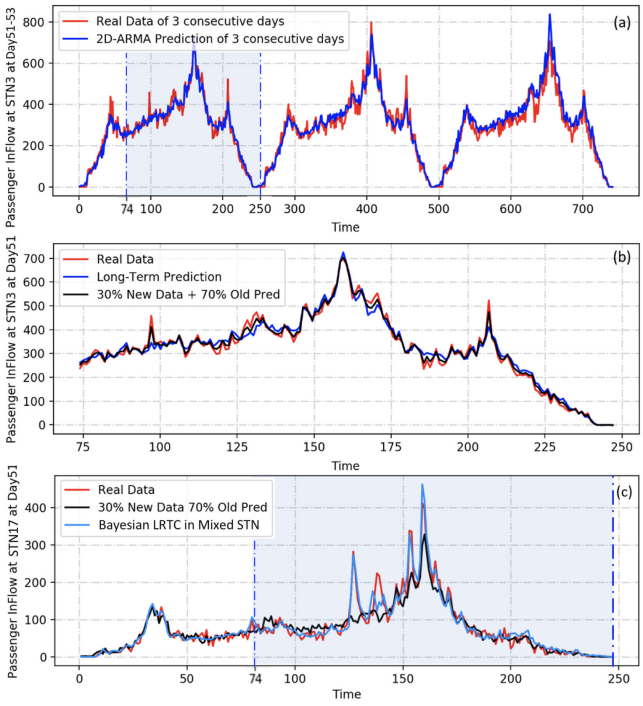

Fig. 6. (a) Inflow Profile Long-Term Prediction for STN3 by 2-step 2D-ARMA prediction; (b) Prediction Improvement for STN3 by involving 30% new data; (c) Inflow Profile Short-Term Prediction for STN17 by 1-step prediction

TABLE I
2-STEP PREDICTION COMPARISON (RES)

Randomly Selected Station Code	1D ARIMA Tensor Prediction (95% Interval)	2D ARMA Tensor Prediction (95% Interval)	Relative improvement (%)
51	$0.1500_{\pm 0.0161}$	$0.1066_{\pm 0.0033}$	**28.90**
56	$0.1043_{\pm 0.0119}$	$0.0759_{\pm 0.0082}$	**27.21**
38	$0.0849_{\pm 0.0258}$	$0.0638_{\pm 0.0432}$	**24.91**
87	$0.1400_{\pm 0.0190}$	$0.1087_{\pm 0.0101}$	**22.39**
54	$0.0939_{\pm 0.0214}$	$0.0739_{\pm 0.0127}$	**21.33**
84	$0.1605_{\pm 0.0139}$	$0.1271_{\pm 0.0107}$	**20.81**
11	$0.0962_{\pm 0.0199}$	$0.0767_{\pm 0.0132}$	**20.29**
16	$0.0871_{\pm 0.0180}$	$0.0703_{\pm 0.0195}$	**19.30**
37	$0.0982_{\pm 0.0197}$	$0.0811_{\pm 0.0178}$	**17.38**
65	$0.1247_{\pm 0.0139}$	$0.1060_{\pm 0.0037}$	**15.00**

A. Proposed 2-Step Tensor Prediction Result

For the tensor $\mathcal{X}^{120\times 50\times 247}$, we conduct CP decomposition. The rank is chosen as 50 by cross-validation, achieving both satisfactory reconstruction and simplicity. For each rank, a 2D ARMA model is constructed and used to predict the coming 6-day-ahead passenger flow. The results of the first 3 days' long-term prediction are shown in Fig. 6(a). In our proposed method, to capture the weekly pattern of the past 2 weeks and the daily pattern of the past 2 days, we set $p_1 = 2, p_2 = 2$, which introduces 8 time-lag components into model. For a fair comparison, the baseline is chosen as the traditional "tensor decomposition + 1D ARIMA" model, with the same model complexity and time-lag equal to 8. The relative residuals (RES) of prediction for next one day on selected stations are shown in Table I. From Table I, we can conclude that our proposed model

TABLE II
IMPROVEMENT ON LONG-TERM PREDICTION (RES)

