Traffic Prediction With Missing Data: A Multi-Task Learning Approach

Ao Wang, Yongchao Ye, *Student Member, IEEE*, Xiaozhuang Song, Shiyao Zhang, *Member, IEEE*, and James J. Q. Yu, *Senior Member, IEEE*

***Abstract*— Traffic speed prediction based on real-world traffic data is a classical problem in intelligent transportation systems (ITS). Most existing traffic speed prediction models are proposed based on the hypothesis that traffic data are complete or have rare missing values. However, such data collected in real-world scenarios are often incomplete due to various human and natural factors. Although this problem can be solved by first estimating the missing values with an imputation model and then applying a prediction model, the former potentially breaks critical latent features and further leads to the error accumulation issues. To tackle this problem, we propose a graph-based spatio-temporal autoencoder that follows an encoder-decoder structure for spatio-temporal traffic speed prediction with missing values. Specifically, we regard the imputation and prediction as two parallel tasks and train them sequentially to eliminate the negative impact of imputation on raw data for prediction and accelerate the model training process. Furthermore, we utilize graph convolutional layers with a self-adaptive adjacency matrix for spatial dependencies modeling and apply gated recurrent units for temporal learning. To evaluate the proposed model, we conduct comprehensive case studies on two real-world traffic datasets with two different missing patterns and a wide and practical missing rate range from 20% to 80%. Experimental results demonstrate that the model consistently outperforms the state-of-the-art traffic prediction with missing values methods and achieves steady performance in the investigated missing scenarios and prediction horizons.**

***Index Terms*— Traffic speed prediction, missing data, spatio-temporal modeling, deep learning, multi-task learning.**

I. Introduction

TRAFFIC speed prediction is among the essential functions of modern intelligent Transportation Systems (ITS) [1]. An accurate traffic prediction model based on geographically interconnected traffic sensor data is the backbone of transportation management. For instance, authorities leverage the traffic forecasting results to make operational decisions, e.g., congestion control, traffic resource allocation, personalized travel recommendations, etc.

Among all traffic forecasting problems, traffic speed prediction is broadly considered indispensable for ITS research. Traffic speed forecasting has been extensively studied in the past decades. Classical methods can be generally categorized into parametric methods (Autoregressive Integrated Moving Average [2], Kalman Filtering [3], etc.) and non-parametric methods (K-Nearest Neighbors algorithm [4], Support Vector Regression [5], etc.). However, the methods mentioned above only utilize the raw data for regression modeling without extracting the data's latent high-dimensional features, thereby being unsatisfactory in prediction performance. In addition, traffic changes are dynamic and complex, i.e., different roads interact with each other, and traffic speed on the same road is also time-variant, rendering the necessity for more advanced methods. The development of deep learning techniques provides researchers and field engineers with new tools for predicting traffic speed more accurately. Contributed by the powerful fine-grained feature extraction capability of deep neural networks (DNN), researchers have embarked on introducing the Convolutional Neural Network (CNN) to extract the spatial dependency in traffic data [6]. However, CNN has limitations on processing traffic data, which is naturally sampled in non-Euclidean space, i.e., along with traffic networks. Considering the similarity between traffic networks and graphs, Graph Neural Networks (GNN) has been proposed to capture the complex spatial dependencies from graph topology [7], [8], [9]. GNN-based approaches have achieved state-of-the-art performance in extracting dependency from traffic data and predicting future traffic speeds [10]. Besides, considering the temporal dependency in the traffic data, researchers have focused on the variants of Recurrent Neural Networks (RNN) [11] such as Long Short-Term Memory (LSTM) [12] and Gated Recurrent Units (GRU) [13] to extract traffic dynamics features among the time series data [7], [9], [14].

However, there is a notable and practical research gap in the aforementioned methods, especially when the traffic data are incomplete or corrupted. The majority of existing traffic prediction methods are proposed based on the hypothesis that the traffic data are complete or have seldom missing values. Nevertheless, they are incapable of or inferior in traffic prediction when there are a larger number of missing values; see [15], [16], [17] for examples. Taking the Attention Based Spatial-Temporal Graph Convolutional Networks (AST-

Manuscript received 30 March 2022; revised 9 September 2022; accepted 29 December 2022. Date of publication 9 January 2023; date of current version 29 March 2023. This work was supported in part by the Stable Support Plan Program of Shenzhen Natural Science Fund under Grant 20200925155105002, in part by the General Program of Guangdong Basic and Applied Basic Research Foundation under Grant 2019A1515011032, and in part by the Guangdong Provincial Key Laboratory under Grant 2020B121201001. The Associate Editor for this article was J. A. Barria. *(Corresponding author: James J. Q. Yu.)*

Ao Wang, Yongchao Ye, Xiaozhuang Song, and James J. Q. Yu are with the Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: 12132359@mail.sustech.edu.cn; 12032868@mail.sustech.edu.cn; shawnsxz97@gmail.com; yujq3@sustech.edu.cn).

Shiyao Zhang is with the Research Institute for Trustworthy Autonomous Systems, Southern University of Science and Technology, Shenzhen 518055, China (e-mail: zhangsy@sustech.edu.cn).

Digital Object Identifier 10.1109/TITS.2022.3233890

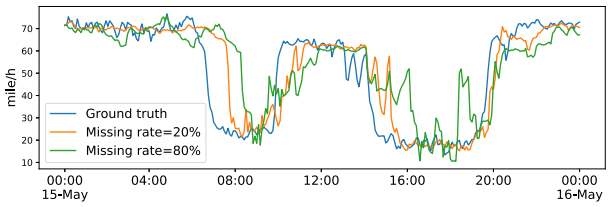


Fig. 1. The 60 min-ahead speed prediction of ASTGCN model on the PEMSD7 dataset with different missing rates.

GCN) [18] as an example, its prediction performance shown in Fig. 1 experiences notable degradation as the missing rate increases, e.g., MAPE rises from 8.18% to 12.20% as the missing rate is increased from 20% to 80%. Besides, the data missing is a practical and common problem for traffic data collected in practice due to various factors. For example, considering the hardware failure and communication error [16], [19], approximately 5% of all traffic data are missing in PeMS, and the missing rate can reach up to 90% in extreme cases in Alberta, Canada [20]. In practical scenarios, the performance of most existing traffic prediction models fluctuates on multiple datasets because of the missing data, which is a common and unpredictable phenomenon in real life. Therefore, we need novel methods to handle the task of traffic prediction with missing values and achieve stable performance.

For traffic speed prediction with missing values, a straightforward solution is first to estimate the missing values using an imputation method and then feed the imputed data into a prediction model [17], [21]. Although recent years have witnessed a plethora of traffic data imputation approaches [22], [23], directly stacking imputation and prediction models has the following limitations. First, the instance-level representations learned in the imputation process are insufficient for the downstream task, which needs fine-grained representations. For example, imputation methods tend to use the average value to estimate the missing position, leading to over-smoothed imputation results, especially for methods based on tensor decomposition [23]. Over-smoothed results may lose the dynamics feature in the traffic data and further render the adverse error accumulation on the downstream tasks, i.e., traffic speed prediction. Second, the impact of imputation results on prediction tasks has not been well-studied, namely, whether accurate imputed data improve prediction performance. Although there exists research on traffic prediction with missing values [15], [16], [17], [19], [24], using traditional training methods and simply concatenating loss functions may undermine prediction accuracy as well as model interpretability. Furthermore, the performance comparison with conventional prediction baselines in simple scenarios is insufficient to demonstrate their superiority.

To jointly tackle the above issues and bridge the research gaps, we propose a novel deep learning model Graph-based Spatio-Temporal AutoEncoder (GSTAE) for traffic speed prediction with missing values. The major focus of this work is to improve traffic prediction performance when historical traffic data contains missing values. GSTAE employs the autoencoder structure, which combines the Graph Convolutional Networks (GCN) with adaptive adjacent matrix and GRU to extract the complex spatio-temporal dependencies in traffic data. Inspired by multi-task learning [25], GSTAE treats imputation and prediction as two parallel tasks rather than standalone and consecutive tasks to eliminate the impact of imputation results on prediction. The encoder module of GSTAE extracts the generalized representation from traffic data with missing values. It is capable of handling different downstream tasks by connecting with different decoder models. For the training process, we design a two-stage training paradigm to achieve multi-task training and improve prediction accuracy. Specifically, the training starts with an imputation task, which trains an encoder that extracts dense representation from traffic data with missing values. Then, the prediction task is trained based on the pre-trained encoder to utilize the domain-specific information in the imputation training process and save computational effort. After the pre-trained encoder extracts the generalized representation from traffic data with missing values, the decoder module makes predictions directly to eliminate the error accumulation issue. The primary contributions are summarized as follows:

- We propose a novel multi-task learning deep neural network model which follows an encoder-decoder structure and treats the imputation and prediction as two parallel tasks to eliminate the impact of imputation on the prediction task.
- We design a two-stage training paradigm to achieve multi-task training and improve performance on the imputation and prediction tasks.
- We evaluate the proposed GSTAE with comprehensive case studies on two real-world traffic datasets and exhaustive missing scenarios. The results indicate that GSTAE consistently outperforms recent methods for traffic prediction with missing values, which also verifies the model's generalization ability.
- We investigate the impact of the imputed results on the prediction models. The result shows that simple splicing well-performed imputation and prediction models are not feasible for traffic prediction with missing data.

The rest of the paper is structured as follows. Sec. II briefly reviews the recent prediction and imputation methods. Sec. III presents the formulation of traffic speed prediction with missing values. Sec. IV introduces the proposed model GSTAE and gives a detailed training scheme. Sec. V presents comprehensive case studies on real-world datasets with different missing scenarios. Finally, the paper is concluded in Sec. VI.

II. Related Work

A. Traffic Prediction

In traffic prediction studies, deep learning models have generated state-of-the-art performance in the past decade. RNN, together with its modern variants LSTM and GRU, can effectively extract traffic dynamics features among the time series data. In addition to temporal correlation, traffic data

also contain complex spatial dependencies. GNN, which can effectively learn non-Euclidean topological correlation, has shown its advantage in capturing complex spatial correlation in traffic graphs. Conventional *spectral* graph convolution is extensively applied in traffic prediction, see [8], [26] for examples. Further studies improved graph convolution by an adaptive dependency matrix [27]. Other studies perform *spatial* convolutions by propagating information to adjacencies. For instance, Li et al. utilized bidirectional random walks on graphs and designed an encoder-decoder to capture the spatio-temporal dependencies [9], Zheng et al. [28] applied the attention mechanism to extra the spatio-temporal dependencies.

