Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 21(4): 4989–5006.
DOI: 10.3934/mbe.2024220
Received: 06 November 2023
Revised: 20 January 2024
Accepted: 24 January 2024
Published: 01 March 2024

Research article

Cross-modal missing time-series imputation using dense spatio-temporal transformer nets

Xusheng Qian[1], Teng Zhang[1], Meng Miao[1], Gaojun Xu[1], Xuancheng Zhang[1], Wenwu Yu[2,*]and Duxin Chen[2]

[1] State Grid Jiangsu Electric Power Company Limited Marketing Service Center, Nanjing 210019, China

[2] Jiangsu Key Laboratory of Networked Collective Intelligence, School of Mathematics, Southeast University, Nanjing 211189, China

* **Correspondence:** Email: wwyu@seu.edu.cn.

Abstract: Due to irregular sampling or device failure, the data collected from sensor network has missing value, that is, missing time-series data occurs. To address this issue, many methods have been proposed to impute random or non-random missing data. However, the imputation accuracy of these methods are not accurate enough to be applied, especially in the case of complete data missing (CDM). Thus, we propose a cross-modal method to impute time-series missing data by dense spatio-temporal transformer nets (DSTTN). This model embeds spatial modal data into time-series data by stacked spatio-temporal transformer blocks and deployment of dense connections. It adopts cross-modal constraints, a graph Laplacian regularization term, to optimize model parameters. When the model is trained, it recovers missing data finally by an end-to-end imputation pipeline. Various baseline models are compared by sufficient experiments. Based on the experimental results, it is verified that DSTTN achieves state-of-the-art imputation performance in the cases of random and non-random missing. Especially, the proposed method provides a new solution to the CDM problem.

Keywords: time-series data missing; complete data missing; time-series data imputation; cross-modal data fusion; dense spatio-temporal transformer nets

1. Introduction

Due to irregular sampling or device failure, the data collected from a sensor device may not be completed, that is, missing values are included in the data. High-quality data enables many data-driven networked control techniques to perform better [1–5]. The missing data issue can be categorized as missing at random (MAR) and missing not at random (MNAR) [6–9]. Among them, complete data missing (CDM) is a special but critical case of MNAR [10]. That is, the missing rates of some network nodes reach 100%. The bottleneck of CDM is that the traffic data of some nodes is unobserved completely, and no time-series data can be used for feature extraction [11].

To address the time-series data missing issue, many methods have been proposed, including weighted-average methods, tensor-based methods, deep-learning methods, and multi-view methods. Among them, weighted-average methods heavily depend on the definition of weights, such as the k-nearest neighbors (KNN) model [12]. The imputation results of these methods are often too rough to use because the weighted average principle are linear. This is not in line with reality. More popular are tensor-based methods. For example, the Bayesian Gaussian CANDECOMP/PARAFAC (BGCP) model [13] is established with the low-rank assumption [14]. It imputes missing data by processing global tensor information and reconstructing sparse tensors though Bayesian inference. However, the usage of spatial information in most tensor-based methods, including [13, 15, 16], are limited in the temporary modal tensor structure. They ignore other spatial modals, such as the locations of sensor network nodes. Thus, the recovered missing data of these methods can hardly match the corresponding nodes in some extreme cases, especially in the CDM case. Furthermore, deep-learning methods provide an alternative way to impute missing data [17–20]. They obtain good performance based on sufficient observation. But, due to the heavy dependence on the large number of training samples required, they often fall into the overfitting problem, and then the performance declines dramatically if the missing rate is too high. By contrast, multi-view methods impute missing data by shared latent representation among different views for one observational object [7, 8]. They rely heavily on the assumptions of view consistency and instance completeness. But, unfortunately, these two assumptions are violated in the CDM problem because the data missing for different views are unbalanced. In the so-called spatial view [21], the missing rate is low because only a few sensor stations are unobservable. But, in the temporal view, the missing rate for these unobservable stations reach 100% because no time-series data can be collected.

Thus, to improve the imputation performance for high missing rates, especially for the CDM issue, this paper proposes a cross-modal missing time-series data imputation method. This method imputes missing data based on a cross-modal principle [22, 23], that is, to use spatial modal data to recover temporal modal data. From the perspective of complex network, it is found that the spatial locations collected from GPS data of sensors are correlated to the evolution of traffic flow [24]. Thus, if the time-series traffic data is missing, it can be recovered from the spatial modal data collected from not only GPS, but also social media, etc. This approach to address missing data is called spatio-temporal cross modal. Based on this, a DSTTN model is specially designed to fuse the spatio-temporal modal information with stacked fusion modules with dense connections. DSTTN enables different modals to be processed together, and output the time-series missing data in an end-to-end way. Moreover, a graph Laplacian regularization constraint is added to the training target function. The constraint enables the spatial modal data to be extrapolated to the missing time-series data, which is an unseen domain if the CDM issue happens.

Thus, the main contributions of this work are as follows:

1) We propose a cross-modal method to solve the time-series missing data problem. This method uses a graph Laplacian regularization constraint to model complex correlations between spatial and temporal modals. It provides a way to extrapolate observation data to a completely unseen data domain, and finally solves the CDM problem.

2) We propose a DSTTN model to fuse spatial and temporal modal data for the realization of a cross-modal method. This model is organized with multiple stackable spatio-temporal transformer blocks, while dense connections are designed to improve communication between these blocks and accelerate model training.
3) By conducting comparative experiments, the state-of-the-art imputation performance of the cross-modal method is verified. The essential analysis is presented to demonstrate the principle of the cross-modal method to solve the CDM problem.

The rest of this paper is organized as follows: Section 2 presents the reviews of related works on time-series data missing problem. Section 3 presents the cross-modal method and the DSTTN model. Section 4 analyzes the results of comparative experiments. Section 5 concludes this paper.

2. Related work

As early as the 1990s, the autoRegressive integrated moving average (ARIMA) model was proposed to predict the trend of short-term freeway flow data, and impute time-series missing data [25, 26]. In 2008, probabilistic principle components analysis (PPCA) was introduced to impute missing values by compressing damaged data and minimizing reconstruction error [27]. It modeled the uncertainty of time series and extracted low-dimensional periodic features from time-series data by singular value decomposition. Based on PPCA model, a kernel PPCA model [28] was further developed to extract nonlinear features by the kernel trick, improving the accuracy of imputation. Furthermore, some non-parametric methods were also proposed to reconstruct damaged data by the weighted average principle, such as KNN [12], fuzzy C-means method [29], and OOC-based Kriging estimation [30]. These calculate missing values by weighted observation features of k-nearest neighbors.

