Transportation Research Part C 117 (2020) 102671

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Graph Markov network for traffic forecasting with missing data

Zhiyong Cui[a], Longfei Lin[b], Ziyuan Pu[a,*], Yinhai Wang[a]

[a] Department of Civil and Environmental Engineering, University of Washington, Seattle, WA 98195, USA
[b] School of Electronic and Information Engineering, Beihang University, 100191 Beijing, China

ARTICLE INFO

Keywords:
Traffic forecasting
Neural network
Missing values
Traffic network
Graph Markov process
Graph convolution

ABSTRACT

Traffic forecasting is a classical task for traffic management and it plays an important role in intelligent transportation systems. However, since traffic data are mostly collected by traffic sensors or probe vehicles, sensor failures and the lack of probe vehicles will inevitably result in missing values in the collected raw data for some specific links in the traffic network. Although missing values can be imputed, existing data imputation methods normally need long-term historical traffic state data. As for short-term traffic forecasting, especially under edge computing and online prediction scenarios, traffic forecasting models with the capability of handling missing values are needed. In this study, we consider the traffic network as a graph and define the transition between network-wide traffic states at consecutive time steps as a graph Markov process. In this way, missing traffic states can be inferred step by step and the spatial–temporal relationships among the roadway links can be incorporated. Based on the graph Markov process, we propose a new neural network architecture for spatial–temporal data forecasting, i.e. the graph Markov network (GMN). By incorporating the spectral graph convolution operation, we also propose a spectral graph Markov network (SGMN). The proposed models are compared with baseline models and tested on three real-world traffic state datasets with various missing rates. Experimental results show that the proposed GMN and SGMN can achieve superior prediction performance in terms of both accuracy and efficiency. Besides, the proposed models' parameters, weights, and predicted results are comprehensively analyzed and visualized.

1. Introduction

Traffic forecasting, as a challenging topic for both academia and industry, has been under active research, development, and implementation for more than 40 years (Laña et al., 2018). Traffic forecasting plays an important role in transportation management and the general planning process. With the exponential increase in the volume of traffic data and the computational capability, traffic forecasting methods have been gradually shifting from classical statistical models to data-driven machine learning-based methods (Vlahogianni et al., 2014). In recent years, the rise of artificial intelligence (AI), especially deep learning methods, has dramatically stimulated the traffic forecasting research field. By leveraging the spatial–temporal patterns in immense traffic data, many deep neural network models, including recurrent neural network (RNN), convolutional neural network (CNN), generative adversarial network (GAN), etc., have been widely applied in traffic forecasting studies and achieved state-of-the-art prediction performance.

However, since network-wide traffic state data are mostly collected by traffic sensors or probe vehicles, sensor failures or irregular sampling from probe vehicles will result in missing values in the collected data. The missing value issue usually leads an apparent decline in the forecasting performance, as most of the existing methods for traffic forecasting are not capable of dealing with missing

* Corresponding authors.
E-mail addresses: zhiyongc@uw.edu (Z. Cui), linlongfei9858@buaa.edu.cn (L. Lin), ziyuanpu@uw.edu (Z. Pu), yinhai@uw.edu (Y. Wang).

https://doi.org/10.1016/j.trc.2020.102671
Received 3 October 2019; Received in revised form 23 April 2020; Accepted 20 May 2020
Available online 10 July 2020
0968-090X/ Published by Elsevier Ltd.

values. Thus, the forecasting performance of the models which only accept complete data as input will be significantly affected and limited.

The regular solution for the missing data issue is to conduct data imputation, which targets to estimate corrupted or missing traffic data. Because of the complex spatial–temporal patterns of traffic data, existing novel and effective data imputation methods, such as the tensor decomposition-based approaches (Tan et al., 2016; Chen et al., 2019), usually need large datasets covering a long period of time to achieve good imputation performance. However, a large dataset covering a long period of time is not always available. Further, in the connected vehicle environments or under the edge computing scenarios, online forecasting computations need to be completed in devices with limited storage and computational capabilities. However, those tensor decomposition-based models require hundreds of iterations to converge, which might take tens of minutes, and achieve their best imputation performance (Tan et al., 2016; Chen et al., 2019). Thus, when processing large datasets, these solutions may not be feasible for real-time traffic forecasting tasks.

To solve the missing value issue and fulfill traffic forecasting at the same time, traffic forecasting models with the capability of dealing with missing values have also been proposed. However, most of the existing models (Zhang and Zhang, 2016; Lee and Fambro, 1999) take the spatial–temporal traffic data as multivariate time series, and thus, they neglect the important spatial influence between the road links in the traffic network. There are several deep learning-based methods (Duan et al., 2016; Tian et al., 2018) taking spatial factors into consideration. However, they still cannot incorporate the intrinsic structure of the traffic network into the traffic forecasting process.

In this study, to overcome the problems mentioned above, we propose a graph Markov network (GMN), which is a new neural network architecture for spatial–temporal data forecasting with missing values. The traffic network is converted into a graph with topological properties. We consider the variations of the traffic states in the traffic network has a Markov property and a graph localization property. Based on the two properties, the traffic state transition process can be considered as a graph Markov process. The GMN is designed based on the graph Markov process, which inherently incorporates the spatial–temporal relationships among the links in the traffic network. By incorporating the spectral graph convolution operation, we also propose a spectral graph Markov network (SGMN). Experimental results indicate that the proposed SGMN and GMN can achieve superior prediction performance with greater efficiency.

The contributions of this study can be summarized as follows:

1. We consider the traffic network as a graph and define the transition between network-wide traffic states at consecutive time steps as a graph Markov process.
2. We propose a new neural network structure, i.e. the graph Markov network, based on the proposed graph Markov process for dealing with missing values and forecasting traffic state simultaneously.
3. By incorporating the spectral graph convolution operation, we also propose a spectral graph Markov network.
4. Experimental results tested on three real-world network-wide traffic state datasets show that the proposed models can achieve superior prediction performance in terms of both accuracy and efficiency.

The rest of this paper is organized as follows: the second section describes the related studies on traffic forecasting with missing values. The third section introduces the proposed graph Markov process and the proposed GMN model. The fourth section discusses the experimental results and the concluding remarks are presented in the fifth section.

2. Literature review

Classical traffic forecasting models can generally be classified into two categories, traditional statistical models and computational intelligence, i.e. machine learning-based, models (Vlahogianni et al., 2014). The statistical methods are mostly parametric approaches, including variants of auto-regressive integrated moving average (ARIMA) models (Williams, 2001), parametric Kalman filtering models (Okutani and Stephanedes, 1984), and other types of time-series models (Ghosh et al., 2009), that are developed based on a predefined model structure with theoretical assumptions and the parameters are calibrated using historical data (Smith et al., 2002). With the ability to accommodate the stochastic and non-linear nature of traffic patterns, classical machine learning methods are widely adopted for the traffic forecasting task, such as support vector regression (Wu et al., 2004), Bayesian network approaches (Sun et al., 2006). In recent years, with the rise of AI, the performance of emerging deep learning-based traffic forecasting methods outperform that of classical methods.

2.1. Deep learning-based traffic forecasting methods

Since the traffic data contain both spatial and temporal attributes, the deep learning-based methods can be grouped by the ways to deal with spatial–temporal traffic data. One type of studies convert the spatial–temporal data into a 2-dimensional (2D) matrix and use long short-term memory (LSTM) recurrent neural network (Ma et al., 2015), bi-directional LSTM (Cui et al., 2016), CNN (Ma et al., 2017), GAN (Liang et al., 2018), or a combination of multiple models (Yang et al., 2019), to extract feature and forecast traffic states. However, a traffic network's spatial features cannot be completely represented by a 2D matrix. Thus, another type of methods (Yu et al., 2017; Ma et al., 2018) is proposed to convert the physical roadway networks as images according to roads' geospatial properties. Although the traffic network images demonstrate the true traffic network structure, those images contain too many noisy pixels and blank pixels without traffic state information. To analyze the traffic network in an efficient way, many studies consider the

traffic network as a graph and predict traffic state by incorporating the graph convolutional network (Yu et al., 2018; Cui et al., 2019; Li et al., 2017).

2.2. Deep learning-based traffic forecasting with missing values

Traffic forecasting performance will be influenced by the missing values (Cui et al., 2016). A bunch of data imputation methods has been developed to solve the missing values issues, including the probabilistic principal component analysis (PCA) (Li et al., 2014), tensor decomposition-based methods (Tan et al., 2016; Ran et al., 2016; Chen et al., 2018, 2019), clustering approaches (Tang et al., 2015; Ku et al., 2016). There are also some deep learning-based data imputation methods proposed in the most recent years, such as denoising stacked auto-encoder (Duan et al., 2016) and generative adversarial imputation network (Yoon et al., 2018). However, the PCA-based (Li et al., 2014) and tensor decomposition-based (Tan et al., 2016; Ran et al., 2016; Chen et al., 2018, 2019) models normally need hundreds or thousands of iterations to converge and achieve their best data imputation performance. Further, the aforementioned deep learning models for data imputation are not originally designed for solving the traffic forecasting with missing values issue.

To combine the data imputation and traffic forecasting together, a few RNN-based approaches, such as the LSTM-M (Tian et al., 2018), have been proposed based the GRU-D (Che et al., 2018) for processing multivariate time series with missing values. Even though these RNN-based methods can recurrently fill missing values in each time step and forecasting the future traffic state, they cannot capture spatial interactions between road links in the traffic network. Further, although recent research (Du et al., 2019; Barredo-Arrieta et al., 2019) is trying to interpret RNN-based models, for the traffic forecasting problem, it is still hard for RNN-based models to interpret the spatial relationship between neighboring links and the links' temporal dependencies between different time steps.

To solve these problems, in this study, we consider the traffic state transition process as a graph Markov process and propose the graph Markov network for the traffic forecasting with missing values. The design of the graph Markov network inherently incorporates the spatial–temporal relationships among the links in the traffic network. The graph Markov network making use of the topological structure of the traffic network can achieve accurate prediction results with efficient training and testing process. The proposed graph Markov process and graph Markov network are introduced in detail in the following section.