[t, t+5]	Long-Term Prediction (95% Interval)	Updated after 30% new data (95% Interval)	Relative improvement (%)
t =75	$0.9221_{\pm 0.0229}$	$\mathbf{0.8436}_{\pm 0.0168}$	**8.520**
t =80	$0.5833_{\pm 0.0199}$	$\mathbf{0.3920}_{\pm 0.0199}$	**32.79**
t =85	$0.7136_{\pm 0.0279}$	$\mathbf{0.6258}_{\pm 0.0267}$	**12.29**
t =90	$1.2381_{\pm 0.0323}$	$\mathbf{1.2042}_{\pm 0.0320}$	**2.735**
t =95	$1.2543_{\pm 0.0251}$	$\mathbf{1.1387}_{\pm 0.0280}$	**9.215**
t =100	$1.0423_{\pm 0.0262}$	$\mathbf{0.9729}_{\pm 0.0322}$	**6.659**
t =105	$0.1153_{\pm 0.0280}$	$\mathbf{0.0743}_{\pm 0.0290}$	**35.57**
t =110	$0.3452_{\pm 0.0311}$	$\mathbf{0.3379}_{\pm 0.0321}$	**2.113**
t =115	$0.4022_{\pm 0.0245}$	$\mathbf{0.3163}_{\pm 0.0278}$	**21.36**
t =120	$0.2292_{\pm 0.0219}$	$0.3084_{\pm 0.0235}$	-34.54
t =125	$0.6245_{\pm 0.0398}$	$0.6865_{\pm 0.0424}$	-9.936

can achieve almost 20% improvements. The overall improvement benefits from the advantage that more strongly correlated temporal patterns have been considered in our model. Note that the scale of improvement varies. This is because some stations may also have a strong temporal correlation with the third, the fourth, and even higher-order time-lags, which yet our model ignores.

When the first 30% new data (in Fig. 6(a) until time stamp $t = 74$) have arrived, the prediction time stamp for the rest 70% of that day (In Fig. 6(a) highlighted in blue block, from time stamp $t = 75$ to $t = 247$) needs to be updated instantly. By using the proposed lean dynamic updating procedure, the rest 70% has been recalculated, as shown in Fig. 6(b). It is clear to observe that the updated prediction is significantly improved with a smaller distance to the real value, especially around the local peak time. To further check the improvement of the rest 70%, RES of updated prediction of Station 3 is calculated from $t = 74$ for every 25 minutes (i.e., 5 time stamps). According to Table II, after involving the first 30% of new data, there is an obvious improvement by around 20% (highlighted in boldface) over the following 3 hours (from $t = 75$ to $t = 115$). After then, the relative improvement based on short-term updating becomes less efficient, with the long-term prediction still being preferred.

B. Proposed Bayesian LRTC Result

For the Bayesian LRTC, some stations from mixed clusters have been randomly picked, and the data since $t = 74$ (around 10 AM) of the last day are to be predicted. The prediction result is shown in Fig. 6(c). The Bayesian LRTC can reduce RES by 29% for station 17 in Fig. 6(c), with some other stations achieving around 10% to 30% smaller residual, as shown in Table III. The good performance for tensor completion in short-term prediction is quite satisfactory compared with the proposed lean dynamic updating method. However, according to Table III, it is to be noted that this improvement (highlighted in boldface) is not universal, with Stations 84, 55, 51, and 87 having a worse prediction. This is because these four stations have quite unique and distinct passenger flow patterns and the low-rank assumption no longer holds.

This can be solved by conducting Bayesian LRTC within the same station cluster. We use the hierarchical clustering with the agglomerative method, where the distance is defined as group

TABLE III
PREDICTION COMPARISON FOR MIXED-CLUSTER (RES)

Station Code (Mixed Cluster)	Updated after 30% new data (95% Interval)	Bayesian LRTC (95% Interval)	Relative improvement (%)
17	$0.1165_{\pm 0.0103}$	$0.0821_{\pm 0.0014}$	**29.59**
56	$0.0885_{\pm 0.0119}$	$0.0649_{\pm 0.0112}$	**26.66**
54	$0.0833_{\pm 0.0331}$	$0.0704_{\pm 0.0042}$	**15.38**
38	$0.1775_{\pm 0.0134}$	$0.1521_{\pm 0.0238}$	**14.31**
35	$0.1020_{\pm 0.0152}$	$0.0909_{\pm 0.0101}$	**10.89**
11	$0.1416_{\pm 0.0296}$	$0.1326_{\pm 0.0059}$	**6.38**
84	$0.0632_{\pm 0.0175}$	$0.0691_{\pm 0.0100}$	-9.38
55	$0.1240_{\pm 0.0411}$	$0.1366_{\pm 0.0165}$	-10.17
51	$0.1090_{\pm 0.0249}$	$0.1279_{\pm 0.0108}$	-17.38
87	$0.1205_{\pm 0.0302}$	$0.1609_{\pm 0.0112}$	-33.60