While the aforementioned approaches have achieved remarkable performance in traffic prediction, these forecasting solutions have a non-negligible limitation: they rely on fully complete raw traffic data for prediction, while such data collected in real-world scenarios inevitably has missing values.

B. Traffic Data Imputation

Data imputation methods that estimate missing values by analyzing the sampled traffic data are proposed to address the missing data issue. Compared to time series imputation methods designed for general purposes [29], [30], [31], traffic data imputation methods pay more attention to spatio-temporal modeling and can be broadly grouped into two categories, namely, tensor decomposition and deep learning. Tensor decomposition methods utilized low-rank matrix factorization and additional spatial constraints to impute the missing traffic data [32], [33]. Besides, Chen et al. improved imputation performance by applying Bayesian inference to the matrix factorization model [23]. For deep learning methods, a conventional scheme is to capture spatio-temporal correlations with CNN layers by converting the traffic data into pictures [34], [35]. In order to extract local information on graph topology directly and improve imputation accuracy, further researches try to apply GNN to the traffic imputation task. For some examples, Ye et al. incorporated the graph attention mechanism and an encoder-decoder structure for traffic data imputation [36]. Wu et al. utilized graph convolution layers and kriging interpolation to handle the imputation task on unsampled sensors [37]. Despite the respective outstanding performance, these methods primarily focus on the imputation task. They do not yet investigate the impact of imputed results on downstream traffic data analytic tasks, e.g., traffic prediction. For the traffic imputation and prediction tasks, they are closely associated, and the impact of imputation results on prediction performance is non-negligible. Nevertheless, there is a notable gap between these two primary ITS applications.

C. Forecasting With Missing Values

It is reported that missing data may compromise the performance of traffic prediction methods [15]. However, few studies focused on this issue and proposed methods for traffic prediction with missing data. Cui et al. proposed a stacked bidirectional LSTM to capture the temporal information and forecast traffic state, e.g., speed and volume [24]. Zhong et al. constructed multiple graphs to simulate the dynamic correlation of the transportation network [17]. Although these studies integrate imputation and prediction tasks, it is complicated for these models to achieve good performance in imputation and prediction due to the lack of a suitable training method or loss function [25], [38], [39]. Moreover, performance comparison conducted on simple scenarios is insufficient to demonstrate their superiority. Specifically, these studies only select some prediction methods for comparison but do not complete the data through the state-of-the-art imputation methods before the comparison.

Inspired by previous studies, in this paper, we propose a new multi-task learning deep neural network model — GSTAE — based on the simple yet effective encoder-decoder structure to predict traffic speed with missing values. To facilitate the multi-task nature of traffic speed imputation and prediction tasks, they are trained sequentially since the prediction is intuitively considered complex and challenging over imputation.

III. Problem Definition

A. Traffic Speed Prediction

The objective of traffic speed prediction is to forecast the future traffic speed based on historical data measured from sensor nodes in urban traffic networks. Generally, this objective can be expressed by

$$[X_{(t-T_h+1):(t)}, \mathcal{G}] \xrightarrow{f(\cdot)} X_{(t+1):(t+T_p)}, \tag{1}$$

where $X_{(t-T_h+1):(t)} \in \mathbb{R}^{T_h \times N}$ is the historical traffic speed of N sensors from time step $(t - T_h + 1)$ to t. The traffic prediction model needs to establish a function f to predict the traffic speed of the next T_p steps based on T_h steps in the past.

B. Traffic Network Graph

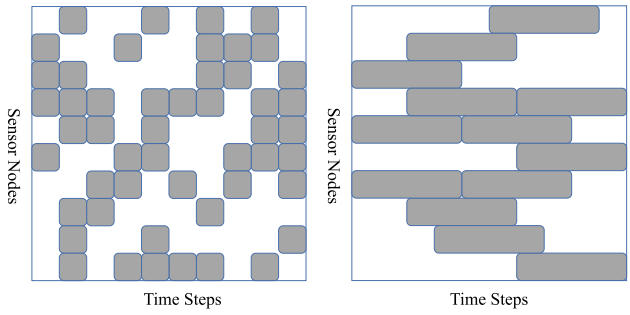
In this paper, we denote the urban traffic network by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ to represent prior geo-information [7], [40], where $\mathcal{V}$ is the set of traffic sensors, and $\mathcal{E}$ is the set of edges, which represents the connectivity between sensors. The adjacent matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ of graph $\mathcal{G}$ represents the relationship among sensors, which is generated by the thresholded Gaussian kernel following the well-recognized approach presented in [7]:

$$\mathbf{A}_{(ij)} = \begin{cases} \exp\left(-\frac{d_{ij}^2}{\sigma^2}\right), & i \neq j \text{ and } \left(-\frac{d_{ij}^2}{\sigma^2}\right) \geq \epsilon \\ 0, & \text{otherwise.} \end{cases}, \tag{2}$$

where d_{ij} is the Euclidean distance between sensor nodes i and j, σ is the standard deviation of distance, and ϵ (set to 0.5 by default [40]) is the threshold that controls the sparsity of weight matrix $\mathbf{A}$, respectively.

C. Traffic Speed Prediction With Missing Values

For time step t, the traffic speed data can be represented by $X_{(t)} \in \mathbb{R}^N$, which is observed on all N sensors in the traffic



(a) Random Missing (b) Non-Random Missing

Fig. 2. Diagram of different missing patterns. The grey blocks represent the missing positions.

network. A missing mask $M_{(t)} \in \mathbb{B}^N$ that records the missing positions is defined by

$$M^i_{(t)} = \begin{cases} 1, \text{ if } X^i_{(t)} \text{ is observed} \\ 0, \text{ if } X^i_{(t)} \text{ is missing} \end{cases}. \tag{3}$$

Following the common practice in previous studies [22], [23], we investigate two representative missing patterns in this study, namely, Random Missing (RM) and Non-Random Missing (NRM) [22], [23], which are defined as follows:

- Random Missing (RM): Missing values appear randomly and independently. Fig. 2a is an example of the Random Missing pattern, where the grey cells denote the masked (missing) values and the white ones are the observed values.
- Non-Random Missing (NRM): In practice, sensor nodes may fail for a notable period of time, rendering data loss. Fig. 2b is an example of such Non-Random Missing patterns.

For the task of traffic speed prediction with missing values, we need to learn a function $g(\cdot)$ which is capable of predicting the traffic speed of the next T_p steps based on the graph $\mathcal{G}$, historical T_h steps traffic speed X, and corresponding mask matrix M. Compared to Eq. (1), $g(\cdot)$ considers the missing items in the traffic data, and its relationship can be expressed as

$$[X_{(t-T_h+1):(t)}, M_{(t-T_h+1):(t)}, \mathcal{G}] \xrightarrow{g(\cdot)} X_{(t+1):(t+T_p)}, \tag{4}$$

where $X_{(t-T_h+1):(t)}, M_{(t-T_h+1):(t)} \in \mathbb{R}^{T_h \times N}$ and $X_{(t+1):(t+T_p)} \in \mathbb{R}^{T_p \times N}$.

IV. Graph-Based Spatio-Temporal AutoEncoder

In this section, we elaborate on the proposed GSTAE in depth. We begin by introducing the overview of the model and design of each constituting component. Following that, we detail each sub-module and its components. Finally, we discuss the unique training scheme of the proposed model.

A. GSTAE Overview

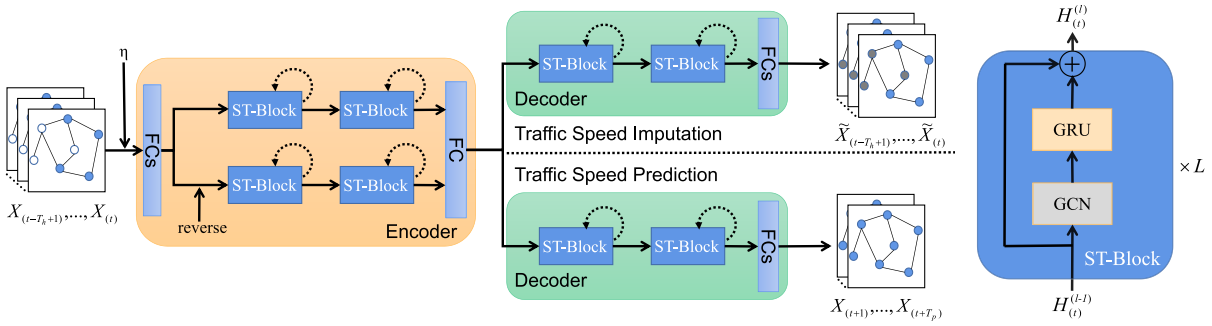
Fig. 3a illustrates the overall structure of the proposed GSTAE. The model is structured as an encoder-decoder structure, which has achieved state-of-the-art performance in a variety of sequence-to-sequence tasks [41]. The encoder module extracts the complex spatio-temporal dependencies from the historical traffic data with missing values and represents them as a generalized dense representation suitable for downstream tasks. The decoder is coupled to perform specific downstream tasks, e.g., imputation and prediction, using the generalized representation from the encoder module.

For the training process, it is possible to train the task of traffic prediction with missing values directly. However, the task from historical traffic speed data with missing values to future traffic speed data is challenging and complicated for the model to train. It is true that we can split the task into two sub-tasks, one for processing historical data with missing values and imputing historical data and another for processing imputed traffic data and predicting future traffic speed. Nonetheless, the accumulation of imputation and prediction errors can adversely affect the combined performance. Inspired by [42], we treat the encoder module as a unified framework to extract features from input data, which is suitable for various downstream tasks, e.g., imputation and prediction. As prediction and imputation have different characteristics, imputation and prediction tasks are trained sequentially. It can be regarded as a trade-off between predicting the future traffic speed from historical data with missing values and splitting this task into imputation and prediction sub-tasks. First, we train the model with the imputation task to enable the encoder to extract dense representations from the original data with missing values. In other words, the imputation is an auxiliary task that speeds up and improves the subsequent prediction. We view the encoder as a unified framework for extracting dense representations of data suited for various downstream tasks. Thus, the encoder extracts features from the original data while not bringing additional features tailored for specific tasks. For traffic imputation, existing methods tend to smoothen the imputed traffic dynamics, which may not be suitable for subsequent traffic prediction. After the encoder module is trained by the imputation task, it extracts the generalized representation from traffic data with missing values. The pre-trained encoder can speed up and improve the subsequent prediction training.