Similarly, based on the principle of data reconstruction, Tan et al. [15] proposed a tensor-based method for imputation of incomplete time-series data. It focused on the preserved spatio-temporal information in multi-way nature of matrix pattern, and achieved good performance while the missing data rate is high. To further exploit the temporal characteristics and the spatial similarity between road links, Wang et al. [31] proposed a low-rank matrix decomposition method to reconstruct damaged time-series via adaptive Laplacian regularization spatial constraint. Chen et al. [32] proposed an SVD-combined tensor decomposition method to capture the main latent features for missing data estimation, and then proposed a Bayesian tensor decomposition approach, called BGCP [13], to achieve great performance. To improve the local consistency in spatio-temporal data, Chen et al. [16] also proposed a low-rank autoregressive tensor completion (LATC) model that introduces temporal variation as a constraint into the completion of third-order tensor. However, the concepts of space and time in these methods are only limited to the spatio-temporal information, but do not consider multimodal labels, such as the location, lane type, and station length of the traffic stations in an intelligent transportation system.

In addition to these tensor-based methods, deep learning technology has attracted more and more academic and industrial interest recently. Based on deep learning, Duan et al. [17] proposed a denoising stacked autoencoder (DSAE) to impute missing data. It considered the missing value as outliers, and used the damaged data as input directly. Then, the missing value was filtered by layer-wise encoding and decoding, which replaced the missing value with the estimated ones. Another popular method was long short-term memory (LSTM) nets [18]. These also modeled the uncertainty of time-series pattern dependency, and realized an end-to-end training process and imputation. Cui et al. [33] also proposed a bidirectional LSTM (BDLSTM) with imputation mechanism and verified that the imputed data improved the accuracy of the LSTM prediction model. Kong et al. [34] proposed a dynamic graph convolutional recurrent imputation network (DGCRIN) to impute missing time-series data with static and dynamic graph features together. Based on this, a graph attention recurrent neural network (GARNN) [35] was also proposed to address two special cases of time-series data missing at random, including temporally continuous missing and spatially continuous missing. Moreover, Chen et al. [19] proposed a time-series imputation method using generative adversarial nets (GAN). Zhang et al. [36] also proposed a GAN-based imputation model, which utilized self-attention technology to extract spatial features. Kang et al. [10, 37] proposed a cross-modal GAN to address CDM issues on spatio-temporal data, and then used the cross-modal GAN to generate scenarios data for dispatching application with missing data.

Furthermore, multi-view methods were also proposed to impute missing values with multi-view information of sensors [7, 8]. Li et al. [38] proposed a hybrid imputation method combining LSTM and a collaborative filtering technique by multi-view learning. To improve the imputation accuracy, Yao et al. [21] proposed a multi-view spatio-temporal graph network (MVSTGN), which combined attention and convolution mechanisms to apply for spatio-temporal pattern analysis. The multi-view design of MVSTGN, however, is only for spatial and temporal characteristics of time-series data. Some other modal data, such as GPS locations of sensors, were not considered.

Overall, among them, weighted-average methods, e.g., KNN and fuzzy C-means, are relatively not robust. Their imputations often dramatically change with the input arguments K or C. ARIMA and PCA-based methods are all regression-based methods. Thus, they need to minimize the residual between true time-series values and the imputed ones. However, due to their linearity, that is, they only use a linear model to fit the observation data, the model representation of the data is not enough. By contrast, deep-learning methods have excellent ability to represent nonlinearity in data. However, the time complexity of deep-learning method is higher than ARIMA and the other PCA-based methods if there are to many neurons in neural network. Thus, if time consumption can be ignored, deep-learning methods are better on imputation performance.

Under the framework of deep learning, multi-view methods and multi-modal methods are two ways to address the missing data issue, especially for the CDM issue. They both pay attention to data processing, but there are still three main differences, as follows: (1) The multiple views of multi-view methods are concentrated on the only observation object, while the cross-modal method faces towards various modals, not limited to a single object. (2) The spatial and temporal views (e.g., in [21]) are only toward the time-series data, and do not involve more spatial information, such as GPS data. (3) The cross-modal method makes full use of these modal data, including GPS data, to solve the CDM problem. Meanwhile, multi-view can not solve the CDM problem, which is verified though comparative experiments.

3. Methodology

To solve the time-series data missing problem, a cross-modal method is proposed, based on the DSTTN model, as shown in Figure 1. The framework mainly has two parts: (1) The introduction of cross-modal data fusion technique. (2) the introduction of the DSTTN model. The module of cross-

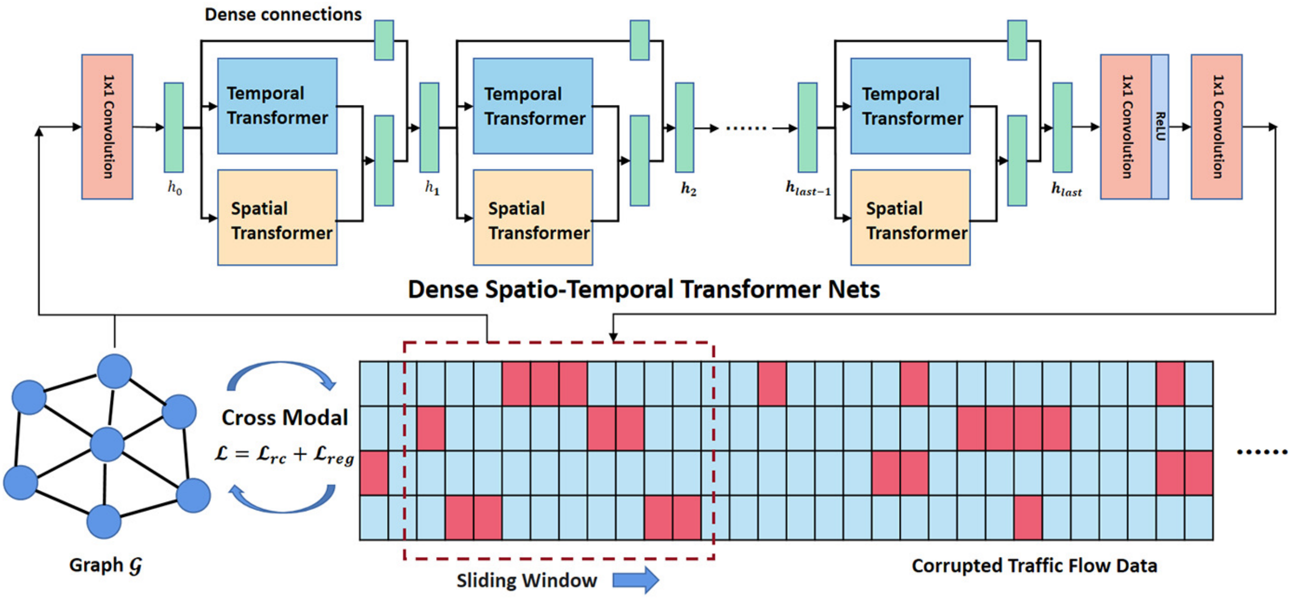

Figure 1. The cross-modal missing time series imputation architecture based on DSTTN.

modal data fusion is specially-designed to address the CDM issue, and DSTTN is used to realize the ability of fitting data. In the following, the implementation and algorithm of this method is presented.

3.1. Cross-modal data fusion

In the real world, the ways to sense the state of a system are different, such as seeing, hearing, and smelling, which are called modals [22]. Though the modals are different, the object to sense is the same one. Thus, these modals can be translated to each other. Here, a cross-modal data fusion technique is proposed to fuse time-series data and graph data, and output the missing time series data. We refer to this as the cross-modal way. Due to the fact that the time-series data and graph data both carry the same information to the same object, the missing value of time series can be recovered from the graph.