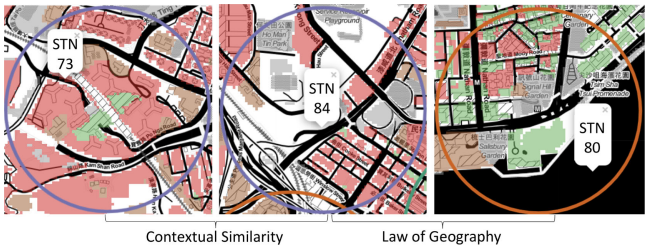

Fig. 7. Land-use Information of Selected Stations (Different Color Pixel Denotes Different Land-use)

TABLE IV
IMPROVEMENT ON LRTC BY SAME CLUSTER (RES)

Station Code (Same Cluster)	Updated after 30% new data (95% Interval)	Bayesian LRTC within a cluster (95% Interval)	Relative improvement (%)
82	$0.1149_{\pm 0.0195}$	$0.0673_{\pm 0.0113}$	**41.43**
77	$0.1775_{\pm 0.0401}$	$0.1133_{\pm 0.0390}$	**36.18**
79	$0.2112_{\pm 0.0295}$	$0.1349_{\pm 0.0248}$	**36.10**
71	$0.1109_{\pm 0.0149}$	$0.0766_{\pm 0.0055}$	**30.89**
84	$0.0632_{\pm 0.0207}$	$0.0489_{\pm 0.0160}$	**22.59**
75	$0.1593_{\pm 0.0219}$	$0.1288_{\pm 0.0074}$	**19.15**
74	$0.1642_{\pm 0.0138}$	$0.1409_{\pm 0.0097}$	**14.19**
73	$0.1773_{\pm 0.0578}$	$0.1567_{\pm 0.0324}$	**11.61**
80	$0.1187_{\pm 0.0207}$	$0.1087_{\pm 0.0061}$	**8.41**
87	$0.1205_{\pm 0.0191}$	$0.1146_{\pm 0.0062}$	**4.92**

average Euclidean. The clustering result reflects two types of spatial correlations observed in our data: Law of Geography and contextual similarity. In other words, if two stations are geographically close or functionally similar, they are in the same cluster. For example, Fig. 7 shows the land-use information of three selected stations (Stations 73, 80, and 84) in one cluster. Though Stations 73 and 84 are far from each other, they share the same land-use pattern (with red dominant, mix-use in grey and brown). Though Stations 84 and 80 are quite different in land-use (Station 84 is dominated with red, Station 80 is dominated with green), they are geographically close (since the stations are indexed consecutively, two stations with close codes indicate that they are physically close.)

By selecting stations within this cluster, we compare the Bayesian LRTC within one cluster with 2-step prediction updated with 30% new data, and the result is shown in Table IV.

Predictions of all the selected stations have been improved (with improvement highlighted in boldface), with the majority improved by 20%. Most importantly, even for the same Stations 84 and 87 in two cases(highlighted with the underline in Table III and IV), conducting Bayesian LRTC within one cluster can improve the prediction by 30% to 40%.

V. CONCLUSION

In this paper, we focused on both Long-term and Short-term URT passenger flow predictions. Our proposed 2-step tensor prediction based on tensor decomposition and time-series model can achieve both long-term and short-term predictions. For the long-term prediction, we propose the "CP Decomposition + 2D ARMA model", which combines the CP decomposition and the 2D ARMA into a unified model. Therefore, it can achieve satisfactory long-term prediction performance given its ability to use the data from both last week or last day and the use of CP decomposition to model the complicated correlation structures. For the short term prediction, we propose to combine 1-step tensor prediction based on Bayesian low rank tensor completion and a tensor cluster technique. In the future, we propose to combine the benefits of both tensor completion and 2D-ARMA for a unified model that can be applied to both long-term and short-term predictions.

ACKNOWLEDGMENT

The authors would like to show their great appreciation to the Metro Corporation for sharing this passenger flow data, and in the protection of privacy, all data have been desensitized. They also give special thanks to Dr. Qibin Zhao for sharing the scripts of their methods.