In summary, we propose an encoder-decoder-based model called GSTAE to extract generalized dense representations from traffic data with missing values and handle two downstream tasks, namely, traffic data imputation and prediction.

1) Traffic Data Imputation: As models that follow the encoder-decoder structure have shown superiority in learning the hidden representation and reconstructing the data [42], [43], we design GSTAE following the encoder-decoder structure. We first train the proposed GSTAE with the imputation task to extract representations from the original data with missing values. After the training process, the GSTAE model can handle the traffic imputation task. We hypothesize that the encoder has extracted good representations from the original data with missing values, and the pre-trained encoder can be transferred to accelerate the prediction training process.

Due to the fact that the traffic data are time series, the historical traffic state imposes a notable influence on the future dynamics, rendering future states to reflect the historical ones. Additionally, traffic data contain complex spatial dependencies



(a) Overview of GSTAE (b) ST-Block

Fig. 3. Structure of the proposed GSTAE. (a) The model consists of an encoder and decoder. The encoder consists of the forward and backward recurrent ST-Blocks, the decoder consists of forward recurrent ST-Blocks. (b) The ST-Block is composed of GCN and GRU layers and can be stacked.

within the traffic network topology. As a result, we devise and employ bidirectional recurrency — *ST-Blocks* as will be introduced in Sec. IV-B — to extract the correlated spatio-temporal dependencies. For the decoder structure, we use forward recurrent *ST-Blocks* to generate the prediction output auto-regressively.

Let $X_{(t-T_h+1):(t)}$, $M_{(t-T_h+1):(t)}$ and $\eta \in \mathbb{R}^{T_h \times N}$ be the input of the model, where $X_m = X_{(t-T_h+1):(t)} \odot M_{(t-T_h+1):(t)}$ is the historical traffic speed data with missing values, $\odot$ denotes element wise multiplication. Before feeding X_m to the encoder, a random noise η sampled from the standard distribution $\mathcal{N}(0, 0.01)$ is first added to historical traffic data X_m based on the design principle of denoising auto-encoder [43]. Next, since the location information of missing data is non-negligible for the model to extract the dense representation [29]. We combine the noisy input with $M_{(t-T_h+1):(t)}$ to assist the model in identifying the missing positions of the historical input data and get the combined input $X_{\text{comb}} \in \mathbb{R}^{T_h \times N \times 2}$. Finally, we transform the combined input X_{comb} to $H^{(0)}_{(t-T_h+1):(t)} \in \mathbb{R}^{T_h \times N \times D}$ through two fully-connected layers to combine the historical traffic information and the observation location information as follows:

$$\begin{aligned} H^{(0)}_{(t-T_h+1):(t)} &= \text{FCs}\left(X_{\text{comb}}\right) \\ &= \text{FCs}\left(\left[X_m + \eta\right] \oplus M_{(t-T_h+1):(t)}\right), \end{aligned} \tag{5}$$

where $\oplus$ denotes the concatenate operation. Next, $H^{(0)}_{(t-T_h+1):(t)}$ is passed to the encoder module to extract the dense representation. The encoder module consists of a *forward ST-Block* and a *backward ST-Block* stacked L layers of ST-Blocks. Their structures and input are the same but with different data processing directions. The *forward ST-Block* processes the $H^{(0)}_{(t-T_h+1):(t)}$ from step $(t-T_h+1)$ to t, and the *backward ST-Block* processes from step t to $(t-T_h+1)$. Fig. 4 demonstrates L layers of ST-Blocks that extract spatio-temporal dependencies. For the input $H^{(0)}_{(t)}$ of time step t, the proposed model extracts the hidden spatio-temporal representation by L layers of ST-Blocks. Then, features of different time steps are aggregated by updating the hidden state of the GRU module in ST-Blocks step by step. Taking the l-th ST-Block in *forward ST-Block* and time step t as an example, $h^{(l)}_{(t-1)}$ is the hidden state of the GRU module in l-th ST-Block from the previous

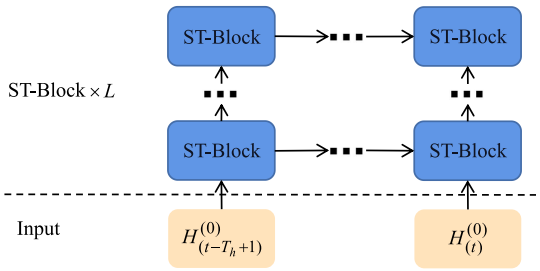


Fig. 4. Diagram of L layers of ST-Blocks.

time step. And the input data $H^{(l-1)}_{(t)}$ is processed based on the following propagation rule:

$$H^{(l)}_{(t)} = \text{STB}\left(H^{(l-1)}_{(t)}, h^{(l)}_{(t-1)}\right), \tag{6}$$

where $H^{(l)}_{(t)}$ is the output of l-th ST-Block, which is the same as the updated $h^{(l)}_{(t)}$ and will be proposed to the next layer of ST-Block.

The last hidden state h^f in the *forward ST-Block* and h^b in the *backward ST-Block* are combined and transformed to h_{dec} through a fully-connected layer to combine features from two directions and keep the feature dimension of h_{dec} consistent with h^f and h^b. The resulting h_{dec} is subsequently passed to the decoder module.

Different from the aforementioned encoder, the decoder only contains the *forward ST-Blocks* to generate the output auto-regressively as follows:

$$\hat{X}_{(t)} = \text{Dec}\left(h_{(t-1)}, \tilde{X}_{(t-1)}\right), \tag{7}$$

where $\hat{X}_{(t)}$ is the estimated traffic speed of time step t. Next, by filling the missing values in $X_{(t)}$ with corresponding values in $\hat{X}_{(t)}$, we can get the imputed traffic speed $\tilde{X}_{(t)} = M_{(t)} \odot X_{(t)} + (1 - M_{(t)}) \odot \hat{X}_{(t)}$, and $\odot$ denotes elementwise multiplication. To initialize the decoder, we set the $h_{(0)} = h_{\text{dec}}$ and $\tilde{X}_{(0)} = 0$. Finally, the output of the decoder $\tilde{X}_{(t-T_h+1):(t)} \in \mathbb{R}^{T_h \times N}$, which denotes the imputed traffic speed, has the same data shape as the input data $X_{(t-T_h+1):(t)}$.

Regarding the purpose of this model, namely, extracting the dense representation of input data with missing values and

reconstructing the data $X_{(t-T_h+1):(t)}$, the model is trained by minimizing the following reconstruction loss:

$$L(\theta, \vartheta) = \sum_{i=1}^{T_h} ||(X_{(t-T_h+i)} - \tilde{X}_{(t-T_h+i)}) \odot (1 - M_{(t-T_h+i)})||^2, \tag{8}$$

where θ and ϑ are the parameters of the encoder and decoder, respectively. For the training process of the model, it needs to adjust the parameters, so that given the input, the model can generate an output that is the same as the label or as close as possible. Because the loss function $L(\theta, \vartheta)$ represents the difference between the model output $\tilde{X}_{(t-T_h+i)}$ and label $X_{(t-T_h+i)}$, for one layer of the model, we can update the weights by computing the partial derivative of $L(\theta, \vartheta)$ with respect to each weight. This updating process can be executed sequentially from the top layer to the bottom layer so that the output of the model is getting increasingly closer to the label. See [44] for more detailed information.

2) Traffic Speed Prediction: After the model is capable of handling data imputation, we may conclude that the comprising encoder module can extract the dense representation from input data, thereby benefiting the traffic prediction training process. As a result, we connect this pre-trained encoder to a new decoder module with the same structure and train the new combined model on the traffic prediction task.

In this process, we still employ the same noisy traffic data $X_{(t-T_h+1):(t)} + \eta$ as the input, making the output to be the traffic speed of the next T_p time steps $\hat{X}_{(t+1):(t+T_p)}$. In order to minimize the difference between the predicted traffic speed $\hat{X}_{(t+1):(t+T_p)}$ and the observation $X_{(t+1):(t+T_p)}$, we train the model with the following mean squared error loss:

$$L(\delta) = \sum_{i=1}^{T_p} ||X_{(t+i)} - \hat{X}_{(t+i)}||^2, \tag{9}$$

where δ is the set of training parameters in the new decoder.

B. Spatio-Temporal Block

Fig. 3 shows the overview of ST-Block, which consists of a GCN layer and a GRU one. Taking the *forward ST-Block* in the encoder module as an example, at time step t, $H_{(t)}^{(0)} \in \mathbb{R}^{N \times D}$ is the input, where N is the number of sensor nodes, and D is the hidden feature dimension. Through the process of extracting the correlated spatio-temporal dependencies, the output of *ST-Block* $H_{(t)}^{(l)}$ is generated. Additionally, in order to prevent the over-smooth issues, we apply a residual connection at the end of ST-Block. For each *ST-Block*, we ensure that the input and output data dimensions are identical to allow the residual connection operation. In the following, we introduce the GCN and GRU modules, respectively.

1) Graph Convolution Layer: Traffic data contain complex spatial correlation because the traffic state of a road section is affected by adjacent sections. Traditional CNN can capture the spatial dependency through a small kernel, which constrains the operation range and provides the weight of information aggregation. It performs convolution operations in Euclidean space. However, such a uniform shape is not suitable for graph structures due to their irregularity, e.g., the number of adjacent sections is different for each road section. To perform the convolution operation in the non-Euclidean space, GCN is proposed to capture the complex spatial dependencies in the graph data. Kipf and Welling [45] proposed the first-order approximation of Chebyshev spectral filter [46]. Let the road sections be the nodes in the traffic network $\mathcal{G}$. For each road section, its adjacent sections denote neighbors. GCN model learns node embedding through aggregating information from neighbors based on the traffic network structure. Given the input data $X \in \mathbb{R}^{N \times D}$, where N and D denote the number of nodes and the feature dimension, respectively, the output $Y_G \in \mathbb{R}^{N \times D}$ is produced based on the following propagation rule:

$$Y_G = \sigma\left(\tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}} X W\right), \tag{10}$$

where $\tilde{A} \in \mathbb{R}^{N \times N}$ is the weight adjacent matrix with self-connection, and $\tilde{D}$ is the degree matrix of $\tilde{A}$, and $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. $W \in \mathbb{R}^{D \times D}$ is the parameter matrix of the GCN module. $\sigma(\cdot)$ denotes the nonlinear activation function, which is set to ReLU by default in GSTAE. For the GCN module in the l-th layer of ST-Block, the input of time step t is $H_{(t)}^{(l-1)}$.