Given the time series that are represented as a 2-dimensional tensor $\mathcal{X} = \{x_{it} | i = 1, \ldots, N, t = 1, \ldots, T\}$, here, N is the number of nodes and T is the number of samples depending on the time intervals. Then, if a MAR or MNAR problem happens, a part of the elements in $\mathcal{X}$ are missing and unobservable, and are set as zero. The missing rate can be defined as

$$\eta = \frac{m}{N \times T} \times 100\%, \tag{3.1}$$

where m is the number of missing elements in tensor $\mathcal{X}$.

Furthermore, a graph $\mathcal{G} = (\mathbf{V}, \mathbf{E}, A)$ is known, $\mathbf{V}$ is the set of nodes, and $\mathcal{E}$ is the set of undirected edges. The weights of the adjacency matrix $A \in \mathbb{R}^{N \times N}$, deriving from the graph $\mathcal{G}$, describe the similarity between different nodes. Then, we can define the loss function as

$$\mathcal{L} = \mathcal{L}_{rc} + \lambda \mathcal{L}_{reg}, \tag{3.2}$$

where $\mathcal{L}_{rc}$ is the reconstruction error. It can be defined as

$$\mathcal{L}_{rc} = \frac{1}{CI[\mathbf{I}_{it} = 1]} \sum_{i,t} \mathbf{I}_{it} \|\hat{x}_{it} - x_{it}\|^2, \tag{3.3}$$

where $\mathbf{I}_{it} = 0$ if x_{it} is missing, otherwise $\mathbf{I}_{it} = 1$, and $CI[\mathbf{I}_{it} = 1]$ is a countif function that counts the element number of $\mathbf{I}_{it} = 1$. Then, $\mathcal{L}_{reg}$ is a graph Laplacian regularization term, which makes the imputed value of similar nodes more similar. Thus, the $\mathcal{L}_{reg}$ can be defined as

$$\begin{aligned} \mathcal{L}_{reg} &= \sum_t \sum_{i,j} A_{ij} \|\hat{x}_{it} - \hat{x}_{jt}\|^2 \\ &= \sum_t \hat{x}_{:t} \Delta \hat{x}_{:t}, \end{aligned} \tag{3.4}$$

where $\Delta = D - A$ is the unnormalized graph Laplacian matrix, and $D_{ii} = \sum_j A_{ij}$ is the degree matrix of A. Besides, λ is a penalty factor, and $\hat{\mathcal{X}} = f(\mathcal{X})$ is the estimator of missing data, where f is a mapping between them. When the mapping is fitted, the dynamic flow information and the static graph information are fused at the end, which means the cross-modal data fusion. Then, the remaining work is to define the path to the fitness of mapping f.

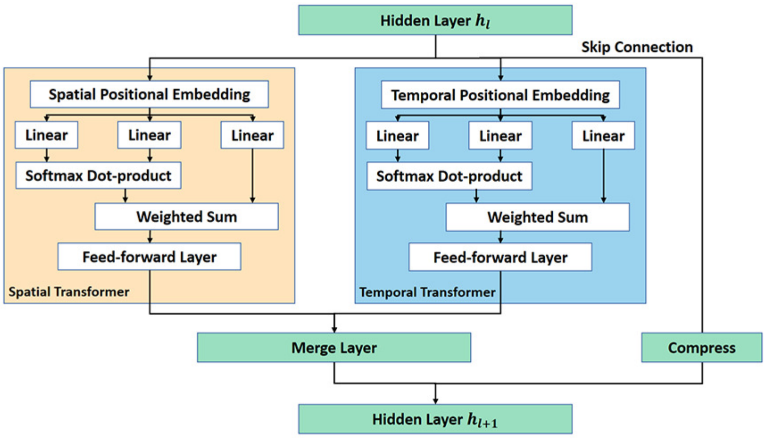

Figure 2. A spatio-temporal transformer block.

3.2. DSTTN

As demonstrated above, a sequential regression model needs to be designed to fit the mapping f. Thus, we design the DSTTN model to complete this issue by an end-to-end missing data imputation pipeline. In DSTTN, the naive transformer structure [39] is redefined as a stackable spatio-temporal transformer block as shown in Figure 2. It is a combination of a spatial transformer, temporal transformer, and skip connection as detailed in the following description:

In the spatial transformer, the initial input X^S is defined as

$$X^S = h_l + E^S, \tag{3.5}$$

where $h_l \in \mathbb{R}^{N\times T\times C}$ is the hidden feature in l-th hidden layer, and $E^{\mathcal{S}} = AW_a^{\mathcal{S}}$, embedding the spatial modal information into positional code by weight matrix $W_a^{\mathcal{S}} \in \mathbb{R}^{N\times C}$. C represents the number of embedding channels, $E^{\mathcal{S}}$ is repeatedly expanded into $\mathbb{R}^{N\times T\times C}$, and next added to h_l. Then, according to the standard transformer model [39], the queries, keys, and values are computed by three linear layers, that is,

$$\begin{aligned} Q^{\mathcal{S}} &= X^{\mathcal{S}}W_q^{\mathcal{S}},\\ K^{\mathcal{S}} &= X^{\mathcal{S}}W_k^{\mathcal{S}},\\ V^{\mathcal{S}} &= X^{\mathcal{S}}W_v^{\mathcal{S}}, \end{aligned} \tag{3.6}$$

where $W_q^{\mathcal{S}}, W_k^{\mathcal{S}}, W_v^{\mathcal{S}} \in \mathbb{R}^{C\times C}$ are the weight matrices for $Q^{\mathcal{S}}, K^{\mathcal{S}}, V^{\mathcal{S}}$, respectively. After that, the dynamical spatial dependencies between different nodes are learned by the softmax dot-product

$$M^{\mathcal{S}} = \text{softmax}(\frac{Q^{\mathcal{S}}(K^{\mathcal{S}})^T}{\sqrt{d_k^{\mathcal{S}}}}), \tag{3.7}$$

where $d_k^{\mathcal{S}}$ is a scaling factor, which equals to the channels C. Note that $Q^{\mathcal{S}}(K^{\mathcal{S}})^T$ is a matrix product at the axis of N and T, which means $M^{\mathcal{S}} \in \mathbb{R}^{N\times N\times C}$. Then, new spatial node features are obtained by weighted sum

$$U^{\mathcal{S}} = M^{\mathcal{S}}V^{\mathcal{S}}, \tag{3.8}$$

where $M^{\mathcal{S}}V^{\mathcal{S}}$ is a matrix product at the axis of N and T, which means $U^{\mathcal{S}} \in \mathbb{R}^{N\times T\times C}$. Then, a simple feed-forward layer is applied to filter feature information as

$$Y^{\mathcal{S}} = \text{ReLU}(\text{ReLU}(U^{\mathcal{S}}W_0^{\mathcal{S}})W_1^{\mathcal{S}})W_2^{\mathcal{S}}, \tag{3.9}$$

where $W_0^{\mathcal{S}}, W_1^{\mathcal{S}}, W_2^{\mathcal{S}} \in \mathbb{R}^{C\times C}$ are the weighted matrices of the three-layer feed-forward neural networks.