3. Problem definition and preliminary

3.1. Traffic forecasting

A traffic network normally consists of multiple roadway links. The traffic forecasting task targets to predict future traffic states of all (road) links or sensor stations in the traffic network based on historical traffic state data. The collected spatial–temporal traffic state data of a traffic network with S links/sensor-stations can be characterized as a T-step sequence $[x_1, x_2, ..., x_t, ..., x_T] \in \mathbb{R}^{T\times S}$, in which $x_t \in \mathbb{R}^S$ demonstrates the traffic states of all S links at the t-th step. The traffic state of the s-th link at time t is represented by x_t^s. In this study, the superscript of a traffic state represents the spatial dimension and the subscript denotes the temporal dimension. The short-term traffic forecasting problem can be formulated as, based on T-step historical traffic state data, learning a function $F(\cdot)$ to generate the traffic state at next time step as follows:

$$F([x_1, x_2..., x_T]) = [x_{T+1}] \tag{1}$$

3.2. Graph representations of traffic network

Since the traffic network is composed of road links and intersections, it is intuitive to consider the traffic network as an undirected graph consisting of vertices and edges. The graph can be denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{A}, \mathcal{D})$ with a set of vertices $\mathcal{V} = \{v_1, ..., v_S\}$ and a set of edges $\mathcal{E}$ between vertices. $\mathcal{A} \in \mathbb{R}^{S\times S}$ is a symmetric (typically sparse) adjacency matrix with binary elements, where $\mathcal{A}_{i,j}$ denotes the connectedness between nodes v_i and v_j. The existence of an edge is represented through $\mathcal{A}_{i,j} = \mathcal{A}_{j,i} = 1$, otherwise $\mathcal{A}_{i,j} = 0$ ($\mathcal{A}_{i,i} = 0$). Based on $\mathcal{A}$, a diagonal graph degree matrix $\mathcal{D} \in \mathbb{R}^{S\times S}$ describing the number of edges attached to each vertex can be obtained by $\mathcal{D}_{i,i} = \sum_{j=1}^{S} \mathcal{A}_{i,j}$.

The $\mathcal{A}$ can only indicate the relationship between different vertices. In some cases, the vertices' relationship with themselves also needs to be characterized. Thus, we define the self-connection adjacency matrix $\mathbf{A} = \mathcal{A} + I$, i.e. $\mathbf{A}_{i,i} = 1$, which implies each vertex in the graph is self-connected. Here, $I \in \mathbb{R}^{S\times S}$ is an identity matrix.

In addition, the connectedness of the graph vertices can also be encoded by the Laplacian matrix, which is essential for spectral graph analysis. The combinatorial Laplacian matrix is defined as $\mathcal{L} = \mathcal{D} - \mathcal{A}$ and the normalized definition is $\mathcal{L} = I - \mathcal{D}^{-1/2}\mathcal{A}\mathcal{D}^{-1/2}$. Since $\mathcal{L}$ is a symmetric positive semi-definite matrix, it can be diagonalized as $\mathcal{L} = U\Lambda U^T$ by its eigenvector matrix U (Defferrard et al., 2016), where $U = [u_0, u_1, ..., u_{S-1}] \in \mathbb{R}^{S\times S}$ and $\Lambda = \mathrm{diag}(\lambda_0, \lambda_1, ..., \lambda_{S-1}) \in \mathbb{R}^{S\times S}$ is the corresponding diagonal eigenvalue matrix satisfying $\mathcal{L}u_i = \lambda_i u_i$.

In this study, under the traffic forecasting scenario, the attribute on vertex v_s (road link s) at time t is denoted as x_t^s. Given the graph representation of the traffic network, the Eq. (1) can be extended as

$$F(\mathcal{G}, [x_1, x_2..., x_T]) = [x_{T+1}] \tag{2}$$

3.3. Traffic forecasting with missing values

Traffic state data can be collected by multi-types of traffic sensors or probe vehicles. When traffic sensors fail or no probe vehicles go through road links, the collected traffic state data may have missing values. We use a sequence of masking vectors $[m_1, m_2, ..., m_T] \in \mathbb{R}^{T\times S}$, where $m_t \in \mathbb{R}^S$, to indicate the position of the missing values in traffic state sequence $[x_1, x_2, ..., x_T]$. The masking vector can be obtained by

$$m_t^s = \begin{cases} 1, & \text{if } x_t^s \text{is observed} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where x_t^s is the traffic state of s-th link at step t.

Missing values in traffic data can be handled by many existing data imputation methods. Most state-of-the-art data imputation methods, such as the Bayesian tensor decomposition approach (Chen et al., 2019) and the Generative Adversarial Imputation Nets (GAIN) (Yoon et al., 2018), need long-term historical data to capture complicated traffic patterns and fill missing values. However, in real-time environments, especially under the connected autonomous vehicle (CAV) and edge computing scenarios, it may not be possible to conduct data imputation on historical data and forecast future traffic states sequentially, because the volume of traffic state data is huge and the computing capability of devices is limited. In these cases, the traffic forecasting models should be able to handle missing values. Taking the missing values into consideration, we can formulate the traffic forecasting as follows

$$F(\mathcal{G}, [x_1, x_2..., x_T], [m_1, m_2..., m_T]) = [x_{T+1}] \tag{4}$$

4. Proposed approach

In this section, we first describe several properties of traffic states. Based on that, we propose a graph Markov process to characterize the variations of traffic states. Then, we introduce the proposed graph Markov Network for traffic forecasting with the capability of dealing with missing values.

4.1. Properties

A traffic network is a dynamic system and the states on all links keep varying resulted by the movements of vehicles in the system. Thus, we assume the traffic network's dynamic process satisfies the Markov property that the future state of the traffic network is conditional on the present state.

Markov property: The future state of the traffic network x_{t+1} depends only upon the present state x_t, not on the sequence of states that preceded it. Taking $X_1, X_2, ..., X_{t+1}$ as random variables with the Markov property and $x_1, x_2, ..., x_{t+1}$ as the observed traffic states. The Markov process can be formulated in a conditional probability form as

$$Pr(X_{t+1} = x_{t+1} | X_1 = x_1, X_2 = x_2, ..., X_t = x_t) = Pr(X_{t+1} = x_{t+1} | X_t = x_t) \tag{5}$$

where $Pr(\cdot)$ demonstrates the probability.

However, the transition matrix is temporal dependent, since at the different time of a day, the traffic state's transition pattern should be different. Based on Eq. (5), the transition process of traffic states can be formulated in the vector form as

$$x_{t+1} = P_t x_t \tag{6}$$

where $P_t \in \mathbb{R}^{S\times S}$ is the transition matrix and $(P_t)_{i,j}$ measures how much influence x_t^j has on forming the state x_{t+1}^i.

The transition process defined in Eq. (6) does not take the time interval between x_{t+1} and x_t into consideration. We denote the time interval between two consecutive time steps of traffic states by Δt. If Δt is small enough ($\Delta t \to 0$), the traffic network's dynamic process can be measured as a continuous process and the difference between consecutive traffic states are close to zero, i.e. $|x_{t+\Delta t} - x_t| \to 0$. However, a long time interval may result in more variations between the present and future traffic states, leading to a more complicated transition process. Since the traffic state data are normally processed into discrete data and the size of transition matrix P_t is fixed, we consider that the longer the Δt is, the lower capability of measuring the actual transition process P_t has. Thus, we multiply a decay parameter $\gamma \in (0, 1)$ in Eq. (6) to represent the temporal impact on the transition process as

$$x_{t+1} = \gamma P_t x_t \tag{7}$$

The transition matrix can measure the contributions made by all roadway links on a specific link, which assumes that the state of a roadway link is influenced by all links in the traffic network. However, since vehicles in the traffic network traverse connected road links one by one and traffic states of connected links are transmitted by those vehicles, the traffic state of a link will only be affected by its neighboring links during a short period of time.

Graph localization property: The traffic state of a specific link s in a traffic network is mostly influenced by localized links, i.e. the link s itself and its neighboring links, during a short period of time. The neighboring links refer to the links in the graph within a specific order of hops with respect to the link s. With the help of the graph's topological structure, the localization property in the graph can be measured based the adjacency matrix in two ways: (1) The self-connection adjacency matrix $\mathbf{A}$, describing the connectedness of vertices, can inherently indicate the localization property of all vertices in the graph. Then, the impacts of localized links can be easily measured by a weighted self-connection adjacency matrix. (2) The other way is to conduct the spectral graph

convolution operation on the traffic state x_t to measure the localized impacts in the graph. The spectral graph convolution on x_t can be defined as $U\Lambda_\theta U^T x_t$ (Defferrard et al., 2016), where U is the eigenvector matrix of the Laplacian matrix $\mathcal{L}$ and Λ_θ is a learnable diagonal weight matrix.

Graph Markov Process: With the aforementioned two properties, we define the traffic state transition process as a graph Markov process (GMP). The graph Markov process can be formulated in a conditional probability form as

$$Pr(X_{t+1} = x_{t+1}^u | X_t = x_t) = Pr(X_{t+1} = x_{t+1}^u | X_t = x_t^v, v \in \mathcal{N}(u)) \tag{8}$$

where the superscripts u and v are the indices of graph links (road links). The $\mathcal{N}(u)$ denotes a set of one-hop neighboring links of link u and link u itself. The properties of this graph Markov process is similar to the properties of the Markov random field (Rue and Held, 2005) with temporal information. Since the influence of a road link on its neighbors is gradually spread by the vehicles traveling on this road link, a road link's one-hop neighbors are the ones directly influencing it. Thus, we assume that road links are only influenced by their one-hop neighbors in the graph. Based on Eq. (7), we can easily incorporate the graph localization properties into the traffic states' transition process by element-wise multiplying the transition matrix P_t with the self-connection adjacency matrix $\mathbf{A}$. Then, the GMP can be formulated in the vector form as

$$x_{t+1} = \gamma(\mathbf{A} \odot P_t)x_t \tag{9}$$

where $\odot$ is the Hadamard (element-wise) product operator that $(\mathbf{A} \odot P_t)_{ij} = \mathbf{A}_{ij} \times (P_t)_{ij}$.

The graph localization property can also be incorporated in the transition process by replacing the transition weight matrix P_t with the spectral graph convolution operation. Then, we define the spectral version of the graph Markov process (SGMP) as

$$x_{t+1} = \gamma U \Lambda_{\theta_t} U^T x_t \tag{10}$$

where $\Lambda_{\theta_t} \in \mathbb{R}^{S \times S}$ is a diagonal weight matrix.