REFERENCES

[1] M. T. Bahadori, Q. R. Yu, and Y. Liu, "Fast multivariate spatio-temporal analysis via low rank tensor learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3491–3499.
[2] L. Zhao, Q. Sun, J. Ye, F. Chen, C.-T. Lu, and N. Ramakrishnan, "Multi-task learning for spatio-temporal event forecasting," in *Proc. 21th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 1503–1512.
[3] R. Zhong, W. Lv, B. Du, S. Lei, and R. Huang, "Spatiotemporal multi-task learning for citywide passenger flow prediction," in *Proc. IEEE SmartWorld, Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People Smart City Innov.*, 2017, pp. 1–8.
[4] D. Tikunov and T. Nishimura, "Traffic prediction for mobile network using holt-winter's exponential smoothing," in *Proc. 15th Int. Conf. Softw., Telecommun. Comput. Netw.*, 2007, pp. 1–5.
[5] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, 2003.
[6] X. Zhang, L. Zhao, A. P. Boedihardjo, C.-T. Lu, and N. Ramakrishnan, "Spatiotemporal event forecasting from incomplete hyper-local price data," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 507–516.
[7] H. Yao *et al.*, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 2588–2595.
[8] X. Geng *et al.*, "Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 3656–3663.
[9] D. M. Dunlavy, T. G. Kolda, and E. Acar, "Temporal link prediction using matrix and tensor factorizations," *ACM Trans. Knowl. Discovery Data*, vol. 5, no. 2, pp. 1–27, 2011.

[10] Q. Guo and H. A. Karimi, "A novel methodology for prediction of spatial-temporal activities using latent features," *Comput., Environ. Urban Syst.*, vol. 62, pp. 74–85, 2017.
[11] J. Ren and Q. Xie, "Efficient od trip matrix prediction based on tensor decomposition," in *Proc. 18th IEEE Int. Conf. Mobile Data Manage.*, 2017, pp. 180–185.
[12] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.
[13] H. Tan, Y. Wu, B. Shen, P. J. Jin, and B. Ran, "Short-term traffic prediction based on dynamic tensor completion," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 8, pp. 2123–2133, Aug. 2016.
[14] B. Ran, H. Tan, Y. Wu, and P. J. Jin, "Tensor based missing traffic data completion with spatial–temporal correlation," *Physica A: Statistical Mech. Appl.*, vol. 446, pp. 54–63, 2016.
[15] J. Luan and Z. Zhang, "Prediction of multi-dimensional spatial variation data via Bayesian tensor completion," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2019.
[16] X. Chen, Z. He, and L. Sun, "A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation," *Transp. Res. Part C: Emerg. Technol.*, vol. 98, pp. 73–84, 2019.
[17] D. Kressner, M. Steinlechner, and B. Vandereycken, "Low-rank tensor completion by riemannian optimization," *BIT Numer. Math.*, vol. 54, no. 2, pp. 447–468, 2014.
[18] T. Yokota, Q. Zhao, and A. Cichocki, "Smooth parafac decomposition for tensor completion," *IEEE Trans. Signal Process.*, vol. 64, no. 20, pp. 5423–5436, Oct. 2016.
[19] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 208–220, Jan. 2012.
[20] Q. Shi, H. Lu, and Y.-m. Cheung, "Tensor rank estimation and completion via CP-based nuclear norm," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 949–958.
[21] Q. Zhao, L. Zhang, and A. Cichocki, "Bayesian CP factorization of incomplete tensors with automatic rank determination," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1751–1763, Sep. 2015.
[22] J. Zielinski, N. Bouaynaya, and D. Schonfeld, "Two-dimensional arma modeling for breast cancer detection and classification," in *Proc. Int. Conf. Signal Process. Commun.*, 2010, pp. 1–4.
[23] L. Yuan, Q. Zhao, L. Gui, and J. Cao, "High-dimension tensor completion via gradient-based optimization under tensor-train format," 2018. [Online]. Available: https://arxiv.org/abs/1804.01983
[24] K. Yu, L. He, S. Y. Philip, W. Zhang, and Y. Liu, "Coupled tensor decomposition for user clustering in mobile internet traffic interaction pattern," *IEEE Access*, vol. 7, pp. 18 113–18 124, 2019.
[25] W. W. Sun and L. Li, "Dynamic tensor clustering," *J. Amer. Statistical Assoc.*, vol. 114, no. 528, pp. 1894–1907, 2019.