The temporal transformer is similar to the spatial transformer, but the positional code is different. Its initial input is defined as

$$X^{\mathcal{T}} = h_l + E^{\mathcal{T}}, \tag{3.10}$$

where $E^{\mathcal{T}} = \text{Embedding}([1,\dots,T])$, and Embedding$(\cdot)$ is a general vector embedding layer. Then, the queries, keys, and values are obtained by

$$\begin{aligned} Q^{\mathcal{T}} &= X^{\mathcal{T}}W_q^{\mathcal{T}},\\ K^{\mathcal{T}} &= X^{\mathcal{T}}W_k^{\mathcal{T}},\\ V^{\mathcal{T}} &= X^{\mathcal{T}}W_v^{\mathcal{T}}, \end{aligned} \tag{3.11}$$

where $W_q^{\mathcal{T}}, W_k^{\mathcal{T}}, W_v^{\mathcal{T}} \in \mathbb{R}^{C\times C}$ are the weight matrices for $Q^{\mathcal{T}}, K^{\mathcal{T}}, V^{\mathcal{T}}$, respectively. Then,

$$M^{\mathcal{T}} = \text{softmax}(\frac{Q^{\mathcal{T}}(K^{\mathcal{T}})^T}{\sqrt{d_k^{\mathcal{T}}}}), \tag{3.12}$$

where $d_k^{\mathcal{T}} = C$, and $(Q^{\mathcal{T}}(K^{\mathcal{T}})^T$ is a matrix product at the axis of N and T, which means $M^{\mathcal{T}} \in \mathbb{R}^{N\times N\times C}$. Then,

$$U^{\mathcal{T}} = M^{\mathcal{T}}V^{\mathcal{T}}, \tag{3.13}$$

where $M^{\mathcal{T}} V^{\mathcal{T}}$ is a matrix product at the axis of N and T, concluding that $M^{\mathcal{T}} \in \mathbb{R}^{N \times T \times C}$. Then, the same feed-forward layer is applied as

$$Y^{\mathcal{T}} = \text{ReLU}(\text{ReLU}(U^{\mathcal{T}} W_0^{\mathcal{T}}) W_1^{\mathcal{T}}) W_2^{\mathcal{T}}, \tag{3.14}$$

where $W_0^{\mathcal{T}}, W_1^{\mathcal{T}}, W_2^{\mathcal{T}} \in \mathbb{R}^{C \times C}$ are weighted matrices of the three-layer feed-forward neural networks.

At the end of the spatio-temporal transformer module, a merge layer is added, as follows:

$$D = \text{ReLU}([Y^{\mathcal{S}}; Y^{\mathcal{T}}] W_m), \tag{3.15}$$

where $[\cdot;\cdot]$ indicates matrix concatenation at the axis of C, and $W_m \in \mathbb{R}^{2C \times \frac{C}{2}}$. Then, the output of the $l+1$-th hidden layer is

$$h_{l+1} = [D; h_l W_c], \tag{3.16}$$

where $W_c \in \mathbb{R}^{C \times \frac{C}{2}}$ is a weighted matrix for compressing h_l into $\mathbb{R}^{N \times T \times \frac{C}{2}}$.

This work supposes the time-series data $X \in \mathbb{R}^{N \times T}$ is collected, and a relation graph $\mathcal{G}$ and its correspond adjacency matrix $A \in \mathbb{R}^{N \times N}$ is given. At first, the input missing data $\mathcal{X}$ is scaled to be C-dimensional, namely as $\mathbb{R}^{N \times T \times C}$, by a 1×1 convolution layer, as shown in Figure 1. The feature information flow goes through the multiple spatio-temporal transformer blocks, and finally outputs the imputed data $\hat{\mathcal{X}}$ by two 1×1 convolution layers as

$$\hat{\mathcal{X}} = \text{Conv}_{1\times1}(\text{ReLU}(\text{Conv}_{1\times1}(h_{last}))), \tag{3.17}$$

where h_{last} is the last hidden feature of these transformer modules. The first convolution layer scales h_{last} from $\mathbb{R}^{N \times T \times C}$ to $\mathbb{R}^{N \times T \times C}$ at the axis of T as a nonlinear transformation with the ReLU function, and is then embedded into $\mathbb{R}^{N \times T}$, namely as $\hat{\mathcal{X}}$.

Note that, due to the bit-wise imputation in the bottom of Figure 1, the parameterized mapping f_θ, namely DSTTN, may be close to the linear function $f(\mathcal{X}) = \mathcal{X}$, but not the same as it. Thus, the skip connection from $\mathcal{X}$ to the output of the last spatio-temporal transformer block, as shown in Figure 3, can facilitate the fitness of DSTTN to be linear, which means stronger representation power with few additional parameters. Thus, the dense deployments of many skip connections, namely as dense connections, are adopted to the whole model, which can be comprehensively described as

$$h_l = f_l([h_{l-1}, h_{l-2}, \ldots, h_0]), \tag{3.18}$$

where $h_0 \in \mathbb{R}^{N \times T \times C}$ is the initial input generated from the first convolution layer in Figure 1. It improves the convergence efficiency and reuse of features for DSTTN.

3.3. Implementation and algorithm

To implement the training algorithm of DSTTN, a sliding window shown in Figure 1 is set to divide tensors into many subsamples with fixed size, and the sliding speed is set as one bit at one step. That is, the evolution process of a node is independent and identically distributed, and all time-series samples of the node are visible for each bit of the same row.

Then, we train the DSTTN by the standard Adam optimizer [40], which is a widely used momentum-based stochastic optimization method. The overall algorithm is presented as **Algorithm 1**.

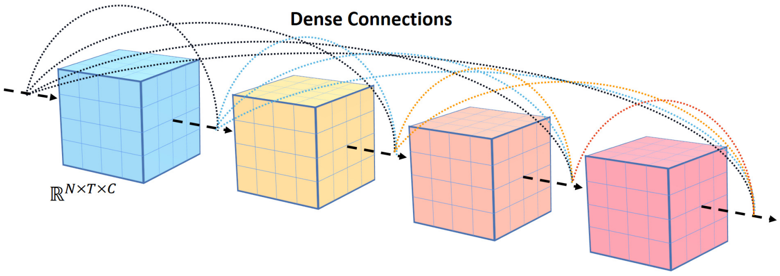

Figure 3. The schematic of the dense connections.

```
Algorithm 1: Cross-modal Missing Data Imputation
Input: Observation $\mathcal{X}$, parameterized $f_\theta$, the parameters of Adam optimizer and learning rate
       $\alpha = 0.0001$
Output: The imputed tensor $\mathcal{X}$
Split $\mathcal{X}$ into subsamples by sliding window for training;
/* Step 1:  Train DSTTN                                                   */
Initialize the DSTTN $f_\theta$ and the Adam optimizer;
while $\theta$ not converge do
    Shuffle the training subsamples;
    for each batch of subsamples $\mathcal{X}_b$ do
        $\hat{\mathcal{X}}_b \leftarrow f_\theta(\mathcal{X}_b)$
        $\theta \leftarrow Adam(\nabla_\theta \mathcal{L}(\hat{\mathcal{X}}_b), \alpha)$
Get the optimal $\theta^*$;
/* Step 2:  Impute missing data                                           */
for each subsample do
    $\hat{\mathcal{X}} \leftarrow f_{\theta^*}(\mathcal{X})$
Impute the missing value in $\mathcal{X}$ by the estimator $\hat{\mathcal{X}}$.
```

4. Experiments

4.1. Data description

The performance and stability of our proposed model is tested on two public traffic datasets, including the Caltrans performance measurement system district 5 (PeMSD5) dataset and the seattle inductive loop detector (SILD) dataset.