4.2. Handling missing values in the graph Markov process

In this section, we theoretically introduce how to deal with the missing values in the graph Markov process.

As we assume the traffic state transition process follows the graph Markov process, the future traffic state can be inferred by the present state. If there are missing values in the present state, we can infer the missing values from previous states. We consider x_t is the observed traffic state at time t and a mask vector m_t can be acquired according to Eq. (3). We denote the completed state by $\widetilde{x}_t$, in which all missing values are filled based on historical data. Hence, the completed state consists of two parts, including the observed state values and the inferred state values, as follows:

$$\widetilde{x}_t = x_t \odot m_t + \widetilde{x}_t \odot (1 - m_t) \tag{11}$$

where $\widetilde{x}_t \odot (1 - m_t)$ is the inferred part. Since $x_t \odot m_t = x_t$, Eq. (11) can be written as

$$\widetilde{x}_t = x_t + \widetilde{x}_t \odot (1 - m_t) \tag{12}$$

Since the transition of completed states follows the graph Markov process, the GMP and SGMP with respect to the completed state can be described as $\widetilde{x}_{t+1} = \gamma(\mathbf{A} \odot P_t)\widetilde{x}_t$ and $\widetilde{x}_{t+1} = \gamma U \Lambda_{\theta_t} U^T \widetilde{x}_t$, respectively. In this section, for simplicity, we denote the (spectral) graph Markov transition matrix by H_t, i.e. $H_t = \mathbf{A} \odot P_t$ or $H_t = U\Lambda_{\theta_t}U^T$. Hence, the transition process of completed states can be represented as

$$\widetilde{x}_{t+1} = \gamma H_t \widetilde{x}_t \tag{13}$$

Applying Eq. (12), the transition process becomes

$$\widetilde{x}_{t+1} = \gamma H_t (x_t + \widetilde{x}_t \odot (1 - m_t)) \tag{14}$$

If we iteratively apply the completed state $\widetilde{x}_t$, i.e. $\widetilde{x}_t = \gamma H_{t-1}(x_{t-1} + \widetilde{x}_{t-1} \odot (1 - m_{t-1}))$, into Eq. (14) itself, we have

$$\begin{aligned}\widetilde{x}_{t+1} =& \gamma H_t (x_t + \gamma H_{t-1}(x_{t-1} + \widetilde{x}_{t-1} \odot (1 - m_{t-1})) \odot (1 - m_t)) \\ =& \gamma H_t x_t + \gamma^2 H_t H_{t-1}(x_{t-1} \odot (1 - m_t)) + \gamma^2 H_t H_{t-1}(\widetilde{x}_{t-1} \odot (1 - m_{t-1}) \odot (1 - m_t))\end{aligned} \tag{15}$$

After iteratively applying n steps of previous states from $x_{t-(n-1)}$ to x_t, $\widetilde{x}_{t+1}$ can be described as

$$\begin{aligned}\widetilde{x}_{t+1} =& \gamma H_t x_t \\ &+ \gamma^2 H_t H_{t-1}(x_{t-1} \odot (1 - m_t)) \\ &+ \gamma^3 H_t H_{t-1} H_{t-2}(x_{t-2} \odot (1 - m_{t-1}) \odot (1 - m_t)) + \cdots \\ &+ \gamma^n H_t \cdots H_{t-(n-1)}(x_{t-(n-1)} \odot (1 - m_{t-(n-2)}) \odot \cdots \odot (1 - m_t)) \\ &+ \gamma^n H_t \cdots H_{t-(n-1)}(\widetilde{x}_{t-(n-1)} \odot (1 - m_{t-(n-1)}) \odot \cdots \odot (1 - m_t))\end{aligned} \tag{16}$$

The n steps of historical steps of states can be written in a summation form as

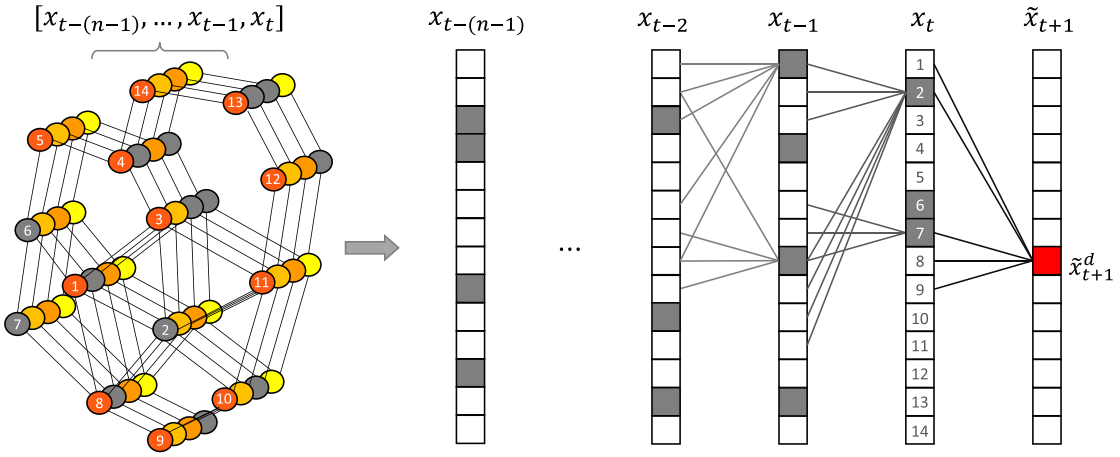


Fig. 1. Graph Markov process. The gray-colored nodes in the left sub-figure demonstrate the nodes with missing values. Vectors on the right side represent the traffic states. The traffic states at time t are numbered to match the graph and the vector. The future state (in red color) can be inferred from their neighbors in the previous steps. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

$$\begin{aligned}\widetilde{x}_{t+1} = &\sum_{i=0}^{n-1} \gamma^{i+1} \Big(\prod_{j=0}^{i} H_{t-j} \Big)\Big(x_{t-i} \odot \bigodot_{j=0}^{i-1} (1 - m_{t-j})\Big) \\ &+\gamma^n H_t \cdots H_{t-(n-1)} \big(\widetilde{x}_{t-(n-1)} \odot (1 - m_{t-(n-1)}) \odot \cdots \odot (1 - m_t)\big)\end{aligned} \tag{17}$$

where $\sum$, $\prod$, and $\odot$ are the summation, matrix product, and Hadamard product operators, respectively. For simplicity, we denote the term with the $\widetilde{x}_{t-n}$ in Eq. (17) as $O(\widetilde{x}_{t-n})$, and the GMP of the complected states can be represented by

$$\widetilde{x}_{t+1} = \sum_{i=0}^{n-1} \gamma^{i+1} \Big(\prod_{j=0}^{i} H_{t-j} \Big)\Big(x_{t-i} \odot \bigodot_{j=0}^{i-1} (1 - m_{t-j})\Big) + O(\widetilde{x}_{t-(n-1)}) \tag{18}$$

In $O(\widetilde{x}_{t-(n-1)})$, when $n \to \infty$, since $\gamma \in (0, 1)$, $\gamma^{n+1} \to 0$. In addition, the product of masking vectors in $O(\widetilde{x}_{t-(n-1)})$ will also approach to zero, i.e. $\odot_{j=0}^{n-1}(1 - m_{t-j}) \to 0$. The probability of each element of $\odot_{j=0}^{n-1}(1 - m_{t-j})$ being zero is related to the value of n and the traffic state values' missing rate (mr), which can be calculated as $1 - (1 - mr)^n$. The larger the n and the mr are, the elements of $\odot_{j=0}^{n-1}(1 - m_{t-j})$ have the higher probability to be zeros. When $mr = 20\%$ and $n = 10$, the probability is $1 - 0.8^{10} = 89.26\%$. Besides, the last term in Eq. (18) contains a γ^n, which will further degrade the contribution of the last term to the traffic forecasting task. Thus, when n is large enough, we consider the $O(\widetilde{x}_{t-(n-1)})$ is a negligibly term.

Fig. 1 demonstrates the graph Markov process for inferring the future state. The traffic network graphs with attribute-missed nodes (in gray color) is converted into traffic state vectors. The inference of $\widetilde{x}^d_{t+1}$ is based on historical traffic states by back-propagating to the $t - (n - 1)$ step.

4.3. Graph Markov network

In this section, we propose a **Graph Markov Network** (GMN) for traffic prediction with the capability of handling missing values in historical data. Suppose the historical traffic data consists of n steps of traffic states $\{x_{t-(n-1)}, ..., x_t\}$. Correspondingly, we can acquire n masking vectors $\{m_{t-(n-1)}, ..., m_t\}$. The traffic network's topological structure can be represented by the adjacency matrix.

The GMN is designed based on the proposed GMP described in the previous section. Since we consider the term $O(\widetilde{x}_{t-(n-1)})$ described in Eq. (18) is small enough, the $O(\widetilde{x}_{t-(n-1)})$ is omitted in the proposed GMN for simplicity.

As described in Eq. (18), the graph Markov process contains n transition weight matrices and the product of the these matrices $(\prod_{j=0}^{i} H_{t-j}) = (\prod_{j=0}^{i} \mathbf{A}^j \odot P_{t-j})$ measures the contribution of x_{t-i} for generating the $\widetilde{x}_t$. To reduce matrix product operations and at the same time keep the learning capability in the GMP, we simplify the $(\prod_{j=0}^{i} \mathbf{A}^j \odot P_{t-j})$ by $(\mathbf{A}^{i+1} \odot W_{i+1})$, where $W_{i+1} \in \mathbb{R}^{S \times S}$ is a weight matrix. In this way, $(\mathbf{A}^{i+1} \odot W_{i+1})$ can directly measure the contribution of x_{t-i} for generating the $\widetilde{x}_t$ and skip the intermediate state transition processes. Further, the GMP still has n weight matrices $(\{\mathbf{A}^1 \odot W_1, ..., \mathbf{A}^n \odot W_n\})$, and thus, the learning capability in terms of the size of parameters does not change. The benefits of the simplification is that the GMP can reduce $\frac{n(n-1)}{2}$ times of multiplication between two $S \times S$ matrices in total.