However, the widely recognized geography-defined adjacent matrix cannot contain the complete spatial dependency correlation. For instance, two nodes have similar traffic states but are far away from each other. Even though we can stack and construct a deep GCN model to make them aggregate information from each, too many layers of GCN can lead to node embedding indistinguishable and over-smooth problems. In addition, this approach cannot be applied to other problems without the knowledge of graph structure. Recent studies have resolved this issue by applying a self-adaptive adjacent matrix and achieving state-of-the-art performance [27], [47]. Following the design principle, we utilize a self-adaptive adjacent matrix $\tilde{A}_{\text{adp}}$ defined as follows:

$$\tilde{A}_{\text{adp}} = \text{SoftMax}\left(\text{ReLU}(E_1 \times E_2^T)\right), \tag{11}$$

where $E_1, E_2^T \in \mathbb{R}^{N \times C}$ are learnable parameters which are initialized randomly before the training process.

Since both the geography-defined and the self-adaptive adjacent matrix are essential for extracting geographic and hidden spatial dependencies in traffic data, we combine the geography-defined normalized adjacent matrix $A_{\text{def}} = \tilde{D}^{-\frac{1}{2}} \tilde{A} \tilde{D}^{-\frac{1}{2}}$, which represents the relationship between different nodes, and the self-adaptive matrix $\tilde{A}_{\text{adp}}$. Through the combination of A_{def} and $\tilde{A}_{\text{adp}}$, the GCN module can extract more complete spatial correlation information and achieve better performance than only using A_{def}. Besides, during the training process, the geography-defined traffic network ensures the GCN operation can extract spatial dependency and stabilizes the training process before the model learns the correct and sufficient hidden spatial information. And the graph convolution process can be expressed as:

$$Y_G = \sigma\left(A_{\text{def}} X W_{\text{def}} + \tilde{A}_{\text{adp}} X W_{\text{adp}}\right). \tag{12}$$

In summary, we can make each node gather information from not only its first-order neighbors but also other correlated nodes through one layer of GCN. Besides, the process in which traffic conditions influence each other will not terminate after one-hop propagation, which indicates that only using one layer of the GCN module is insufficient to extract the spatial correlations [48]. This issue can be solved by stacking GCN layers to extend the aggregation. However, stacking too many GCN layers may bring a massive computational burden and lead to "over-smooth" issues [49]. Thus, in order to expand the coverage of information gathering and prevent the over-smoothing issue, we use three layers of ST-Block to extract the complex spatial correlation in traffic data [50].

2) Gated Recurrent Unit: After extracting spatial features by GCN layers, we apply recurrent layers to learn the temporal characteristics in traffic data. The RNNs are effective tools to process the sequence data and model temporal dependency. However, RNNs have limitations in processing the long sequence data because of the gradient vanishing and explosion problem. In *ST-Block*, we leverage GRU [13] to learn the temporal dependency, which is a simple and effective variant of RNNs. GRU introduces the concept of gate units that control the flow of information to solve gradient problems, and the structure of GRU is more simple than LSTM [12] relatively. Specifically, the gate units consist of two types, namely, reset gate and update gate, to control how much state information of previous time steps can be remembered and passed to the current time step:

$$\begin{aligned} u^{(t)} &= \sigma(W_z[X^{(t)}, h^{(t-1)}] + b_u), \\ r^{(t)} &= \sigma(W_r[X^{(t)}, h^{(t-1)}] + b_r), \\ c^{(t)} &= \tanh(W[X^{(t)}, r^{(t)} \odot h^{(t-1)}] + b_c), \\ h^{(t)} &= u^{(t)} \odot h^{(t-1)} + (1 - u^{(t)}) \odot c^{(t)}, \end{aligned} \tag{13}$$

where $h^{(t)}$ is the hidden state at time t, $X^{(t)}$ is the input feature of step t, $r^{(t)}$ is the reset gate, and $u^{(t)}$ is the update gate, respectively. Because the GCN and GRU modules are stacked consecutive, the input of GRU $X^{(t)} = Y_G$ is the output of the GCN module. By sequentially incorporating traffic data into the model, GRU is capable of capturing the dynamic temporal dependency inherent in traffic time series.

C. Training and Inference

Since GSTAE is proposed to handle two sub-tasks, namely, imputation and prediction, the typical end-to-end training scheme is not suitable for GSTAE. Moreover, due to the complexity of the imputation and prediction tasks being quite different, the common multi-task training strategy, such as training together by computing the loss function for all sub-tasks, is also not suitable for GSTAE. We treat the prediction task as the main task and the imputation task as the auxiliary task to solve these issues and train the model through a two-stage training paradigm. First, the imputation task is trained to enable the encoder module of GSTAE to extract dense representations from the original data with missing values. After the imputation training process, the GSTAE model is trained with the traffic prediction task.

Training scheme:

1) Initialize the model parameters θ, ϑ, and the training dataset of the imputation task. Input is traffic speed data with missing values $X \odot M$, and the label is the corresponding completed traffic speed data X.
2) Use the Adam optimization algorithm [51] to update the model parameters θ and ϑ based on the loss function $L(\theta, \vartheta)$.
3) Freeze the encoder module and connect with a new decoder module whose parameters are δ. Initialize the training dataset of the prediction task. Input is historical traffic speed data with missing values $X_{(t-T_h+1):t} \odot M_{(t-T_h+1):t}$, and the label is the traffic speed data of the next T_p time steps $X_{(t+1):(t+T_p)}$.
4) Use the Adam optimization algorithm to update the decoder parameters δ based on the loss function $L(\delta)$.

In summary, considering the complexity of the imputation and prediction tasks being quite different, the imputation and prediction tasks are trained sequentially instead of training together. Besides, through the first training of the imputation task, the encoder module can obtain a suitable parameter set to extract the dense representation from input data. It can speed up the training process of the subsequent prediction task, which is much more complex than the imputation task.

V. Case Studies

In this section, we comprehensively evaluate the performance of the proposed GSTAE with two real-world datasets. We first assess the efficacy of traffic prediction and imputation tasks under a variety of missing scenarios and compare the proposed model with state-of-the-art approaches. Then, we conduct a series of ablation tests to reveal the effectiveness of GSTAE sub-modules. Finally, we analyze the sensitivity of hyperparameters on the model performance.

A. Dataset and Configurations

All case studies are conducted on two real-world datasets: **PEMSD7** [40] and **METR-LA** [9]. PEMSD7 is a popular traffic speed prediction dataset collected from Caltrans Performance Measurement System (PeMS) by over 39 000 sensor stations on highways in California, United States. We follow the data processing method in [40] and adopt weekday traffic data from May 1st, 2012 to June 30th, 2012. METR-LA data was collected from 207 sensors along the Los Angeles Freeway during weekdays between March 1st, 2012 and June 30th, 2012. These two datasets share the same sampling interval of 5 min, i.e., each sensor generates 288 data points per day. All missing data in the ground truth of these two datasets are imputed by linear interpolation and are excluded in the evaluation stage, following the common practice in literature [7], [18], [40].

In this study, we investigate two missing patterns, as previously mentioned in Sec. III, and a wide missing rate range from 20% to 80% with 20% as the interval. For each sample, the missing values are masked according to the specific missing pattern. For cross-validation, each dataset is grouped

into non-overlapping training, validation, and testing subsets with 60%, 20%, and 20% of the complete dataset, respectively.

All case studies are conducted on an Intel Xeon E5 CPU and an nVidia GTX 2080 Ti GPU. The proposed model and all baselines use the historical traffic speed of the past $T_h = 12$ time steps (60 min) to predict that of $T_p = 3, 6, 12$ time steps (15, 30, 60 min) ahead. For NRM, we set the length of mask block $l = 4$ as introduced in Sec. III-C. Unless otherwise specified, the number of ST-Block layers in each recurrent sub-modules is set to $L = 3$, and the output dimension of each ST-Block is set to $D = 64$. For the fully-connected layers that process the input data and generate the output, the hidden dimension is set to $K = 256$. The proposed GSTAE is trained with the efficient stochastic optimization method Adam, which updates the model parameter based on the first-order gradients with the weight decay of 0.7 for every five epochs. The maximum epoch is 200, the initial learning rate is 0.001, and the batch size is 32. To avoid the overfitting problem, we apply the early stopping in the training process and set the patience to 10.

To evaluate the prediction performance, we employ Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE) as evaluation metrics, which are defined as follows:

$$\begin{aligned} \text{RMSE} &= \sqrt{\frac{1}{n}\sum_{i=1}^{n}(X_i - \hat{X}_i)^2}, \\ \text{MAE} &= \frac{1}{n}\sum_{i=1}^{n}\left|X_i - \hat{X}_i\right|, \\ \text{MAPE} &= \frac{1}{n}\sum_{i=1}^{n}\left|\frac{X_i - \hat{X}_i}{X_i}\right| \times 100\%, \end{aligned} \tag{14}$$

where X_i is the real traffic speed at time step i, and $\hat{X}_i$ is the predicted speed at time step i, respectively.

B. Prediction Accuracy

We first compare the prediction accuracy of the proposed GSTAE with baselines. A common practice for traffic predicting with missing data is to impute incomplete data with an external imputation model and then feed the imputed data into prediction models [17], [21]. Thus, we combine state-of-the-art traffic imputation and prediction methods to develop baselines of the typical two-phase traffic predictors. For the imputation methods, we adopt

- Bidirectional Recurrent Imputation for Time Series (BRITS) [29]: BRITS is a general imputation model for time series data. It utilizes the bidirectional recurrent network for time series modeling. To get better imputation performance, we set the dimension of the RNN hidden state to 64 based on offline preliminary experiments.
- Bayesian temporal matrix factorization (BTMF) [22]: BTMF employs the matrix factorization method with Bayesian inference to capture the temporal dependency and has remarkable performance compared to other imputation methods based on tensor decomposition.
- BTMF (50) [22]: To get better imputation accuracy, we set the matrix rank to 50 in BTMF based on preliminary experiments against the default 10 as used above.