The **PeMSD5** dataset was collected from real-time traffic data of district 5 of the Caltrans performance measurement system in 2019. It provides real traffic flow data samples, which are aggregated with 53 sensor stations and traffic speed samples over 7 months. The time interval is 5 minutes, and the flow value represents the average traffic speed (km/h) within the corresponding 5 minutes interval. The location of these sensors is represented with latitude and longitude, and the distribution of all sensors is shown in Figure 4.

The **SILD** dataset is a public dataset, widely used in traffic prediction scenarios [41]. In order to adapt to the scenarios of this paper, the raw data is aggregated with 323 sensor stations and traffic speed (km/h) samples with 5-minute time intervals over a whole year. It also contains the corresponding latitude and longitude of stations.

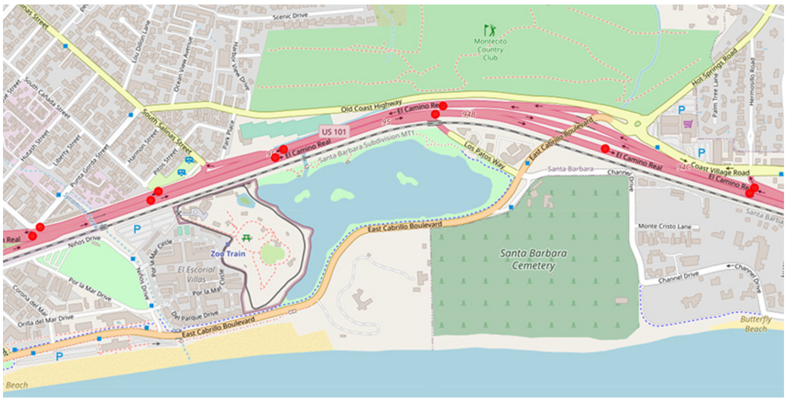

Figure 4. The distribution of the traffic sensor stations.

4.2. Experimental configuration

Various baseline methods are selected, including BGCP [13], LATC [16], DSAE [17], GAN-DSAE [19], BDLSTM [33], MVSTGN [21], and DGCRIN [34]. In the case of MAR, the original data which has no missing value is damaged randomly at different missing rates, and the damaged value is set to be zero. Then, the damaged data is considered as training data and the original is considered as test data. Similarly, CDM experiments are also conducted, where a part of the traffic nodes are randomly selected to be completely unobservable. The way to split training data and test data is same as the case of MAR.

For DSTTN, the graph $\mathcal{G}$ is set with adjacency matrix A. It is generated by $A_{ij} = 0$ if $i = j$, otherwise, $A_{ij} = e^{-d_{ij}^2}$ where d_{ij} is the spatial distance between node i and j. The spatial transformer is also initialized by the graph $\mathcal{G}$ as spatial positional code. The λ in Eq. (3.2) is set as 0.1. DSTTN is organized with 2 spatio-temporal transformer blocks. The size of embedded channels C is set as 32. The size of the sliding window T is set as 96, which is the same as in the sampling process of the other deep learning models.

Furthermore, the hidden layers of the DSAE are set with neural units of [2048, 1024, 2048]. The activation function is a sigmoid function. The broken 2-dimensional tensor is flattened into a 1-dimensional vector as the input of the DSAE. Meanwhile, the GAN-DSAE model adopts an extra GAN to generate so-called paralleled data [19] to expand training samples before training the DSAE. The generator of the GAN in GAN-DSAE is set with neural units of [512, 512, 512], and the discriminator is set with neural units of [288, 36, 1]. The activation functions are tanh functions applied to the first two layers and a sigmoid activation applied to the last layer.

For the BDLSTM model, it has 3 bidirectional LSTM layers. After the hidden feature extraction by BDLSTM, a single linear layer with 96 neural units is adopted to fit the mapping from the extracted features to the damaged tensor.

For the MVSTGN model, the number of attention layers is set as 2, and the number heads of each layer is also set as 2. The other configurations are the same as that in the source research paper.

For the DGCRIN model, the parameters are set as the default configuration in section 5.3 in [34]. Note that, the static graph in DGCRIN is obtained by $A_{ij} = 0$ if $i = j$, otherwise, $A_{ij} = e^{-d_{ij}^2}$. Here, d_{ij} is the spatial distance between node i and j, which is the same as the graph in DSTTN.

BGCP and LATC are both tensor-based methods. In BGCP, the Gibbs sampling iteration is set as 100 and the maximum iteration epochs is set as 1000. Besides, the rank degree is set as 10. In addition to the above special statements, the other model configurations are the same as the original papers for all comparative models.

4.3. Evaluation

To evaluate imputation performance, two indices are selected, including mean absolute percentage error (MAPE) and root mean square error (RMSE). They are defined as

$$\begin{aligned} MAPE &= \frac{1}{CI[\tilde{\mathbf{I}}_{it} = 1]} \sum_{i,t} \tilde{\mathbf{I}}_{it} \frac{|\hat{x}_{it} - x_{it}|}{x_{it}} \times 100\%, \\ RMSE &= \sqrt{\frac{1}{CI[\tilde{\mathbf{I}}_{it} = 1]} \sum_{i,t} \tilde{\mathbf{I}}_{it} (\hat{x}_{it} - x_{it})^2}, \end{aligned} \tag{4.1}$$

where $\tilde{\mathbf{I}}_{i,t} = 1$ if x_{it} is missing, otherwise, $\tilde{\mathbf{I}}_{i,t} = 0$. $CI[\tilde{\mathbf{I}}_{i,t} = 1]$ is a countif function of $\tilde{\mathbf{I}}_{i,t} = 1$.

Table 1. Performance comparison (in RMSE / MAPE) for MAR on PeMSD5 dataset.