Based on the GMP and the aforementioned simplification, we propose the *graph Markov network* for traffic forecasting with the capability of handling missing values as

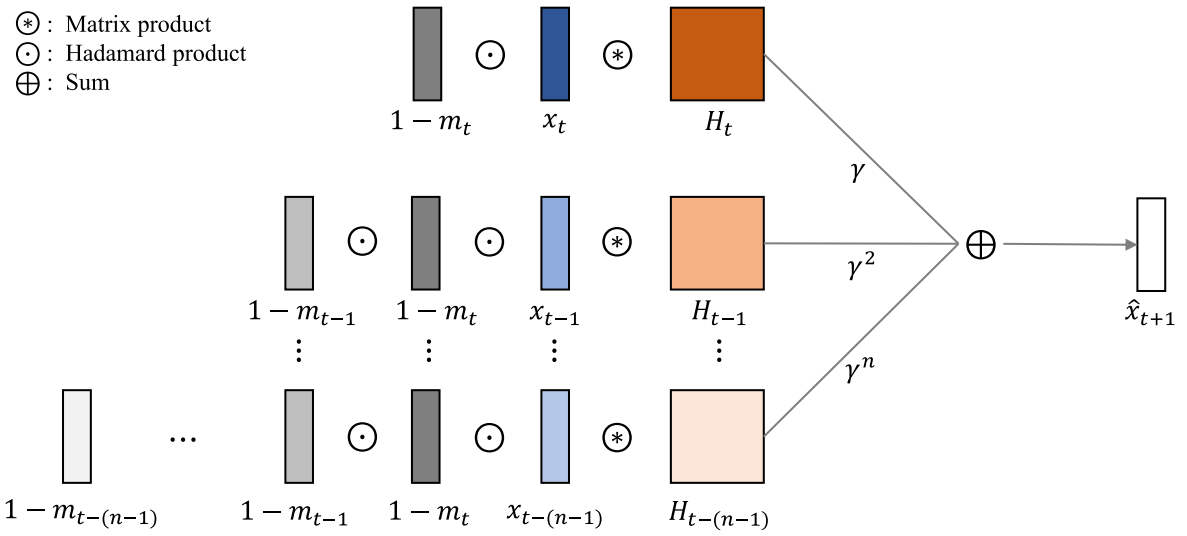


Fig. 2. Structure of the proposed graph Markov network. Here, $H_{t-j} = \mathbf{A}^j \odot W_j$. As for the spectral version of GMN, $H_{t-j} = U\Lambda_j U^T$.

$$\hat{x}_{t+1} = \sum_{i=0}^{n-1} \gamma^{i+1}(\mathbf{A}^{i+1} \odot W_{i+1})(x_{t-i} \odot \bigodot_{j=0}^{i-1} (1 - m_{t-j})) \tag{19}$$

where $\hat{x}_{t+1}$ is the predicted traffic state for the future time step $t + 1$ and $\{W_1, ..., W_n\}$ are the model's weight matrices that can be learned and updated during the training process.

As for the spectral version of the graph Markov process, the product of the transition weight matrices ($\prod_{j=0}^{i} H_{t-j}$) can also be simplified. Because of the orthogonality of the eigenvectors of $\mathcal{L}$ (Wang and Van Mieghem, 2015), $U^T = U^{-1}$, and thus, $(\prod_{j=0}^{i} H_{t-j}) = (\prod_{j=0}^{i} U\Lambda_{\theta_{t-j}}U^T) = U(\prod_{j=0}^{i} \Lambda_{\theta_{t-j}})U^T$. We further simplify the product of the transition weight matrices by replacing the $\prod_{j=0}^{i} \Lambda_{\theta_{t-j}}$ with a diagonal weight matrix Λ_{i+1}. Similar to the simplification of GMP, this simplification process will not reduce the learning capability of the SGMP because the amount of the weight parameters does not change. In this way, the SGMP can reduce at least $\frac{n(n-1)}{2}$ times of vector multiplication in total. Then, the spectral version of the graph Markov network (SGMN) can be defined as

$$\hat{x}_{t+1} = \sum_{i=0}^{n-1} \gamma^{i+1}(U\Lambda_{i+1}U^T)(x_{t-i} \odot \bigodot_{j=0}^{i-1} (1 - m_{t-j})) \tag{20}$$

where $\{\Lambda_1, ...,\Lambda_n\}$ are the diagonal weight matrices that can be learned and updated during the training process.

The structure of GMN for predicting traffic state x_{t+1} is demonstrated in Fig. 2. The spectral version of GMN has the same model structure that it only need to replace the $\mathbf{A}^j \odot W_j$ with the $U\Lambda_j U^T$, as shown Fig. 2. During the training process, the loss can be calculated by measuring the difference between the predicted value $\hat{y} = \hat{x}_{t+1}$ and the label $y = x_{t+1}$.

5. Experimental results

In this section, we compared the proposed approach with state-of-the-art traffic forecasting models with the capability of handling missing values. The graph Markov network predicts one time step ahead in this section, and if needed, two or more times ahead can also be predicted by using the last prediction as the input to the model. The time intervals depend on the datasets tested in the experiments. The datasets, hyper-parameters, software, and hardware used in the experiments are introduced in this section.

5.1. Datasets

In this study, we conduct experiments on three real-world network-wide traffic state datasets. The topological structures of the traffic networks are also used in the experiments.

5.1.1. PEMS-BAY

This dataset named as PEMS-BAY is collected by California Transportation Agencies (CalTrans) Performance Measurement System (PeMS). This dataset contains the speed information of 325 sensors in the Bay Area lasting for six months ranging from Jan 1st, 2017 to Jun 30th, 2017. The interval of time steps is 5-min. The total number of observed traffic data points is 16,941,600. The adjacency matrix of this dataset is defined according to Li et al. (2018). The dataset is published by Li et al. (2018) on the Github (https://github.com/liyaguang/DCRNN).

5.1.2. METR-LA

This dataset is collected from loop detectors on the freeway of Los Angeles County (Jagadish et al., 2014). This dataset contains the speed information of 207 sensors lasting for 4 months ranging from Mar 1st, 2012 to Jun 30th, 2012. The interval of time steps is 5-min. The total number of observed traffic data points is 6,519,002. Similar to the PEMS-BAY dataset, the adjacency matrix of this dataset is defined according to Li et al. (2018), and the dataset is published on the Github (https://github.com/liyaguang/DCRNN).

5.1.3. INRIX-SEA

This dataset is collected by the INRIX company from multiple data sources, including GPS probes, road sensors, and cell phone data. This dataset contains the speed information of the traffic network in the Seattle downtown area consisting of 725 road segments. The traffic network covers both freeways and urban roadways. The dataset covers a one-year period from Jan 1st, 2012 to Dec 31st, 2012. The interval of time steps is 5-min. The total number of observed traffic data points is 76,212,000. This dataset is provided by Washington Department of Transportation and has been used in Cui et al. (2019). Due to privacy policies, this dataset is not published.

5.2. Missing values and data formatting

The dataset forms as a spatial–temporal matrix, whose spatial dimension size is the number of sensors and temporal dimension size is the number of time steps. In the experiments, the dataset is split into a training set, a validation set, and a testing set, with a size ratio of 6:2:2. In the training and testing process, the speed values of the dataset are normalized within a range of [0,1].

The PEMS-BAY and METR-LA datasets originally have missing values and their percentages of missing values are 0.003% and 8.11%, respectively. It should be noted that the original PEMS-BAY dataset should have already imputed the missing values linearly (Chen et al., 2002). But the PEMS-BAY dataset is acquired from the online link mentioned in Section 5.1.1. There are no missing values in the INRIX-SEA dataset. To test the model's capability of handling missing values with different missing rates, we randomly set a portion of speed values in the input sequences as zeros according to a specific missing rate and generate the masking vectors accordingly. In this study, based on each of the three datasets, three sub-datasets with artificial missing rates of 10%, 20%, and 40%, respectively, are generated. In METR-LA datasets, the original missing values are integrated with the artificial missing values. It also should be noted that, besides random missing, other missing patterns, such as consecutive missing values, are also common in the traffic domain (Boquet et al., 2019; Laña et al., 2018; Gondara and Wang, 2017). Since the proposed models dealing with the traffic network as a graph, they are more suitable for processing data with randomly missing values. Other patterns will be further studied in the future work.

5.3. Hardware and software environments

The experiments are conducted on a computer with 128 GB memory, a Intel Core i9-7900X CPU, and a NVIDIA GTX 1080 Ti GPU. The proposed approach and all neural network-based baseline models are implemented based on PyTorch 1.0.1 using the Python language 3.6.8.

5.4. Baseline models

- GRU (Cho et al., 2014): GRU referring to gated recurrent units is a type of RNN. GRU can be considered as a simplified LSTM.
- GRU-I: GRU-I is designed based on GRU. Since GRU is a type of a RNN with the recurrent structure, the predicted values from a previous step $\hat{x}_t$ can be used to infer the missing values in the next step. The completed traffic states with all missing values filled can be represented by $\widetilde{x}_{t+1} = x_t + \hat{x}_t \odot (1 - m_t)$.
- GRU-D (Che et al., 2018): GRU-D is a neural network structure that is designed based on GRU for multivariate time series prediction. It can capture long-term temporal dependencies in time series. GRU-D incorporates the masking information and missing values' time interval as input such that it can utilize the missing patterns.
- LSTM (Hochreiter and Schmidhuber, 1997): LSTM is a powerful variant of RNN, which can overcome the gradients exploding or vanishing problem. It is suitable for being a model's basic structure for traffic forecasting.
- LSTM-I: LSTM-I is designed based on LSTM. The missing value inferring process of LSTM-I is similar to that of GRU-I.
- LSTM-M (Tian et al., 2018): LSTM-M is a neural network structure designed based on LSTM for traffic prediction with missing data. It employs multi-scale temporal smoothing methods to infer lost data.

5.5. Model parameters

The batch size of the training samples is set as 64. The number of steps of historical data incorporated in the GMN model will have an influence on the prediction performance. Hence, the GMNs with 6-steps, 8-steps, and 10-steps of historical data are tested in the experiments, i.e. the n in Eqs. (19) and (20) are set as 6, 8, and 10. It should be noted that the value of n is not fixed that the larger the n is, the better performance the GMN model can achieve. In the following sections, we denoted these GMN models as GMN-6, GMN-8, and GMN-10, respectively. The corresponding SGMN models with different steps are denoted as SGMN-6, SGMN-8,and SGMN-10, respectively. As analyzed in Section 5.8.3, a decay rate ranging from 0.6 to 0.9 helps the model generate better performance. Thus, the decay parameter γ is set as 0.9 in the experiments. For the RNN-based baseline models, including GRU, GRU-I, GRU-D, LSTM,

LSTM-I, and LSTM-M, their input sequences all have 10 time steps. The numbers of parameters of GRU-based and LSTM-based models are about $3S^2$ and $4S^2$, respectively, where S is the spatial dimension of the input data. The number of parameters of GMN and SGMN are nS^2 and nS, respectively.