For the traffic prediction models, we employ

- Diffusion Convolutional Recurrent Neural Network (DCRNN) [9]: DCRNN embeds the graph by bidirectional random walks and applies gated recurrent units into an encoder-decoder model to capture the spatio-temporal dependencies.
- ASTGCN [18]: ASTGCN introduces the spatial and temporal attention mechanisms to extract the dynamic spatio-temporal dependencies and considers the historical data with a more extended range (e.g., historical traffic data one day ago, one week ago).
- Graph Multi-Attention Network (GMAN) [28]: GMAN utilizes a new spatio-temporal attention mechanism which is calculated through input data and pre-defined spatio-temporal embedding information to extract the complex spatio-temporal correlations.
- Fully Connected Gated Graph Architecture (FC-GAGA) [52]: FC-GAGA builds the hard graph gate mechanism to extract spatial dependency without requiring the geographic-knowledge and extracts temporal dependency through the time gate mechanism.

Furthermore, we also implement recent studies on traffic prediction with missing values based on the introductions and configurations from their original proposals.

- Graph Convolutional Bidirectional Recurrent Neural Network (GCBRNN) [16]: GCBRNN combines the GCN and GRU modules to build the graph convolutional gated recurrent bidirectional unit, which imputes historical data and make prediction consecutively.
- Recurrent Imputation based Heterogeneous Graph Convolution Network (RIHGCN) [17]: RIHGCN utilizes heterogeneous graph structure and LSTM to extract dynamic spatio-temporal correlations.

Table I and II summarize the simulation results on the PEMSD7 and METR-LA datasets respectively. Generally, the proposed GSTAE outperforms all baselines in most missing scenarios and prediction lengths. Especially for the NRM missing type, a more complex missing type than RM, GSTAE is better than other baseline models for all missing rates. This is because the bidirectional ST-Block proposed in GSTAE can extract the dynamic temporal correlation in the traffic data. At the same time, the GCN model in the ST-Block module can capture the complex spatial correlation, and the introduced adaptive adjacency matrix can also learn the hidden spatial information outside the network topological map. Furthermore, the two-step training method used in the training process ensures that the encoder module can extract enough spatio-temporal information to process downstream tasks and alleviates the error accumulation issues.

For RM missing, there are three cases where the proposed GSTAE slightly underperforms the baseline models based on ASTGCN or GMAN. This is because that ASTGCN has a broader receptive field. It can obtain important information from historical data one day and one week before. At the

TABLE I

PERFORMANCE COMPARISON WITH DIFFERENT PREDICTION LENGTHS (15 min / 30 min / 60 min) ON PEMSD7 DATASET WITH MAPE. MAPE IS REPORTED IN PERCENTAGE (%)

Missing Type	Method		Missing Rate (%)			
	Impute	Predict	20	40	60	80
RM	BRITS	DCRNN	12.84 / 13.64 / 15.35	12.71 / 13.26 / 15.79	13.27 / 13.78 / 15.56	14.91 / 15.35 / 16.54
		ASTGCN	4.54 / 6.14 / 8.61	4.76 / 6.17 / 8.80	5.42 / 6.83 / 8.80	7.62 / 8.48 / 10.40
		GMAN	4.26 / 5.48 / 7.05	4.49 / **5.53** / 6.85	5.12 / 6.04 / 7.12	6.44 / 6.78 / 7.72
		FC-GAGA	28.06 / 28.93 / 29.79	28.07 / 28.45 / 29.09	28.16 / 29.16 / 30.02	29.05 / 29.71 / 30.25
	BTMF	DCRNN	14.41 / 14.56 / 15.40	14.62 / 14.95 / 15.24	14.41 / 14.66 / 15.59	14.68 / 14.81 / 15.65
		ASTGCN	8.69 / 8.73 / 9.44	8.76 / 8.70 / 9.50	8.64 / 8.83 / 9.49	8.91 / 8.93 / 9.61
		GMAN	8.18 / 8.55 / 8.80	8.16 / 8.40 / 8.75	8.28 / 8.34 / 8.57	8.38 / 8.68 / 8.97
		FC-GAGA	29.59 / 29.87 / 30.26	29.41 / 29.66 / 30.12	29.49 / 29.74 / 30.15	29.30 / 29.54 / 29.95
	BTMF (50)	DCRNN	12.61 / 13.26 / 14.45	12.48 / 13.04 / 14.93	13.10 / 13.45 / 15.15	13.39 / 13.84 / 15.18
		ASTGCN	5.70 / 6.48 / 7.99	5.75 / 6.87 / 8.48	5.98 / 6.75 / 8.23	6.88 / 7.44 / 8.55
		GMAN	5.30 / 5.77 / 7.12	5.50 / 6.04 / 7.18	5.58 / 6.32 / 7.23	6.25 / 6.96 / 7.77
		FC-GAGA	28.19 / 28.79 / 29.49	27.74 / 28.67 / 29.39	28.72 / 29.15 / 29.77	28.31 / 28.89 / 29.62
	GCBRNN		8.46 / 8.88 / 9.12	8.61 / 9.22 / 9.64	8.77 / 9.41 / 10.12	9.57 / 9.87 / 10.25
	RIHGCN		5.51 / 7.02 / 7.95	6.12 / 7.54 / 9.19	6.77 / 7.87 / 8.79	8.19 / 9.23 / 9.69
	GSTAE		**4.13 / 5.21 / 6.51**	**4.44** / 5.54 / **6.67**	**4.98 / 5.89 / 6.89**	**5.90 / 6.68 / 7.60**
NRM	BRITS	DCRNN	12.99 / 13.54 / 15.26	13.89 / 14.76 / 16.25	15.53 / 16.19 / 17.66	18.28 / 18.83 / 20.82
		ASTGCN	5.18 / 6.88 / 8.66	6.68 / 7.75 / 10.08	8.40 / 9.06 / 10.74	9.14 / 11.95 / 13.31
		GMAN	4.78 / 5.83 / 7.21	5.41 / 6.33 / 7.55	6.47 / 7.02 / 7.81	7.74 / 8.01 / 9.02
		FC-GAGA	28.13 / 28.75 / 29.49	29.01 / 30.13 / 31.04	30.13 / 30.47 / 30.81	30.36 / 30.81 / 31.44
	BTMF	DCRNN	14.53 / 14.54 / 15.52	14.51 / 14.41 / 15.56	14.53 / 14.79 / 15.67	15.26 / 15.42 / 16.05
		ASTGCN	8.73 / 9.05 / 9.77	6.23 / 8.86 / 9.61	8.71 / 9.15 / 9.84	9.38 / 9.55 / 10.16
		GMAN	8.18 / 8.55 / 8.80	8.32 / 8.58 / 8.80	8.44 / 8.67 / 9.07	9.12 / 9.32 / 9.47
		FC-GAGA	29.43 / 29.87 / 30.40	29.26 / 29.71 / 30.23	29.08 / 29.43 / 29.93	29.12 / 29.38 / 29.76
	BTMF (50)	DCRNN	12.60 / 12.81 / 14.76	13.32 / 13.53 / 15.07	13.35 / 14.17 / 15.57	15.01 / 15.30 / 16.56
		ASTGCN	5.94 / 6.62 / 8.50	8.81 / 7.14 / 8.59	6.96 / 7.54 / 9.08	8.59 / 9.56 / 10.07
		GMAN	5.41 / 5.96 / 7.31	5.69 / 6.32 / 7.59	6.51 / 7.00 / 7.86	8.16 / 8.32 / 9.23
		FC-GAGA	27.70 / 28.54 / 29.55	28.60 / 29.01 / 29.64	28.90 / 29.62 / 30.31	29.40 / 29.80 / 30.30
	GCBRNN		8.61 / 9.05 / 9.66	8.80 / 9.57 / 10.25	9.19 / 9.55 / 10.16	10.27 / 10.30 / 10.70
	RIHGCN		7.01 / 7.94 / 8.32	7.85 / 8.29 / 8.66	8.69 / 8.38 / 9.66	9.61 / 9.78 / 10.44
	GSTAE		**4.30 / 5.39 / 6.60**	**4.97 / 5.85 / 6.92**	**5.83 / 6.60 / 7.71**	**7.40 / 7.81 / 8.53**

same time, the multi-layer attention mechanism introduced by GMAN performs well in extracting spatio-temporal correlations from traffic data and making predictions. Nevertheless, the proposed GSTAE achieves superior results to all baselines in most missing scenarios and prediction lengths. Although there are two scenarios where the proposed GSTAE performs worse than GMAN or ASTGCN, the performance gap is negligible. In general, the proposed GSTAE is more capable of extracting complex spatio-temporal correlation information from historical traffic data with missing values and has more obvious advantages in long-term traffic speed prediction tasks.

For all missing scenarios and prediction lengths, using FC-GAGA as the prediction model has clearly overfits. One possible reason is that the missing data makes the feature difference between nodes negligible, complicating the hard graph gate mechanism to learn spatial dependencies. Compared with combinations of imputation and prediction methods (e.g., ASTGCN-based and GMAN-based ones), the GCBRNN and RIHGCN, which focus on traffic prediction with missing values, have lower accuracy. This is due to the fact that these methods merely integrate the imputation and prediction task without designing a suitable training method or loss function, which complicates the optimization process of the model.

For the PEMSD7 dataset, when the missing type is RM, the performance of DCRNN-based baseline models is the worst. Only processing data in the forward direction based on the encoder-decoder model makes the DCRNN most susceptible to missing data. From the perspective of imputation methods, the performance of the baseline models based on BTMF (50) is generally better than that of the baseline models based on BTMF. This is because increasing the model's hyperparameter (rank of the matrix) improves the model's capabilities. The baseline models based on BRITS have the best performance when the missing rate is relatively small (from 20% to 60%), but the advantage over BTMF and variant is gradually narrowing. When the missing type is NRM, there is no change in the relative performance of each baseline model. However, since NRM is a more complex type of missing data than RM, all models' absolute prediction performance observes a slight decrease. On the METR-LA dataset, the performance relationship of different baseline models has changed compared with the results on PEMSD7. BTMF (50) is better than that of the baseline models based on BRITS for any missing rate.