Model / η	30%		40%		50%		60%		70%	
BGCP	3.53	3.73%	3.55	3.74%	3.56	3.76%	3.61	3.82%	3.63	3.83%
LATC	2.52	2.38%	2.55	2.88%	2.79	3.01%	3.14	3.22%	3.23	3.37%
DSAE	3.54	4.08%	3.61	4.27%	4.03	4.75%	4.28	5.10%	4.36	5.29%
GAN-DSAE	3.28	3.60%	3.51	3.90%	3.99	4.67%	4.27	5.07%	4.41	5.32%
BDLSTM	2.88	2.75%	2.91	2.77%	2.95	2.97%	3.21	3.08%	3.63	3.48%
MVSTGN	2.52	2.53%	2.70	2.59%	2.89	2.91%	2.98	2.99%	3.32	3.28%
DGCRIN	2.21	2.19%	2.59	2.51%	2.79	2.80%	3.01	2.98%	3.31	3.25%
DSTTN	2.05	1.94%	2.31	1.96%	2.45	2.51%	2.95	2.95%	3.28	3.13%

4.4. Imputation accuracy analysis

The performance comparison of all models on the PeMSD5 dataset is presented in Table 1. It is clear that DSTTN obtains state-of-the-art imputation performance in RMSE and MAPE. By visual inspection, BGCP and LATC perform stably at each η, but the imputation accuracy is not high. By contrast, deep-learning methods, including BDLSTM, MVSTGN, and DGCRIN, obtain lower RMSE and MAPE. It is worthy noting that the performance of the GAN-DSAE model is slightly better than DSAE as η is relatively low. That is, the synthetic data generated from GAN improves the overfitting problem by data augmentation to some extent. Thus, the DSTTN can also be further improved by combination with GAN-like data augmentation techniques, but this is not presented in this paper.

Moreover, there is a gap between DSTTN, DGCRIN, and MVSTGN, and the gap diminishes as the missing rate increases. Due to the fact that they are all deep-learning models, this is possibly a common problem. That is, the performances of deep-learning models drop to a same level if the missing rate of time-series data is too high. Despite this, it is still found that the cross-modal data fusion technique has some advantages to improve the imputation performance. Between them, MVSTGN only uses a spatial encode on the self-attention module. The spatial attention module just has a dynamic encode, rather than the location information from GPS data. Meanwhile, DGCRIN uses a graph-based gated recurrent unit model to sequentially extract the temporal features, and DSTTN embeds the cross-modal data fusion technique into the self-attention mechanism. Thus, the possible reason of performance improvements is induced from the self-attention mechanism and extra cross-modal information from GPS data.

4.5. Local complete data missing

CDM is a special case of MNAR. If a CDM issue happens, all time-series observation data are missing for some nodes in graph. That is, for the nodes, the missing rate is 100%, which is highest possible missing rate. But, at the same time, the other nodes are still observed, and their data are not missing. Thus, the missing rate here is the proportion of the nodes with missing data to all nodes, according to Eq (3.1).

In practice, if there arises long-term device failure for some traffic stations, or even if they have no equipped sensors, then a local CDM happens. For these traffic nodes, no data is collected to analyze traffic pattern and running law, as shown in the red curve of Figure 5. In general, we can not know the traffic flow evolution law of an unobservable node without any information about it. However, from the cross-modal perspective, the traffic flow data of different sensor nodes with short spatial distance is similar under a specific pattern. This is intuitively presented in Figure 5. The Pearson correlation r between sensor #49 and other sensors are calculated with corresponding p values for significance testing. It can be seen that sensor #49 has the similar traffic pattern to sensors #50 and #51, and not only is this reflected in strong correlations, but also in low p values. Thus, for example, if sensor #49 is unobservable, then we can still infer the missing data by the complete data of other adjacent sensors. In our experiments, we assume that only the local CDM case happens. That is, only a few sensors are set to be unobserved and the data of other sensors is collected.

Tables 2 and 3 present the results of comparative experiments on the PeMSD5 dataset and the SILD dataset, respectively. These experiments validate the imputation performance of DSTTN, compared to the other methods in local CDM cases at missing rates of $\eta = 5\%, 10\%, 15\%$. For the PeMSD5 dataset, the results show that these traditional deep-learning imputation methods all fail to deal with this problem with bad performance. This is similar to the results on the SILD dataset.

To provide more reliable evidence, we remove the representation loss function of the GAN-based models (see Eq (3) in [19]), and further reproduce and implement the naive Wasserstein GAN (WGAN) model [42] to solve the same CDM problem. Actually, the imputation performance of WGAN is still bad, and, by contrast, the performance of DSTTN is relatively better. Meanwhile, we also compare the imputation results of MVSTGN and non-cross-modal DSTTN (DSTTN*). Note that the objective function of DSTTN* does not have the graph Laplacian regularization term (see Eq (3.4)). They both achieve similar performance for CDM, and are both worse than the performance of DSTTN. Thus, the cross-modal technology has contributed in extrapolating the observed data to the unseen domain. It breaks though the bottleneck of CDM for deep-learning imputation methods as stated in [11].

Moreover, the performance of DSTTN exceeds that of the BGCP and LATC. For LATC, the MAPE is near 100%. This is because the objective function of LATC encourages the time-series data to show stronger temporal consistency (see Eq (2) in [16]). This consistency term can not work well in the case of CDM. By contrast, taking the missing rate at 5% as an example of BGCP, the percent of RMSE and MAPE on the PeMSD5 dataset decreases by 27.8% and 25.5%, respectively, compared to DSTTN. Actually, since, the loss of positional embedding information, BGCP can only reproduce the missing data by the structural property of traffic flow tensor, regardless of the heterogeneity of each traffic node to be imputed. Thus, given the recovered missing data, BGCP can not offer a classification to match data and nodes, which results in the homogeneous and coarse imputation of BGCP.

Thus, it is reasonable for DSTTN to consider cross-modal constraints based on the spatial similarity of traffic flow. That is, completely unobserved traffic flow data can be recovered through the cross-modal way from spatial modals to temporal modals. To obtain higher accuracy, the used graph can be constructed by other similarity measurements, but it will not change the reasonability of the cross-modal data fusion.

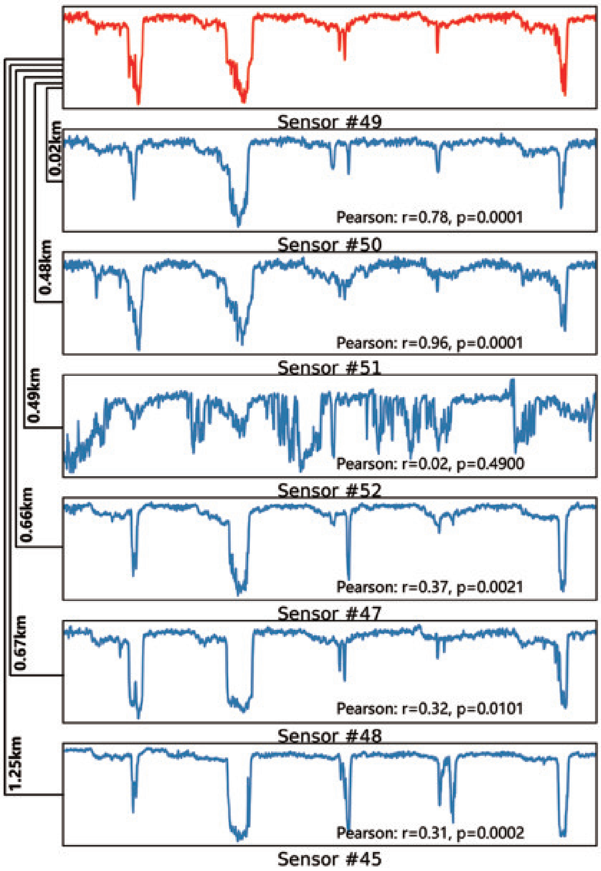

Figure 5. The pattern similarity is positively correlated to sensor distance on the PeMSD5 dataset.