5.6. Training and hyper-parameters

In the training process, the mean square error (MSE) between the label y_t and the predicted value $\hat{y}_t$, i.e. $\frac{1}{n}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$ is used as the loss function, where N is the sample size. The Adam (Kingma and Ba, 2014) optimization method is adopted for both GMN models and baseline models to update parameters, as Adam is also used in Che et al. (2018) and Tian et al. (2018). Early stopping mechanism is used to avoid over-fitting. If there is no improvement on the MSE of the validation set in 5 consecutive epochs, the training will be stopped. The minimum improvement in MSE is set as 0.00001. We also design a learning rate decay mechanism for the training process to speed up the models' convergence. The initial learning rate of all models is set as 10^{-3}, which is identical to the learning rate in Tian et al. (2018). If there is no improvement in 4 consecutive epochs, the learning rate is reduced an order of magnitude until it reaches 10^{-5}.

5.7. Evaluation metric

The prediction accuracy of all tested models are evaluated by three metrics, including mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean square error (RMSE).

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \tag{21}$$

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{22}$$

$$RMSE = \left(\frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|^2\right)^{\frac{1}{2}} \tag{23}$$

5.8. Experimental results

5.8.1. Comparing with baseline models

The prediction results tested on the PEMS-BAY, METR-LA, and INRIX-SEA datasets with respect to different missing rates are displayed in Tables 1–3, respectively. All results presented these three tables were averaged with different runs of the experiment. Overall, the SGMN models are superior to other baseline models. The GMN models also perform well, especially on the PEMS-BAY dataset. However, the prediction performance of GMN models decreases faster than that of SGMN models along with the increase of the missing rate. Among the baseline models, the GRU-I model performs well that it achieve smaller RMSEs on the PEMS-BAY and INRIX-SEA datasets.

As shown in Table 1, the SGMN-10 achieves the smallest MAEs and MAPEs for all the three missing rates on the PEMS-BAY dataset. However, the RMSEs of GRU-I are the smallest ones for all missing rates. The test results on the INRIX-SEA dataset, as shown

Table 1
Prediction performance on PEMS-BAY dataset.

	PEMS-BAY								
Model	Missing Rate = 10%			Missing Rate = 20%			Missing Rate = 40%		
	MAE (mph)	MAPE(%)	RMSE (mph)	MAE (mph)	MAPE(%)	RMSE (mph)	MAE (mph)	MAPE(%)	RMSE (mph)
GRU	1.608	3.133	2.608	1.787	3.522	2.911	2.052	4.095	3.320
GRU-I	1.108	2.133	**1.831**	1.185	2.296	**1.968**	1.385	2.729	**2.327**
GRU-D	5.320	13.584	9.163	5.347	13.609	9.160	5.387	13.67	9.180
LSTM	2.368	4.809	3.952	2.457	5.098	4.258	2.428	5.117	4.181
LSTM-I	2.218	4.001	7.472	2.373	4.278	7.742	2.058	3.989	5.863
LSTM-M	1.198	2.351	1.968	1.236	2.437	2.055	1.472	2.904	3.111
GMN-6	1.084	2.077	2.565	1.101	2.145	2.029	1.819	3.634	3.543
GMN-8	1.089	2.086	2.611	1.196	2.297	2.827	1.376	2.730	2.678
GMN-10	1.089	2.086	2.614	1.202	2.308	2.864	1.327	2.615	2.470
SGMN-6	1.009	1.930	1.877	1.064	2.048	2.067	1.930	3.671	4.375
SGMN-8	1.008	1.929	1.875	1.062	2.043	2.024	1.291	2.517	2.867
SGMN-10	**1.007**	**1.927**	1.874	**1.058**	**2.037**	2.018	**1.207**	**2.362**	2.473

The bold values represent the best prediction results for each metric.

Table 2
Prediction performance on METR-LA dataset.

Model	METR-LA								
	Missing Rate = 10%			Missing Rate = 20%			Missing Rate = 40%		
	MAE (mph)	MAPE(%)	RMSE (mph)	MAE (mph)	MAPE(%)	RMSE (mph)	MAE (mph)	MAPE(%)	RMSE (mph)
GRU	3.427	7.971	5.923	3.667	8.611	6.249	4.037	9.622	6.744
GRU-I	3.322	7.625	5.543	3.402	7.846	5.642	**3.389**	7.917	5.903
GRU-D	9.912	25.28	12.195	9.904	25.302	12.193	10.022	25.444	12.269
LSTM	3.477	8.050	6.015	3.652	8.559	6.263	3.899	9.300	6.663
LSTM-I	3.180	7.228	5.363	**3.267**	7.417	5.653	3.393	**7.826**	5.879
LSTM-M	3.253	7.374	5.540	3.368	7.666	5.717	3.410	7.837	**5.812**
GMN-6	3.384	7.300	5.624	3.477	7.488	5.583	3.913	8.518	6.362
GMN-8	3.565	7.684	6.126	3.653	7.852	6.001	3.864	8.365	6.083
GMN-10	3.708	7.969	6.512	3.792	8.131	6.411	3.961	8.518	6.216
SGMN-6	**3.145**	**6.836**	5.331	3.333	7.232	5.578	3.952	8.593	6.894
SGMN-8	3.174	6.889	5.362	3.318	7.203	5.552	3.699	8.053	6.186
SGMN-10	3.152	6.852	**5.321**	3.310	**7.187**	**5.525**	3.680	8.005	6.079

The bold values represent the best prediction results for each metric.

Table 3
Prediction performance on INRIX-SEA dataset

Model	INRIX-SEA								
	Missing Rate = 10%			Missing Rate = 20%			Missing Rate = 40%		
	MAE (mph)	MAPE(%)	RMSE (mph)	MAE (mph)	MAPE(%)	RMSE (mph)	MAE (mph)	MAPE(%)	RMSE (mph)
GRU	1.097	3.964	2.158	1.146	4.143	2.257	1.256	4.530	2.443
GRU-I	0.888	3.220	**1.850**	0.939	3.419	**1.920**	1.057	3.889	**2.086**
GRU-D	3.039	11.597	5.408	2.947	11.399	5.160	2.873	11.136	5.008
LSTM	1.256	4.451	2.446	1.450	5.364	2.956	1.433	5.260	2.902
LSTM-I	0.945	3.363	2.400	0.910	3.255	2.094	1.592	5.155	5.156
LSTM-M	1.096	4.357	2.633	1.001	3.787	2.264	0.986	3.584	2.098
GMN-6	2.354	8.541	4.832	2.704	9.588	5.545	3.063	10.700	5.960
GMN-8	2.356	8.547	4.835	2.712	9.613	5.559	2.938	10.277	5.803
GMN-10	2.355	8.545	4.835	2.713	9.618	5.561	2.923	10.224	5.778
SGMN-6	**0.768**	2.715	1.922	0.829	2.940	2.038	1.355	4.949	2.983
SGMN-8	**0.768**	**2.713**	1.921	**0.826**	**2.929**	2.026	1.024	3.679	2.399
SGMN-10	**0.768**	2.716	1.921	0.827	2.934	2.026	**0.973**	**3.485**	2.283

The bold values represent the best prediction results for each metric.

in Table 3, have the similar situation that SGMN models perform better in terms of the MAE and MAPE metrics and GRU-I achieves better RMSE results. Since RMSE takes the square root of the average squared errors, it gives a relatively high weight to large errors. The smaller RMSEs of GRU-I indicate that GRU-I's prediction results tend to have less large errors. It may seem contradictory that GRU-I outperforms other models on RMSE, and the proposed model achieves best prediction performance on MAE and MAPE. By checking the prediction results, we found this is resulted by the different residual distributions of GRU-I and SGMN. Thus, although GRU-I performs better on several cases, prediction performance of SGMN at least achieves the same level of that of stat-of-the-art models.

Since traffic states in INRIX-SEA are influenced by traffic lights, the INRIX-SEA dataset is more complicated than others. Thus, SGMN containing more nonlinear functions works better than GMN for capturing complicated patterns.As for the results tested on the METR-LA dataset, shown in Table 2, the SGMN models outperform other models when the missing rates are 10% and 20%. When the missing rate increases to 40%, the GRU-I, LSTM-I, and LSTM-M models achieve better prediction performance in terms of all the three metrics. This phenomenon shows the proposed model is good at processing data with small missing rates. From the three tables, it can be observed that the time horizon n, ranging from 6 to 10, of the proposed model influences the performance. The model performs better when n is larger. This is more obvious when the missing rate is greater.