TABLE II

PERFORMANCE COMPARISON WITH DIFFERENT PREDICTION LENGTHS (15 min / 30 min / 60 min) ON METR-LA DATASET WITH MAPE. MAPE IS REPORTED IN PERCENTAGE (%)

Missing Type	Method		Missing Rate (%)			
	Impute	Predict	20	40	60	80
RM	BRITS	DCRNN	8.25 / 9.14 / 11.13	8.04 / 9.12 / 11.25	8.39 / 9.45 / 11.58	10.47 / 11.09 / 12.69
		ASTGCN	6.63 / 8.06 / 9.55	6.59 / 7.87 / 9.88	6.87 / 8.00 / 9.80	8.28 / 9.38 / 10.10
		GMAN	29.42 / 29.45 / 29.18	29.34 / 29.39 / 29.02	29.54 / 29.46 / 29.27	29.58 / 29.20 / 29.14
		FC-GAGA	23.29 / 23.40 / 23.65	23.97 / 24.02 / 24.01	25.02 / 25.04 / 25.27	25.53 / 25.70 / 25.87
	BTMF	DCRNN	9.11 / 9.55 / 10.71	8.99 / 9.57 / 11.04	9.09 / 9.86 / 10.94	9.45 / 10.22 / 11.43
		ASTGCN	7.83 / 8.24 / 9.44	7.90 / 8.46 / 9.39	7.98 / 8.61 /9.70	8.38 / 8.92 / 9.61
		GMAN	7.99 / 9.31 / 9.80	8.09 / 8.63 / 9.32	8.29 / 8.78 / 9.93	8.56 / 8.95 / 9.75
		FC-GAGA	24.12 / 24.10 / 24.21	24.08 / 24.15 / 24.19	23.85 / 24.08 / 24.11	24.09 / 24.12 / 24.21
	BTMF (50)	DCRNN	7.01 / 8.09 / 10.04	7.14 / 8.28 / 10.28	7.41 / 8.48 / 10.40	7.70 / 8.72 / 10.49
		ASTGCN	5.96 / 7.05 / 8.54	6.04 / 7.24 / 8.76	6.27 / 7.22 / 9.24	**6.62** / 7.54 / 9.11
		GMAN	6.36 / 7.99 / 8.44	6.36 / 7.05 / 8.27	6.40 / 7.15 / 9.33	6.99 / **7.53** / 8.73
		FC-GAGA	23.96 / 24.10 / 24.21	23.98 / 24.04 / 24.10	23.88 / 24.17 / 24.09	23.59 / 23.77 / 23.90
	GCBRNN		8.40 / 8.66 / 9.02	9.22 / 9.85 / 10.22	9.91 / 10.11 / 10.83	12.09 / 12.34 / 12.40
	RIHGCN		7.10 / 7.92 / 9.19	7.36 / 8.13 / 9.36	7.61 / 8.57 / 9.79	9.27 / 9.76 / 11.63
	GSTAE		**5.70 / 6.71 / 7.71**	**5.88 / 6.76 / 7.74**	**6.23 / 6.99 / 7.93**	7.05 / 7.67 / **8.37**
NRM	BRITS	DCRNN	8.91 / 9.84 / 11.65	10.07 / 10.71 / 13.32	12.11 / 12.75 / 14.37	15.13 / 15.42 / 17.16
		ASTGCN	6.98 / 8.52 / 10.26	7.92 / 9.34 / 11.14	9.21 / 10.56 / 11.98	10.87 / 11.40 / 13.25
		GMAN	29.31 / 29.03 / 29.23	29.34 / 29.26 / 28.87	29.05 / 29.33 / 29.18	28.96 / 29.09 / 29.25
		FC-GAGA	23.84 / 23.94 / 23.99	24.33 / 24.42 / 24.58	24.56 / 24.62 / 24.67	25.72 / 25.71 / 25.69
	BTMF	DCRNN	9.10 / 9.70 / 10.76	9.00 / 9.83 / 10.90	9.32 / 9.93 / 11.18	10.15 / 10.65 / 11.95
		ASTGCN	7.82 / 8.38 / 9.51	8.03 / 8.66 / 9.58	8.36 / 8.56 / 9.49	8.76 / 9.28 / 10.60
		GMAN	8.12 / 8.55 / 9.52	8.36 / 8.94 / 9.52	8.56 / 8.96 / 9.79	9.07 / 10.03 / 10.21
		FC-GAGA	24.09 / 24.05 / 24.03	24.10 / 24.11 / 24.13	24.02 / 24.00 / 24.11	24.05 / 24.05 / 24.08
	BTMF (50)	DCRNN	7.07 / 8.25 / 10.27	7.42 / 8.56 / 10.32	7.74 / 8.97 / 10.61	9.09 / 10.10 / 11.79
		ASTGCN	6.02 / 7.07 / 8.85	6.33 / 7.21 / 9.07	6.68 / 7.55 / 9.20	7.72 / 8.39 / 9.81
		GMAN	6.15 / 7.24 / 8.72	7.21 / 7.36 / 8.50	6.95 / 7.99 / 8.92	7.77 / 8.39 / 9.30
		FC-GAGA	23.52 / 23.64 / 23.83	23.92 / 24.01 / 24.07	23.77 / 23.92 / 24.04	24.11 / 24.18 / 24.10
	GCBRNN		8.90 / 9.33 / 10.11	9.31 / 9.95 / 10.38	9.79 / 10.18 / 10.51	10.98 / 10.87 / 11.85
	RIHGCN		7.11 / 8.02 / 9.39	8.36 / 8.27 / 9.61	9.26 / 9.81 / 10.18	9.61 / 10.98 / 12.15
	GSTAE		**5.80 / 6.66 / 7.68**	**6.22 / 6.93 / 8.04**	**6.67 / 7.35 / 8.23**	**7.62 / 8.24 / 8.88**

The baseline model BRITS+GMAN has obvious overfitting problems.

C. Imputation Accuracy

In addition, we compare the imputation performance of the proposed GSTAE with the imputation baselines mentioned in the previous section. Table III shows the comparison result on PEMSD7 and METR-LA datasets. Generally, the proposed GSTAE outperforms other imputation baseline models. This is because the GCN module in ST-Block and bidirectional ST-Block structure benefits from mining spatio-temporal correlations from traffic data with missing values. Compared to BTMF, the performance of BTMF (50) is better. When the missing rate is small (from 20% to 60%), the imputation performance of BRITS is better than BTMF and BTMF (50). When the missing rate is 80%, the performance of BRITS is inferior to that of BTMF and BTMF (50). Because of the complexity of the model optimization objective, the performance of GCBRNN and RIHGCN is inferior to other baselines, which only focus on the imputation task. From the perspective of imputation performance changing trend, as the missing rate increases, BRITS deteriorates rapidly, while the average performance of BTMF-driven models declines slowly. This is also consistent with the comparison results of the traffic prediction task on the PEMSD7 dataset in Table I.

This is due to the different implementation principles of different models. The BRITS imputes data based on a bidirectional recurrent network, which means that BRITS can only remember hidden features for a period of time. With the missing rate increasing, the complexity of BRITS in extracting spatio-temporal correlation information and imputing accurate values will increase significantly. Compared with BRITS, the proposed GSTAE is advantageous in processing traffic data because of the proposed GCN module in ST-Block. For the BTMF method, it performs matrix decomposition and reconstruction of the entire data for each iteration. Therefore, compared to BRITS, BTMF is easier to grasp the overall characteristics of the data but more time-consuming.

In addition to simply comparing the performance of the models on the imputation task, we also investigate the relationship between imputation and prediction. If we only pay attention to the imputation and prediction results of the PEMSD7 dataset, we can easily conclude that the prediction results obtained based on the data imputed by the well-performing imputation model have to be accurate. This is also consistent with our basic cognition.

TABLE III
IMPUTATION PERFORMANCE OF DIFFERENT MISSING RATE WITH RM MISSING TYPE. (MAE / MAPE (%) / RMSE)

Dataset	Imputation Model	Missing Rate (%)			
		20	40	60	80
PEMSD7	BRITS	1.49 / 3.87% / 3.09	1.80 / 4.79% / 3.64	2.20 / 6.00% / 4.37	3.55 / 10.45% / 7.16
	BTMF	3.50 / 8.65% / 5.80	3.51 / 8.70% / 5.82	3.54 / 8.73% / 5.84	3.61 / 8.93% / 5.94
	BTMF (50)	2.50 / 5.75% / 3.92	2.55 / 5.88% / 4.01	2.61 / 6.04% / 4.13	2.86 / 6.70% / 4.57
	GCBRNN	3.38 / 8.65% / 6.02	3.60 / 9.22% / 6.42	3.86 / 10.10% / 6.90	4.57 / 12.34% / 8.16
	RIHGCN	2.25 / 6.32% / 5.15	2.36 / 6.38% / 5.36	2.76 / 7.43% / 5.96	3.51 / 9.53% / 7.03
	GSTAE	**1.09 / 2.41% / 1.90**	**1.22 / 2.73% / 2.19**	**1.44 / 3.25% / 2.71**	**1.92 / 4.53% / 3.84**
METR-LA	BRITS	2.27 / 5.54% / 4.24	2.46 / 6.34% / 4.79	2.84 / 7.54% / 5.62	3.84 / 11.09% / 7.67
	BTMF	3.65 / 9.62% / 6.03	3.70 / 9.69% / 6.04	3.72 / 9.81% / 6.10	3.81 / 10.03% / 6.21
	BTMF (50)	2.61 / 6.25% / 4.14	2.66 / 6.40% / 4.23	2.75 / 6.66% / 4.39	2.97 / 7.30% / 4.76
	GCBRNN	4.56 / 10.08% / 8.70	4.78 / 11.14% / 8.80	4.77 / 12.04% / 8.76	4.88 / 12.86% / 8.83
	RIHGCN	2.88 / 7.93% / 6.01	3.03 / 8.30% / 6.30	3.31 / 9.11% / 6.81	3.82 / 11.14% / 7.72
	GSTAE	**2.01 / 4.79% / 3.53**	**2.03 / 4.86% / 3.60**	**2.14 / 5.17% / 3.88**	**2.38 / 5.99% / 4.55**