Table 2. The results of CDM imputation on PeMSD5 dataset.

RMSE / MAPE	5%	10%	15%
BGCP	5.71 / 7.01%	5.86 / 6.94%	5.96 / 7.11%
LATC	62.72 / 99.99%	63.44 / 99.99%	64.25 / 100.00%
DSAE	32.12 / 38.12%	31.12 / 40.49%	35.67 / 45.21%
GAN-DSAE	35.47 / 48.29%	42.65 / 56.46%	43.91 / 55.93%
WGAN	29.68 / 39.73%	32.65 / 43.13%	37.29 / 49.38%
BDLSTM	38.23 / 51.50%	40.36 / 56.80%	41.81 / 53.85%
MVSTGN	31.09 / 40.22%	34.18 / 41.59%	35.82 / 47.33%
DSTTN*	28.93 / 36.11%	31.21 / 39.47%	31.97 / 40.62%
DSTTN	**4.12** / **5.22%**	**4.17** / **5.24%**	**5.45** / **6.94%**

*Note: * denotes DSTTN without graph Laplacian regularization term.*

Table 3. The results of CDM imputation on SILD dataset.

RMSE / MAPE	5%	10%	15%
BGCP	10.18 / 24.02%	10.35 / 24.27%	10.83 / 29.05%
LATC	57.84 / 99.99%	58.08 / 99.99%	58.23 / 100.00%
DSAE	54.58 / 93.60%	54.68 / 94.40%	54.70 / 93.78%
GAN-DSAE	34.74 / 61.29%	35.99 / 63.80%	38.60 / 66.48%
WGAN	48.65 / 81.23%	53.38 / 90.47%	55.75 / 94.77%
BDLSTM	35.45 / 65.20%	38.60 / 62.11%	42.97 / 71.61%
MVSTGN	55.84 / 92.81%	54.40 / 93.79%	54.70 / 92.31%
DSTTN*	55.91 / 94.26%	53.54 / 90.20%	55.45 / 94.14%
DSTTN	**9.72** / **21.18%**	**10.30** / **24.12%**	**9.81** / **24.12%**

*Note: * denotes DSTTN without graph Laplacian regularization term.*

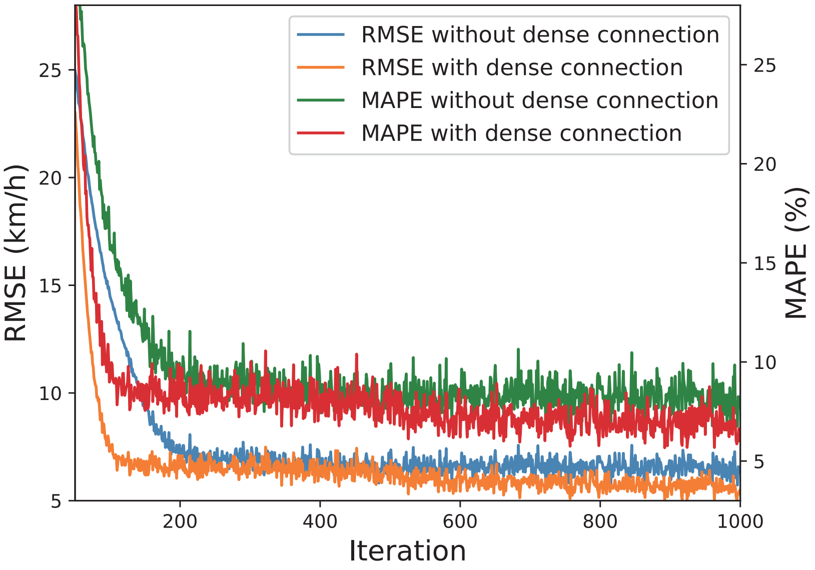

Figure 6. The comparison for DSTTN with and without dense connections.

4.6. Dense connections

To test the effects of dense connections, the decline curves concerning the real-time MAPE and RMSE are plotted in Figure 6. In the controlled experiments, the configurations of DSTTN are same, except for the dense connections. It is observed that the real-time MAPE and RMSE with dense connections decline faster than that without dense connections as the iteration increases from 50 to 1000. Furthermore, the real-time MAPE and RMSE with dense connections are always lower than that without dense connections within the limited iterations. Thus, the deployment of the dense connections accelerates the training process and improves the accuracy of DSTTN.

5. Conclusions

To solve the time-series data missing problems, we propose a cross-modal method. This method uses a graph Laplacian regularization constraint to extrapolate the remaining observation data to the target data domain. It uses the DSTTN model to fuse spatial and temporal modal data, and finally impute the expected time-series data.

Sufficient comparative experiments are conducted to test the performance of the cross-modal method. The experimental results verify that DSTTN achieves state-of-the-art performance not only in the MAR case, but also in the CDM case. The results reveal that the graph Laplacian regularization constraint enables deep-learning methods to extrapolate data efficiently.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgment

This work was supported by the National Natural Science Foundation of China (Grant Nos. 62233004, 62073076, 62273090, 61903079) and the Science and Technology Project from State Grid Jiangsu Electric Power CO. Ltd (Grant No. SGJSYF00YHJS2200052)

Conflict of interest

The authors declare there is no conflict of interest.

References

1. X. Li, X. Yang, J. Cao, Event-triggered impulsive control for nonlinear delay systems, *Automatica*, **117** (2020), 108981. https://doi.org/10.1016/j.automatica.2020.108981
2. X. Li, D. Peng, J. Cao, Lyapunov stability for impulsive systems via event-triggered impulsive control, *IEEE Trans. Autom. Control*, **65** (2020), 4908–4913. https://doi.org/10.1109/TAC.2020.2964558

3. P. Dai, W. Yu, G. Wen, S. Baldi, Distributed reinforcement learning algorithm for dynamic economic dispatch with unknown generation cost functions, *IEEE Trans. Ind. Inf.*, **16** (2019), 2258–2267. https://doi.org/10.1109/TII.2019.2933443

4. X. Miao, Y. Wu, L. Chen, Y. Gao, J. Wang, J. Yin, Efficient and effective data imputation with influence functions, in *Proceedings of the VLDB Endowment*, **15** (2021), 624–632. https://doi.org/10.14778/3494124.3494143

5. A. Blázquez-García, K. Wickstrøm, S. Yu, K. Ø. Mikalsen, A. Boubekki, A. Conde, et al., Selective imputation for multivariate time series datasets with missing values, *IEEE Trans. Knowl. Data Eng.*, **2023** (2023). https://doi.org/10.1109/TKDE.2023.3240858

6. Y. Li, Z. Li, L. Li, Missing traffic data: comparison of imputation methods, *IET Intell. Transp. Syst.*, **8** (2014), 51–57. https://doi.org/10.1049/iet-its.2013.0052