5.8.2. Analysis on training time

In this section, we analyze the training time of the proposed models and baseline models. Fig. 3 shows the training time per epoch of the compared models tested on the PEMS-BAY datasets. The training times tested on different datasets have the same patterns. Since the GMN models have less matrix product operations than the SGMN models, GMN models take slightly less time per epoch than other models. The GMN and SGMN models apparently cost less running time than the baseline models because they get rid of the recurrent structure. The training times of GMN and SGMN increase when they incorporate more historical steps. The GRU and LSTM

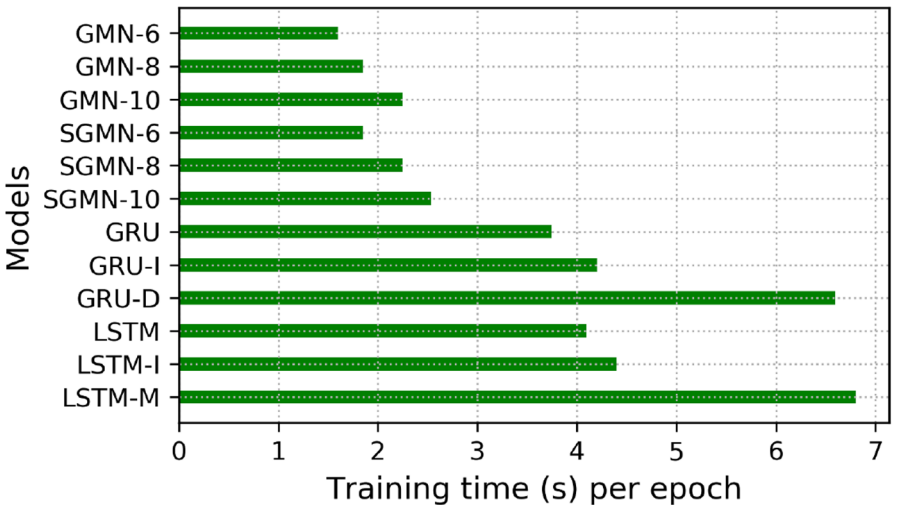


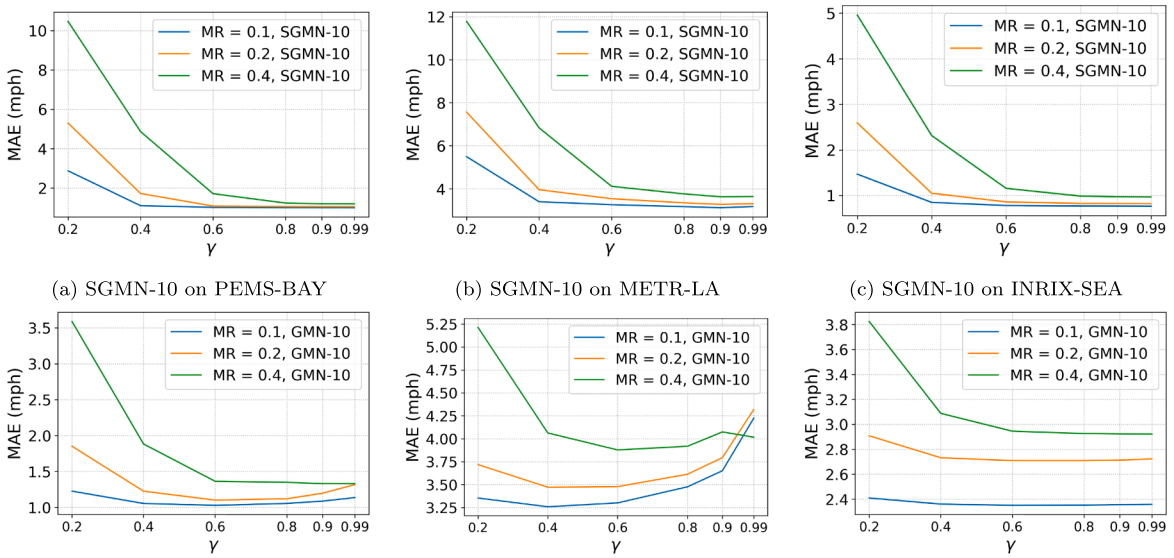
Fig. 3. Training time of the compared models.

have similar training time per epoch. Since the GRU-I and LSTM-I have an imputation operation, their training times take a little bit more. In addition, since the GRU-D and LSTM-M both take more types of data as the input, their training time is much more than GRU and LSTM.

5.8.3. Analysis on decay rates of GMN and SGMN

The proposed graph Markov process adopts the decay rate $\gamma \in (0, 1)$ to represent the temporal impact of Δt on the traffic state transition process. In previous analysis sections, the Δt is 5-min, and the decay rates of GMN models are set as 0.9. In this section, we analyze the influence of the decay rate on the proposed models' prediction performance. The prediction performance of SGMN-10 and GMN-10 w.r.t. various decay rates are shown in Fig. 4. The models are tested on the three datasets with different missing rates. Generally, the missing rate affects the prediction performance a lot that large missing rate results in large prediction errors.

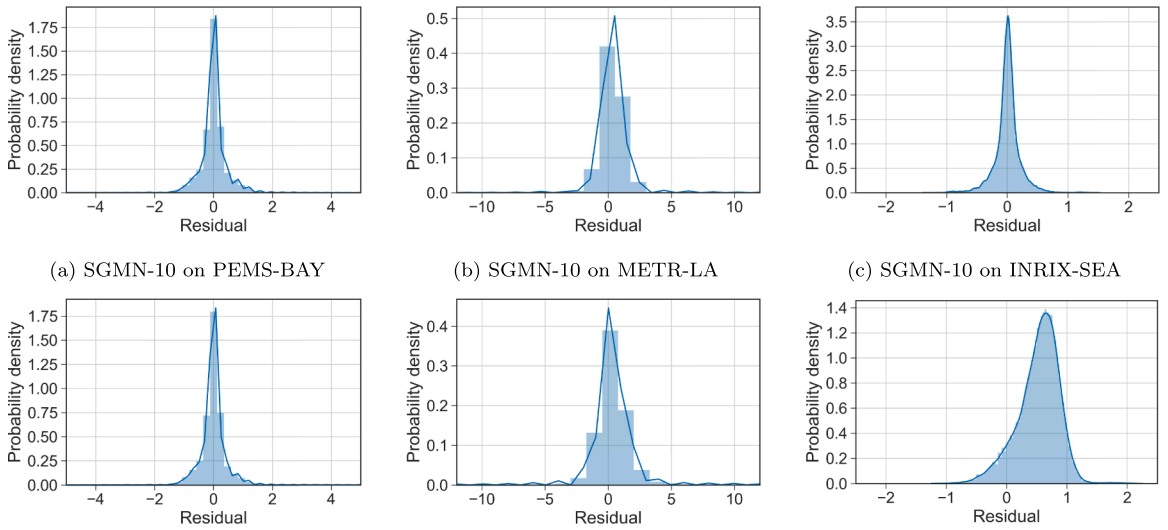
The six sub-figures in Fig. 4 all indicate the prediction errors (MAE) decrease along with the increase of γ. The prediction results of the SGMN-10 models tested on the three datasets have the similar curve patterns, as shown by the line-charts in Fig. 4a, b, and c. When γ is close to zero, the prediction errors are relatively large. When γ is increasing, the prediction errors seem to be monotonically decreasing. When γ is close to one, the curves are almost flat and prediction errors nearly keep the same. However, as shown in Fig. 4b, the MAE tested on the METR-LA dataset increases a little bit when γ increases from 0.9 to 0.99.



(a) SGMN-10 on PEMS-BAY (b) SGMN-10 on METR-LA (c) SGMN-10 on INRIX-SEA

(d) GMN-10 on PEMS-BAY (e) GMN-10 on METR-LA (f) GMN-10 on INRIX-SEA

Fig. 4. Prediction performance metric (MAE) w.r.t. the decay rate γ. The SGMN-10 and GMN-10 are tested on the PEMS-BAY, METR-LA, and INRIX-SEA datasets with different missing rates.



(a) SGMN-10 on PEMS-BAY (b) SGMN-10 on METR-LA (c) SGMN-10 on INRIX-SEA

(d) GMN-10 on PEMS-BAY (e) GMN-10 on METR-LA (f) GMN-10 on INRIX-SEA

Fig. 5. Prediction residuals of the proposed models tested on three datasets when the missing rate is 20%.

The MAEs of the GMN-10 models shown in Fig. 4d, e, and f have slightly different patterns than those of SGMN-10 models. The MAE curves are not monotonically decreasing. When γ is close to one, the prediction errors start to increase. This phenomenon is particularly obvious in the results tested on the METR-LA dataset.

In addition, it should be noted that the y-axes of those sub-figures have various ranges. When the decay rate is relatively small (close to one), the prediction capability of GMN models is better than that of SGMN models. One possible reason is that GMN models contains more weight parameters than SGMN models. In summary, the selection of decay rate, which can be considered as a hyper-parameter and need to be tuned, depends on the dataset, the model, and even other hyper-parameters.

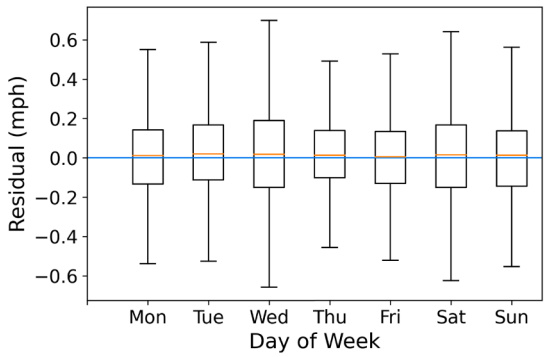
5.8.4. *Analysis on forecasting residuals*

Since residual is an critical indicator for evaluating whether a model is systematically correct, the residuals of predictions are analyzed in this section. The residual equals the ground truth value subtracts the predicted value, i.e. $y - \hat{y}$. Fig. 5 shows the residual distributions of SGMN-10 and GMN-10 tested on the three datasets. Most of the sub-figures display that the residual distributions follow normal distributions with zero means, except for the result of GMN-10 tested on the INRIX-SEA dataset. Although the proposed models are more complex than regression models, the residuals' normal distributions indicate that the proposed models have sufficient prediction capabilities and capture enough predictive information from the input data.

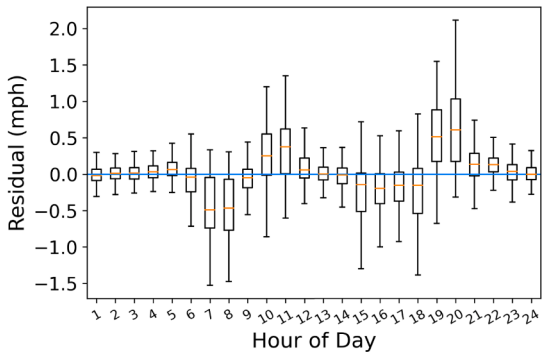
The prediction performance will also be influenced by the temporal information, such as hour of day and day of the week. Normally, during peak hours, traffic states with more variations are harder to be predicted. Thus, in this section, the influence of hour of day and day of the week is measured. The residuals of SGMN-10 tested on the three datasets with respect to day of the week and hour of day are shown in Fig. 6. As displayed by the box-plots in Fig. 6a, c, e, the prediction residuals on each day of the week are almost the same. That means the proposed models has the capability of forecasting traffic states on each day of the week. The residuals with respect to hour of day are displayed in Fig. 6b, d, f. The influence of peak hours on traffic forecasting is particularly obvious on the PEMS-BAY dataset. However, the residual distributions in each hour of the day on the METR-LA dataset do not have much differences. The residual distributions on the INRIX-SEA dataset are abnormal to some extent that the residuals are large during the afternoon and midnight. This phenomenon may be lead by the various traffic patterns of different types of roadways in different cities.