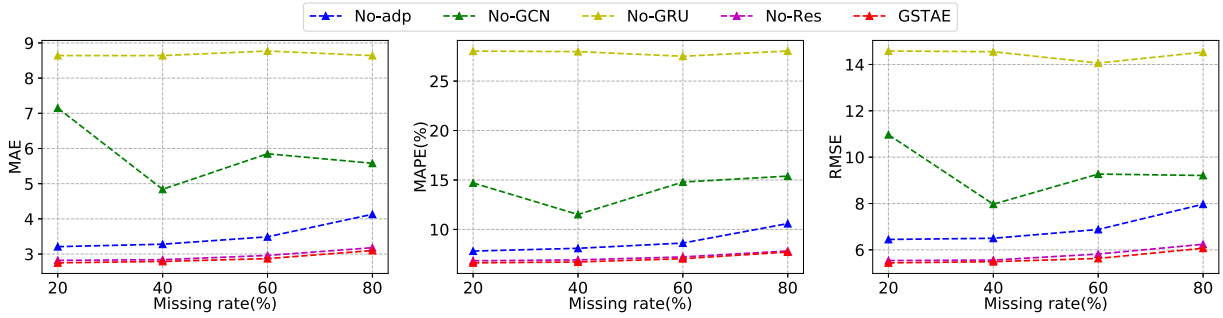


Fig. 5. Performance comparison of ablation models on PEMSD7 dataset for 60 min-ahead prediction with RM missing type.

However, when we pay attention to the prediction results of the METR-LA dataset, we will find that it conflicts with the conclusions we got before. When the missing rate is small, since the imputation effect of BRITS is better, the prediction performance of the baseline models based on BRITS should also be better. However, we found that the performance of baseline models based on BRITS is worse than the baseline models based on BTMF (50). Therefore, we can conclude that the prediction results obtained based on the data imputed by the well-performing imputation model are not necessarily good. In other words, we cannot simply splice an existing model with the best imputation performance and another model with the best prediction performance to assemble a new model and subjectively think it can handle the traffic speed prediction task with missing values.

D. Ablation Test

In this study, we conduct ablation tests to verify the contribution of each GSTAE sub-module to the overall performance. The following four variants of the GSTAE are investigated:

- No-adp: For graph convolution layers, the self-adaptive adjacent matrix $\tilde{A}_{\text{adp}}$ is removed.
- No-GCN: For graph convolution layers, the self-adaptive adjacent matrix $\tilde{A}_{\text{adp}}$ is removed and the widely geography-defined matrix A_{def} is replaced with an identity matrix.
- No-GRU: The gated recurrent units is replaced with linear layers.
- No-Res: The residual connections in the ST-Blocks is removed.

For a fair comparison, all variants are trained with the same setting as introduced in Sec. V-A. The simulations are conducted on the PEMSD7 dataset for 60 min-ahead prediction with RM missing type.

Fig. 5 shows the prediction performance comparison of GSTAE with its variants. To summarize, the proposed GSTAE has the best performance compared to the other variants over all missing rates, demonstrating each sub-module's contribution. Among the four variants, both GRU units and GCN layers contribute the most to the accuracy, indicating that extracting spatio-temporal relationships plays a crucial role in the traffic prediction task. Besides, thanks to the aid of the self-adaptive adjacency matrix, the model can learn additional spatial dependence and further enhance its performance. Lastly, residual connections in ST-blocks have the most negligible impact on the accuracy since residual connections are designed to reduce computational costs in training and prevent the model from overfitting. Compared to the No-Res model, the average computation time per epoch of the proposed GSTAE model decreases from 49.71 s to 43.84 s.

Furthermore, through the two-stage training paradigm, the proposed GSTAE can achieve a faster training process and convergence than training the prediction task directly (e.g., the average training time per epoch decreases from 92.65 s to 43.84 s, and the average number of training iterations required for GSTAE to converge decreases from 102 to 40).

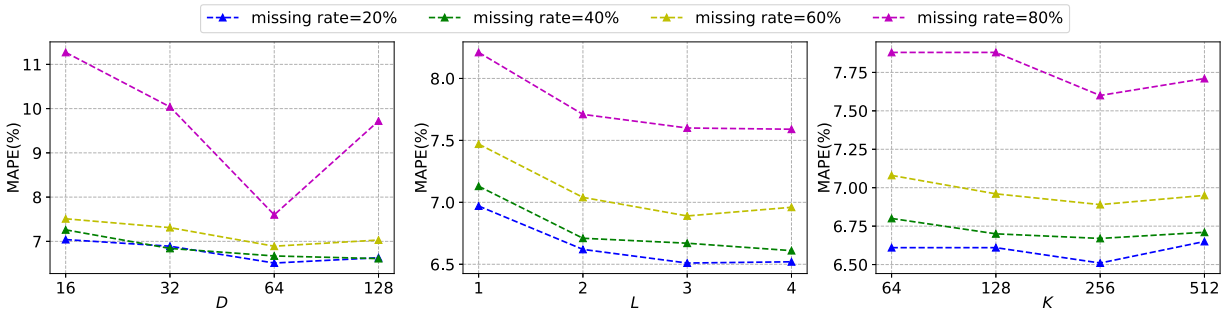


Fig. 6. Performance comparison of different hyperparameters on PEMSD7 dataset for 60 min-ahead prediction with RM missing type.

This is because the pre-trained encoder module can extract dense representation from traffic data through the previous imputation training process, which subsequently accelerates the subsequent prediction training process.

E. Hyperparameter Test

The hyperparameters of the proposed GSTAE are empirically set, since they are critical to the model performance. In this section, we investigate the sensitivity and influence of three important hyperparameters, i.e., the dimension of ST-Block D, the number of stacked ST-Blocks L, and the hidden dimension of the fully connected layers K. Specifically, these hyperparameters are set to $D \in \{16, 32, 64, 128\}$, $L \in \{1, 2, 3, 4\}$, and $K \in \{64, 128, 256, 512\}$ for evaluation, respectively. Additionally, all the simulations are conducted with RM missing type and 60 min-ahead prediction on the PEMSD7 dataset.

As shown in Fig. 6, we have the following observations. First, three hyperparameters generally show the same trend: increasing the values improves prediction accuracy. The reason is that enlarging model parameters can improve the learning capacity of latent features (e.g., D increases from 16 to 64). However, overly large model parameters may lead to overfitting and result in performance degradation (e.g., D increases from 64 to 128). Taking the number of stacked ST-Blocks L as an example, stacking ST-Blocks deepens each recurrent submodule with GCN, which means that the range of information obtained by each node based on the network topology is enlarged. As a result, each node can aggregate more information, and the model accordingly makes more accurate speed predictions. However, if too many GCN modules are stacked, it increases the difficulty for each node to distinguish the importance of various neighbor nodes (e.g., which nodes are closer to itself and which are far away). Furthermore, the excessive GCNs offset the critical function of the GCN module in extracting local information. Second, D has the most significant impact on prediction accuracy among the three hyperparameters, with MAPE fluctuating from approx. 11% to approx. 8%. This is consistent with the ablation results in Sec. V-D that GCN layers and GRUs in ST-Blocks have a more significant impact on prediction accuracy.

VI. Conclusion

In this paper, we propose a new multi-task learning deep neural network model GSTAE that follows an encoder-decoder structure to handle the task of traffic speed prediction with missing values and eliminate the error accumulation issues during the imputation and prediction process. Specifically, the model consists of multiple ST-Block modules that combine GCN layer with an adaptive adjacency matrix for spatial modeling and GRU unit for temporal learning. We treat the imputation and prediction as two parallel tasks rather than standalone and consecutive tasks so as to eliminate the error accumulation issue from performing imputation and prediction sequentially. Additionally, we design a two-stage training paradigm to accelerate the prediction training process and improve performance. Concretely, the imputation task is trained first to make the encoder extract dense representation from the input with missing values. We subsequently train the task of traffic speed prediction with missing values based on the pre-trained encoder.

To evaluate the performance of the proposed GSTAE, we conduct comprehensive experiments on two real-world traffic datasets with various missing scenarios, i.e., two different missing patterns and a wide missing rate range from 20% to 80%. Compared to the state-of-the-art traffic prediction with missing values methods, the proposed model shows superiority and stable performance. In addition, the proposed GSTAE also shows the state-of-the-art performance on imputation task as a side effect. Simulation results on the imputation task show that an accurate imputation result does not necessarily positively impact prediction, which falsifies the common assumption in the literature.

Through the two-stage training paradigm, the proposed multi-task learning deep neural network model GSTAE achieves superiority and stable performance on both sub-tasks, namely, traffic data imputation and traffic prediction with missing values. However, since GRU modules process data iteratively and can not be parallelized, a great computational effort is needed when adopting the proposed model on large-scale datasets and predicting long-term traffic speed. In the future, we will accelerate the training process by improving the model structure using modules like CNN and Temporal Convolution layer (TCN). Besides, long-term traffic prediction with missing values is also a direction worth studying.

References

[1] M. Veres and M. Moussa, "Deep learning for intelligent transportation systems: A survey of emerging trends," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3152–3168, Aug. 2019.

[2] S. Lee and D. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1678, pp. 179–188, Jan. 1999, doi: 10.3141/1678-22.

[3] J. Guo, W. Huang, and B. M. Williams, "Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 50–64, Jun. 2014.

[4] H. Yu, N. Ji, Y. Ren, and C. Yang, "A special event-based K-nearest neighbor model for short-term traffic state prediction," *IEEE Access*, vol. 7, pp. 81717–81729, 2019.

[5] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 276–281, Dec. 2004.

[6] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "DNN-based prediction model for spatio-temporal data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, New York, NY, USA, Oct. 2016, pp. 1–4, doi: 10.1145/2996913.2997016.

[7] C. Zhang, J. J. Q. Yu, and Y. Liu, "Spatial–temporal graph attention networks: A deep learning approach for traffic forecasting," *IEEE Access*, vol. 7, pp. 166246–166256, 2019.

[8] J. J. Q. Yu and J. Gu, "Real-time traffic speed estimation with graph convolutional generative autoencoder," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 10, pp. 3940–3951, Oct. 2019.

[9] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, Vancouver, BC, Canada, 2018, pp. 1–16.

[10] W. Jiang and J. Luo, "Graph neural network for traffic forecasting: A survey," 2021, *arXiv:2101.11174*.

[11] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," 2014, *arXiv:1409.2329*.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[13] J. Chung, C. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," 2014, *arXiv:1412.3555*.

[14] X. Meng et al., "D-LSTM: Short-term road traffic speed prediction model based on GPS positioning data," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 3, pp. 2021–2030, Mar. 2022.

[15] Y. Zhu, M. Zhang, B. Jiang, H. Jin, J. Huang, and X. Wang, "Networked time series prediction with incomplete data," 2021, *arXiv:2110.02271*.

[16] Z. Zhang, X. Lin, M. Li, and Y. Wang, "A customized deep learning approach to integrate network-scale online traffic data imputation and prediction," *Transp. Res. C, Emerg. Technol.*, vol. 132, Nov. 2021, Art. no. 103372.

[17] W. Zhong, Q. Suo, X. Jia, A. Zhang, and L. Su, "Heterogeneous spatio-temporal graph convolution network for traffic forecasting with missing values," in *Proc. IEEE 41st Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Washington, DC, USA, Jul. 2021, pp. 707–717.

[18] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial–temporal graph convolutional networks for traffic flow forecasting," in *Proc. 33rd AAAI Conf. Artif. Intell., (AAAI)*. Honolulu, HI, USA: AAAI Press, 2019, pp. 922–929, doi: 10.1609/aaai.v33i01.3301922.

[19] Z. Cui, L. Lin, Z. Pu, and Y. Wang, "Graph Markov network for traffic forecasting with missing data," *Transp. Res. C, Emerg. Technol.*, vol. 117, Aug. 2020, Art. no. 102671.

[20] H. Tan, G. Feng, J. Feng, W. Wang, Y.-J. Zhang, and F. Li, "A tensor-based method for missing traffic data completion," *Transp. Res. C, Emerg. Technol.*, vol. 28, pp. 15–27, Mar. 2013.

[21] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," 2016, *arXiv:1606.01865*.

[22] X. Chen and L. Sun, "Bayesian temporal factorization for multidimensional time series prediction," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 9, pp. 4659–4673, Sep. 2022.

[23] X. Chen, Z. He, and L. Sun, "A Bayesian tensor decomposition approach for spatiotemporal traffic data imputation," *Transp. Res. C, Emerg. Technol.*, vol. 98, pp. 73–84, Jan. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X1830799X

[24] Z. Cui, R. Ke, Z. Pu, and Y. Wang, "Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values," *Transp. Res. C, Emerg. Technol.*, vol. 118, Sep. 2020, Art. no. 102674.

[25] S. Ruder, "An overview of multi-task learning in deep neural networks," 2017, *arXiv:1706.05098*.

[26] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2014, *arXiv:1312.6203*.

[27] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 17804–17815.

[28] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. 34th AAAI Conf. Artif. Intell., (AAAI)*. New York, NY, USA: AAAI Press, 2020, pp. 1234–1241. [Online]. Available: https://aaai.org/ojs/index.php/AAAI/article/view/5477

[29] W. Cao, D. Wang, J. Li, H. Zhou, L. Li, and Y. Li, "BRITS: Bidirectional recurrent imputation for time series," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31. Montreal, QC, Canada: Curran Associates, 2018, pp. 6776–6786.

[30] J. Yoon, J. Jordon, and M. Van Der Schaar, "GAIN: Missing data imputation using generative adversarial nets," in *Proc. 35th Int. Conf. Mach. Learn. (ICML)*, J. G. Dy and A. Krause, Eds. Stockholm, Sweden: PMLR, 2018, pp. 5675–5684.

[31] Y. Luo, Y. Zhang, X. Cai, and X. Yuan, "E^2GAN: End-to-end generative adversarial network for multivariate time series imputation," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Macao, China, Aug. 2019, pp. 3094–3100.

[32] Y. Wang, Y. Zhang, X.-L. Piao, H. Liu, and K. Zhang, "Traffic data reconstruction via adaptive spatial–temporal correlations," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 4, pp. 1531–1543, Aug. 2018.

[33] Z. Zhang, M. Li, X. Lin, and Y. Wang, "Network-wide traffic flow estimation with insufficient volume detection and crowdsourcing data," *Transp. Res. C, Emerg. Technol.*, vol. 121, Dec. 2020, Art. no. 102870.

[34] O. Benkraouda, B. T. Thodi, H. Yeo, M. Menéndez, and S. E. Jabari, "Traffic data imputation using deep convolutional neural networks," *IEEE Access*, vol. 8, pp. 104740–104752, 2020.

[35] Y. Zhuang, R. Ke, and Y. Wang, "Innovative method for traffic data imputation based on convolutional neural network," *IET Intell. Transp. Syst.*, vol. 13, no. 4, pp. 605–613, Apr. 2019.

[36] Y. Ye, S. Zhang, and J. J. Q. Yu, "Spatial–temporal traffic data imputation via graph attention convolutional network," in *Proc. Artif. Neural Netw. Mach. Learn. (ICANN)*, vol. 12891. Bratislava, Slovakia: Springer, 2021, pp. 241–252.

[37] Y. Wu, D. Zhuang, A. Labbe, and L. Sun, "Inductive graph neural networks for spatiotemporal Kriging," in *Proc. 35th AAAI Conf. Artif. Intell.*, vol. 35, no. 5, 2021, pp. 4478–4485.

[38] O. Sener and V. Koltun, "Multi-task learning as multi-objective optimization," in *Proc. Annu. Conf. Neural Inf. Process. Syst.*, S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Montreal, QC, Canada: Curran Associates, 2018, pp. 525–536.

[39] M. Luong, Q. V. Le, I. Sutskever, O. Vinyals, and L. Kaiser, "Multi-task sequence to sequence learning," in *Proc. 4th Int. Conf. Learn. Represent. (ICLR)*, Y. Bengio and Y. LeCun, Eds. San Juan, Puerto Rico: OpenReview.net, May 2016, pp. 1–10.

[40] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, J. Lang, Ed. Stockholm, Sweden: International Joint Conferences on Artificial Intelligence Organization, Jul. 2018, pp. 3634–3640, doi: 10.24963/ijcai.2018/505.

[41] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, Montreal, QC, Canada, Sep. 2014, pp. 3104–3112.

[42] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, "A transformer-based framework for multivariate time series representation learning," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, New York, NY, USA, Aug. 2021, pp. 2114–2124.

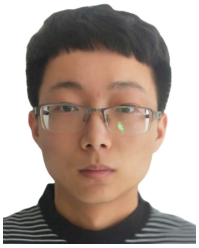
[43] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, 2008, pp. 1096–1103, doi: 10.1145/1390156.1390294.

[44] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, pp. 533–536, Oct. 1986, doi: 10.1038/323533a0.

[45] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, Toulon, France, Apr. 2017, pp. 1–14.

[46] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. NIPS*, Barcelona, Spain, vol. 29, 2016, pp. 3837–3845.

[47] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial–temporal graph modeling," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, S. Kraus, Ed. Macao, China: International Joint Conferences on Artificial Intelligence Organization, Aug. 2019, pp. 1907–1913.
[48] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *Proc. 36th Int. Conf. Mach. Learn.*, Long Beach, CA, USA, vol. 97, Jun. 2019, pp. 6861–6871.
[49] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Mar. 2020.
[50] J. J. Q. Yu, C. Markos, and S. Zhang, "Long-term urban traffic speed prediction with deep learning on graphs," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 20, pp. 1–12, Apr. 2021.
[51] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[52] B. N. Oreshkin, A. Amini, L. Coyle, and M. J. Coates, "FC-GAGA: Fully connected gated graph architecture for spatio-temporal traffic forecasting," in *Proc. AAAI*, 2021, pp. 9233–9241.

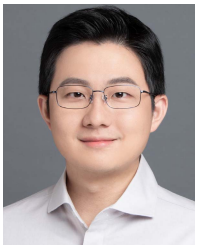


Ao Wang received the B.Eng. degree in computer science and technology from the Southern University of Science and Technology, Shenzhen, China, in 2021, where he is currently pursuing the master's degree with the Department of Computer Science and Engineering. His research interests include smart city, intelligent transportation systems, and deep learning.

Yongchao Ye (Student Member, IEEE) received the B.Eng. degree in computer science and technology from Ningbo University, Ningbo, China, in 2020. He is currently pursuing the master's degree with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China. His research interests include spatio-temporal data mining in smart city, intelligent transportation systems, and federated learning.

Xiaozhuang Song received the B.Eng. degree in digital media technology from the Beijing University of Posts and Telecommunications, Beijing, China, in 2015, and the M.S. degree in computer science and technology from the Southern University of Science and Technology, Shenzhen, China, in 2022. His research interests include graph neural networks and intelligent transportation systems.

Shiyao Zhang (Member, IEEE) received the B.S. degree (Hons.) in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2014, the M.S. degree in electrical engineering from the University of Southern California, Los Angeles, CA, USA, in 2016, and the Ph.D. degree from The University of Hong Kong, Hong Kong. He was a Post-Doctoral Research Fellow with the Academy for Advanced Interdisciplinary Studies, Southern University of Science and Technology, from 2020 to 2022, where he is currently a Research Assistant Professor with the Research Institute for Trustworthy Autonomous Systems. His research interests include smart cities, smart energy systems, intelligent transportation systems, optimization theory and algorithms, and deep learning applications.



James J. Q. Yu (Senior Member, IEEE) received the B.Eng. and Ph.D. degrees in electrical and electronic engineering from The University of Hong Kong, Hong Kong, in 2011 and 2015, respectively. He was a Post-Doctoral Fellow with The University of Hong Kong from 2015 to 2018. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China, and an Honorary Assistant Professor with the Department of Electrical and Electronic Engineering, The University of Hong Kong. His research is mainly on forecasting and decision making of future transportation systems and basic artificial intelligence techniques for industrial applications. His research interests include smart city and urban computing, deep learning, intelligent transportation systems, and smart energy systems. He was ranked World's Top 2% Scientists of 2019 and 2020 by Stanford University. He is an Editor of the *IET Smart Cities* journal.