5.8.5. *Model weight analysis and visualization*

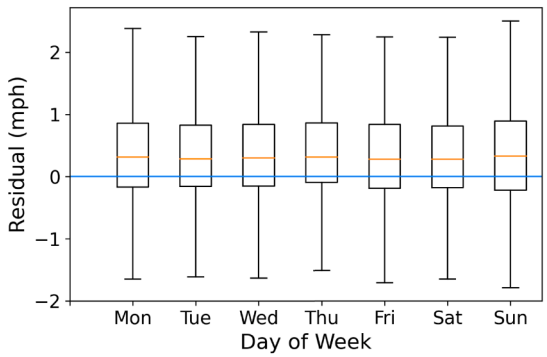
In this section, the proposed model's weights are analyzed and visualized. We take the SGMN-10 and GMN-10 trained on METR-LA dataset as an example. Fig. 7a shows the 207 sensor locations in the METR-LA dataset denoted by blue dots, and Fig. 7b shows the top 20 most influential sensor locations in terms of the influence on forecasting traffic states of the future $(t + 1)$ step from the states of the current (t) step. The influence of a sensor of the k-th location is reflected by the sum/average of the squared element values in the k-th row/column of the model's weight matrix at the t step, i.e. the H_t described in Eq. (18). For example, the averaged squared element values of the k-th row of H_t is calculated as $\frac{1}{n}\sum_{i=0}^{n-1}(H_t)_{k,i}^2$. Here, $H_t = \mathbf{A}^1 \odot W_1$ for the GMN case, and for the SGMN case, $H_t = U\Lambda_1 U^T$. As depicted in the map in Fig. 7b, the selected top 20 influential sensor locations are mostly distributed near intersection areas, which has great potential to affect nearby traffic states. Fig. 7c and d display the influence of the 89-th sensor location on its neighboring sensor locations. This sensor location with the sensor ID 767351 is represented by an orange dot on the maps. The influence is reflected by the element values of the model's weight matrix at the 89-th row/column. The positive and negative weight



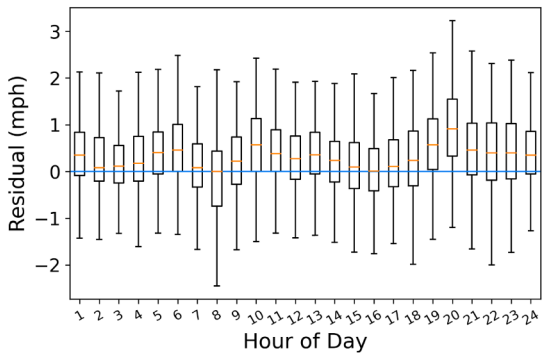
(a) PEMS-BAY



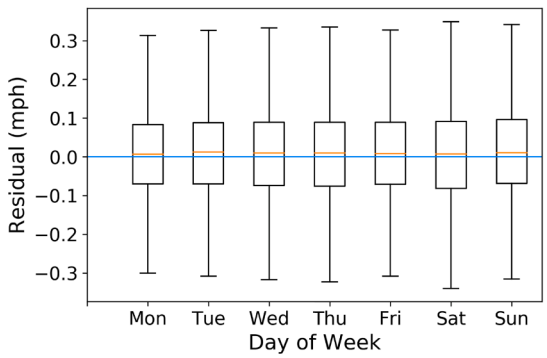
(b) PEMS-BAY



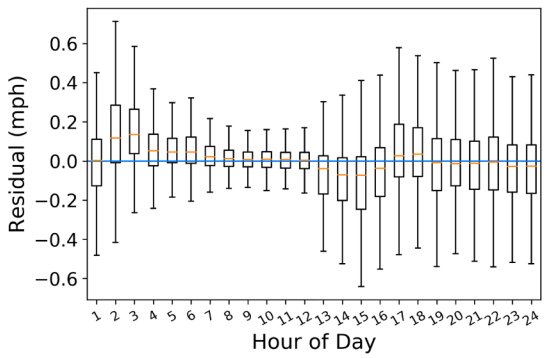
(c) METR-LA



(d) METR-LA



(e) INRIX-SEA



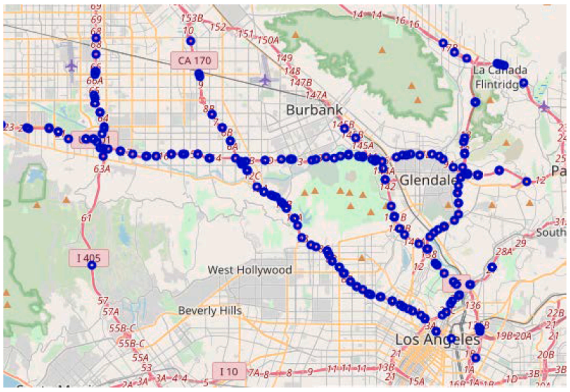
(f) INRIX-SEA

Fig. 6. Prediction residuals of SGMN-10 with respect to day of the week and hour of day, tested on three datasets with the missing rate of 20%.

values of other sensor locations are demonstrated by blue and pink colors, respectively. The darker the color is, the larger the absolute value of the weight element is. The difference between these two figures is that the illustrated neighboring locations in Fig. 7d are confined within a small one-hop neighboring area by the weight matrix of GMN-10, i.e. the $\mathbf{A}^1 \odot W_1$. However, as shown by the two figures, the surrounding sensor locations with respect to the 89-th sensor location (the orange dot) are obviously darker, which means the traffic state of a location is influenced more by the states of its neighbors. Thus, by quantitatively analyzing or visualizing the weight matrices of the proposed models, the influence of nodes/locations in a traffic network on their neighbors can be measured.

5.8.6. Traffic forecasting result visualization

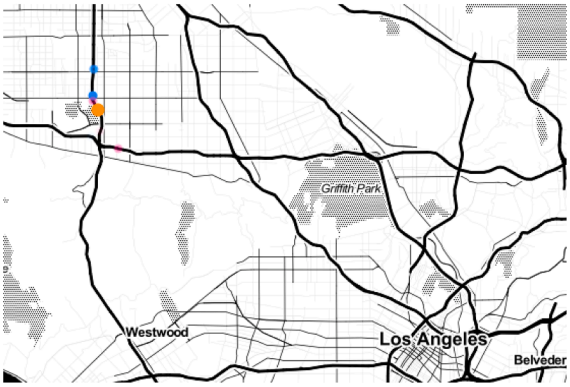
The locations covered by those datasets actually have various traffic patterns. In this section, to demonstrate the proposed model's prediction performance, we select several sensor locations/links from the three datasets and visualize the ground truth and predicted speed values. The three sub-figures in Fig. 8 display the ground truth and speed values predicted by the GMN-10 and SGMN-10. The missing rates of the tested datasets are all set as 20%. Both GMN-10 and SGMN-10 work well on the PEMS-BAY dataset. Since the METR-LA dataset originally has missing values, there are some blue spikes reaching the bottom of Fig. 8b demonstrating the original



(a) All sensor locations in the METR-LA dataset

(b) Top 20 influential sensor locations

(c) Weight $(U\Lambda_1 U^T)$ visualization of SGMN-10 w.r.t. the 89^{th} sensor location, denoted by the orange dot



(d) Weight $(\mathbf{A}^1 \odot W_1)$ visualization of GMN-10 w.r.t. the 89^{th} sensor location, denoted by the orange dot

Fig. 7. Visualization of sensor locations and models weights. The top 20 influential sensor locations in (b) are ones with the top 20 largest row-wise averaged squared element values of the weight matrix $H_1 = U\Lambda_1 U^T$ in SGMN-10, which is introduced in Section 5.8.5. The blue and pink dots in (c) and (d) represent positive and negative weight values, respectively. The darker the color is, the larger the absolute value of the weight is. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

missing values. The prediction performance of SGMN-10 on the INRIX-SEA dataset is better than that of GMN-10. Overall, the proposed models have the capability of forecasting traffic states with missing values.

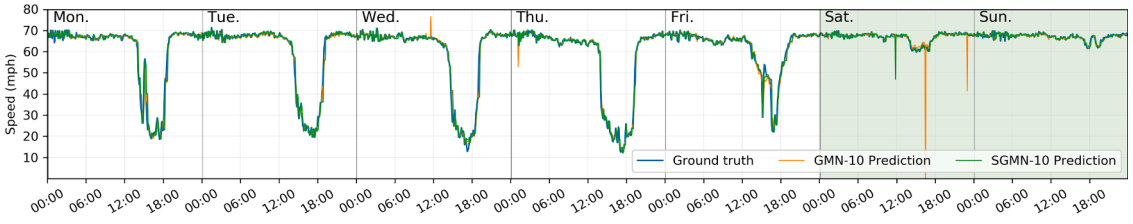
6. Conclusion

In this study, we propose the GMN, which is a new neural network architecture for spatial–temporal data forecasting. We introduce two properties of the traffic state transition process and define a graph Markov process. Unlike other existing recurrent neural network (RNN)-based models dealing with traffic data as multivariate time series, the GMN handling the traffic state transition process as a graph Markov process. The proposed GMN can incorporate the spatial relationship between neighboring links and the links' temporal dependencies between different time steps. By incorporating the spectral graph convolution operation, we also propose a spectral graph Markov network (SGMN). The experimental results tested on a real-world dataset shows show that the proposed GMN and SGMN achieves superior prediction performance. Further, the proposed models' parameters, weights, and prediction residuals are discussed and visualized.

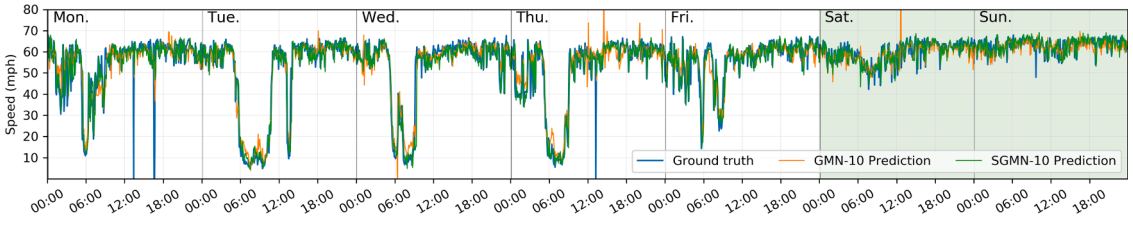
The future work will focus on enhancing the theoretical basis of the proposed graph Markov process. We will attempt to build a connection between the graph Markov process with the Markov random field to analyze the hidden factors influencing traffic states. In addition, we will conduct more experiments on multiple public accessible datasets.

CRediT authorship contribution statement

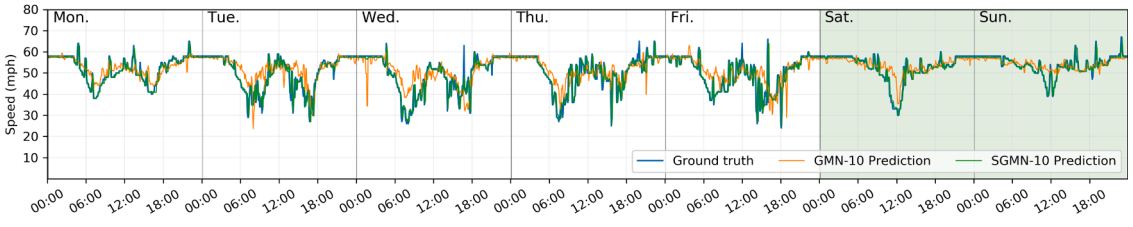
Zhiyong Cui: Conceptualization, Data curation, Methodology, Software, Visualization, Writing - original draft, Writing - review & editing. **Longfei Lin:** Investigation, Writing - review & editing, Validation. **Ziyuan Pu:** Investigation, Writing - review & editing, Validation. **Yinhai Wang:** Supervision, Investigation, Validation.



(a) Results tested on the PEMS-BAY dataset from Jan. 30, 2017 to Feb. 5, 2017. Sensor ID = 400097



(b) Results tested on the METR-LA dataset from Mar. 12, 2012 to Mar. 18, 2012. Sensor ID = 765273



(c) Results tested on the INRIX-SEA dataset from Jan. 1, 2012 to Jan. 7, 2012. link ID = 114+04549

Fig. 8. Comparison of the ground truth and the speed predicted by GMN-10 and SGMN-10 tested on three datasets with the missing rate of 20% under the random missing scenario. The white and green regions in these figures demonstrate weekdays and weekends, respectively. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Acknowledgments

This work was supported by the Connected Cities with Smart Transportation (C2SMART) Tier 1 University Transportation Center with the USDOT Award No.: 69A3551747124. Thanks to Washington State Department of Transportation (WSDOT) for providing the research datasets. Thanks to Xinyu Chen for sharing the academic-drawing code on GitHub. Also, the authors would like to thank Ruimin Ke and Shuyi Yin for helpful discussions and comments.

References

Barredo-Arrieta, A., Laña, I., Del Ser, J., 2019. What lies beneath: A note on the explainability of black-box machine learning models for road traffic forecasting. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC), IEEE, pp. 2232–2237.

Boquet, G., Vicario, J.L., Morell, A., Serrano, J., 2019. Missing data in traffic estimation: A variational autoencoder imputation method. In: ICASSP 2019–2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, pp. 2882–2886.

Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y., 2018. Recurrent neural networks for multivariate time series with missing values. Sci. Rep. 8 (1), 6085.

Chen, C., Kwon, J., Varaiya, P., 2002. The quality of loop data and the health of california's freeway loop detectors. PeMS Develop. Group 5–6.

Chen, X., He, Z., Wang, J., 2018. Spatial-temporal traffic speed patterns discovery and incomplete data recovery via svd-combined tensor decomposition. Transp. Res. Part C: Emerg. Technol. 86, 59–77.

Chen, X., He, Z., Sun, L., 2019. A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation. Transp. Res. Part C: Emerg. Technol. 98, 73–84. https://doi.org/10.1016/j.trc.2018.11.003.. https://www.sciencedirect.com/science/article/pii/S0968090X1830799X.

Chen, X., He, Z., Sun, L., 2019. A bayesian tensor decomposition approach for spatiotemporal traffic data imputation. Transp. Res. Part C: Emerg. Technol. 98, 73–84.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y., 2014. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1724–1734.

Cui, Z., Ke, R., Wang, Y., et al., 2016. Deep stacked bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction. In: 6th International Workshop on Urban Computing (UrbComp 2017).

Cui, Z., Henrickson, K., Ke, R., Wang, Y., 2019.. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. IEEE Trans. Intell. Transp. Syst.

Defferrard, M., Bresson, X., Vandergheynst, P., 2016. Convolutional neural networks on graphs with fast localized spectral filtering. Adv. Neural Inform. Process. Syst. 3844–3852.

Duan, Y., Lv, Y., Liu, Y.-L., Wang, F.-Y., 2016. An efficient realization of deep learning for traffic data imputation. Transp. Res. Part C: Emerg. Technol. 72, 168–181.

Du, M., Liu, N., Yang, F., Ji, S., Hu, X., 2019. On attribution of recurrent neural network predictions via additive decomposition. In: The World Wide Web Conference, ACM, pp. 383–393.

Ghosh, B., Basu, B., O'Mahony, M., 2009. Multivariate short-term traffic flow forecasting using time-series analysis. IEEE Trans. Intell. Transp. Syst. 10 (2), 246.

Gondara, L., Wang, K., 2017. Multiple imputation using deep denoising autoencoders, arXiv preprint arXiv: 1705.02737.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.
Jagadish, H., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., Shahabi, C., 2014. Big data and its technical challenges. Commun. ACM 57 (7), 86–94.
Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization, arXiv preprint arXiv: 1412.6980.
Ku, W.C., Jagadeesh, G.R., Prakash, A., Srikanthan, T., 2016. A clustering-based approach for data-driven imputation of missing traffic data. In: 2016 IEEE Forum on Integrated and Sustainable Transportation Systems (FISTS). IEEE, pp. 1–6.
Laña, I., Olabarrieta, I.I., Vélez, M., Del Ser, J., 2018. On the imputation of missing data for road traffic forecasting: New insights and novel techniques. Transp. Res. Part C: Emerg. Technol. 90, 18–33.
Lee, S., Fambro, D.B., 1999. Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting. Transp. Res. Rec. 1678 (1), 179–188.
Liang, Y., Cui, Z., Tian, Y., Chen, H., Wang, Y., 2018. A deep generative adversarial architecture for network-wide spatial-temporal traffic-state estimation. Transp. Res. Rec. 2672 (45), 87–105.
Li, L., Su, X., Zhang, Y., Hu, J., Li, Z., 2014. Traffic prediction, data compression, abnormal data detection and missing data imputation: An integrated study based on the decomposition of traffic time series. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 282–289.
Li, Y., Yu, R., Shahabi, C., Liu, Y., 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, arXiv preprint arXiv: 1707.01926.
Li, Y., Yu, R., Shahabi, C., Liu, Y., 2018. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In: International Conference on Learning Representations (ICLR '18).
Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y., 2015. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transp. Res. Part C: Emerg. Technol. 54, 187–197.
Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., Wang, Y., 2017. Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. Sensors 17 (4), 818.
Ma, X., Li, Y., Cui, Z., Wang, Y., 2018. Forecasting transportation network speed using deep capsule networks with nested lstm models. IEEE Trans. Intell. Transp. Syst.
Okutani, I., Stephanedes, Y.J., 1984. Dynamic prediction of traffic volume through kalman filtering theory. Transp. Res. Part B: Methodological 18 (1), 1–11.
Ran, B., Tan, H., Wu, Y., Jin, P.J., 2016. Tensor based missing traffic data completion with spatial–temporal correlation. Physica A 446, 54–63.
Rue, H., Held, L., 2005. Gaussian Markov Random Fields: Theory and Applications. Chapman and Hall/CRC.
Smith, B.L., Williams, B.M., Oswald, R.K., 2002. Comparison of parametric and nonparametric models for traffic flow forecasting. Transp. Res. Part C: Emerg. Technol. 10 (4), 303–321.
Sun, S., Zhang, C., Yu, G., 2006. A bayesian network approach to traffic flow forecasting. IEEE Trans. Intell. Transp. Syst. 7 (1), 124–132.
Tan, H., Wu, Y., Shen, B., Jin, P.J., Ran, B., 2016. Short-term traffic prediction based on dynamic tensor completion. IEEE Trans. Intell. Transp. Syst. 17 (8), 2123–2133.
Tang, J., Zhang, G., Wang, Y., Wang, H., Liu, F., 2015. A hybrid approach to integrate fuzzy c-means based imputation method with genetic algorithm for missing traffic volume data estimation. Transp. Res. Part C: Emerg. Technol. 51, 29–40.
Tian, Y., Zhang, K., Li, J., Lin, X., Yang, B., 2018. Lstm-based traffic flow prediction with missing data. Neurocomputing 318, 297–305.
Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C., 2014. Short-term traffic forecasting: Where we are and where we're going. Transp. Res. Part C: Emerg. Technol. 43, 3–19.
Wang, X., Van Mieghem, P., 2015. Orthogonal eigenvector matrix of the laplacian. In: 2015 11th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS). IEEE, pp. 358–365.
Williams, B.M., 2001. Multivariate vehicular traffic flow prediction: evaluation of arimax modeling. Transp. Res. Rec. 1776 (1), 194–200.
Wu, C.-H., Ho, J.-M., Lee, D.-T., 2004. Travel-time prediction with support vector regression. IEEE Trans. Intell. Transp. Syst. 5 (4), 276–281.
Yang, H., Liu, C., Gottsacker, C., Ban, X., Zhang, C., Wang, Y., 2019. Cell-speed prediction neural network (cpnn): A deep learning approach for trip-based speed prediction, Tech. rep..
Yoon, J., Jordon, J., Schaar, M., 2018. Gain: Missing data imputation using generative adversarial nets. In: International Conference on Machine Learning, pp. 5675–5684.
Yu, H., Wu, Z., Wang, S., Wang, Y., Ma, X., 2017. Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. Sensors 17 (7), 1501.
Yu, B., Yin, H., Zhu, Z., 2018. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In: Proceedings of the 27th International Joint Conference on Artificial Intelligence, pp. 3634–3640.
Zhang, Y., Zhang, Y., 2016. A comparative study of three multivariate short-term freeway traffic flow forecasting methods with missing data. J. Intell. Transp. Syst. 20 (3), 205–218.