7. C. Zhang, Y. Cui, Z. Han, J. T. Zhou, H. Fu, Q. Hu, Deep partial multi-view learning, *IEEE Trans. Pattern Anal. Mach. Intell.*, **44** (2020), 2402–2415. https://doi.org/10.1109/TPAMI.2020.3037734

8. M. Yang, Y. Li, P. Hu, J. Bai, J. C. Lv, X. Peng, Robust multi-view clustering with incomplete information, *IEEE Trans. Pattern Anal. Mach. Intell.*, **45** (2022), 1055–1069. https://doi.org/10.1109/TPAMI.2022.3155499

9. X. Miao, Y. Wu, L. Chen, Y. Gao, J. Yin, An experimental survey of missing data imputation algorithms, *IEEE Trans. Knowl. Data Eng.*, **2022** (2022). https://doi.org/10.1109/TKDE.2022.3186498

10. M. Kang, R. Zhu, D. Chen, X. Liu, W. Yu, CM-GAN: A cross-modal generative adversarial network for imputing completely missing data in digital industry, *IEEE Trans. Neural Networks Learn. Syst.*, **2023** (2023). https://doi.org/10.1109/TNNLS.2023.3284666

11. J. Wang, C. Lan, C. Liu, Y. Ouyang, T. Qin, W. Lu, et al., Generalizing to unseen domains: A survey on domain generalization, *IEEE Trans. Knowl. Data Eng.*, **2022** (2022). https://doi.org/10.24963/ijcai.2021/628

12. B. Sun, L. Ma, W. Cheng, W. Wen, P. Goswami, G. Bai, An improved k-nearest neighbours method for traffic time series imputation, in *2017 Chinese Automation Congress*, (2017), 7346–7351. https://doi.org/10.1109/CAC.2017.8244105

13. X. Chen, Z. He, L. Sun, A bayesian tensor decomposition approach for spatiotemporal traffic data imputation, *Transp. Res. Part C Emerging Technol.*, **98** (2019), 73–84. https://doi.org/10.1016/j.trc.2018.11.003

14. S. Coogan, C. Flores, P. Varaiya, Traffic predictive control from low-rank structure, *Transp. Res. Part B Methodol.*, **97** (2017), 1–22. https://doi.org/10.1016/j.trb.2016.11.013

15. H. Tan, G. Feng, J. Feng, W. Wang, Y. J. Zhang, F. Li, A tensor-based method for missing traffic data completion, *Transp. Res. Part C Emerging Technol.*, **28** (2013), 15–27. https://doi.org/10.1016/j.trc.2012.12.007

16. X. Chen, M. Lei, N. Saunier, L. Sun, Low-rank autoregressive tensor completion for spatiotemporal traffic data imputation, *IEEE Trans. Intell. Transp. Syst.*, **23** (2022), 12301–12310. https://doi.org/10.1109/TITS.2021.3113608

17. Y. Duan, Y. Lv, Y. L. Liu, F. Y. Wang, An efficient realization of deep learning for traffic data imputation, *Transp. Res. Part C Emerging Technol.*, **72** (2016), 168–181. https://doi.org/10.1016/j.trc.2016.09.015

18. Y. Tian, K. Zhang, J. Li, X. Lin, B. Yang, LSTM-based traffic flow prediction with missing data, *Neurocomputing*, **318** (2018), 297–305. https://doi.org/10.1016/j.neucom.2018.08.067

19. Y. Chen, Y. Lv, F. Y. Wang, Traffic flow imputation using parallel data and generative adversarial networks, *IEEE Trans. Intell. Transp. Syst.*, **21** (2019), 1624–1630. https://doi.org/10.1109/TITS.2019.2910295

20. X. Miao, Y. Wu, J. Wang, Y. Gao, X. Mao, J. Yin, Generative semi-supervised learning for multivariate time series imputation, in *Proceedings of the AAAI conference on artificial intelligence*, **35** (2021), 8983–8991. https://doi.org/10.1609/aaai.v35i10.17086

21. Y. Yao, B. Gu, Z. Su, M. Guizani, MVSTGN: A multi-view spatial-temporal graph network for cellular traffic prediction, *IEEE Trans. Mobile Comput.*, **2021** (2021). https://doi.org/10.1109/TMC.2021.3129796

22. T. Baltrušaitis, C. Ahuja, L. P. Morency, Multimodal machine learning: A survey and taxonomy, *IEEE Trans. Pattern Anal. Mach. Intell.*, **41** (2018), 423–443. https://doi.org/10.1109/TPAMI.2018.2798607

23. D. Wang, Y. Yan, R. Qiu, Y. Zhu, K. Guan, A. Margenot, et al., Networked time series imputation via position-aware graph enhanced variational autoencoders, in *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, (2023), 2256–2268. https://doi.org/10.1145/3580305.3599444

24. D. Chen, Q. Shao, Z. Liu, W. Yu, C. L. P. Chen, Ridesourcing behavior analysis and prediction: A network perspective, *IEEE Trans. Intell. Transp. Syst.*, **23** (2022), 1274–1283. https://doi.org/10.1109/TITS.2020.3023951

25. S. Lee, D. B. Fambro, Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting, *Transp. Res. Record*, **1678** (1999), 179–188. https://doi.org/10.3141/1678-22

26. M. Zhong, S. Sharma, P. Lingras, Genetically designed models for accurate imputation of missing traffic counts, *Transp. Res. Record*, **1879** (2004), 71–79. https://doi.org/10.3141/1879-09

27. L. Qu, L. Li, Y. Zhang, J. Hu, PPCA-based missing data imputation for traffic flow volume: A systematical approach, *IEEE Trans. Intell. Transp. Syst.*, **10** (2009), 512–522. https://doi.org/10.1109/TITS.2009.2026312

28. L. Li, Y. Li, Z. Li, Efficient missing data imputing for traffic flow by considering temporal and spatial dependence, *Transp. Res. Part C Emerging Technol.*, **34** (2013), 108–120. https://doi.org/10.1016/j.trc.2013.05.008

29. J. Tang, G. Zhang, Y. Wang, H. Wang, F. Liu, A hybrid approach to integrate fuzzy c-means based imputation method with genetic algorithm for missing traffic volume data estimation, *Transp. Res. Part C Emerging Technol.*, **51** (2015), 29–40. https://doi.org/10.1016/j.trc.2014.11.003

30. S. Wang, G. Mao, Missing data estimation for traffic volume by searching an optimum closed cut in urban networks, *IEEE Trans. Intell. Transp. Syst.*, **20** (2018), 75–86. https://doi.org/10.1109/TITS.2018.2801808

31. Y. Wang, Y. Zhang, X. Piao, H. Liu, K. Zhang, Traffic data reconstruction via adaptive spatial-temporal correlations, *IEEE Trans. Intell. Transp. Syst.*, **20** (2018), 1531–1543. https://doi.org/10.1109/TITS.2018.2854968

32. X. Chen, Z. He, J. Wang, Spatial-temporal traffic speed patterns discovery and incomplete data recovery via svd-combined tensor decomposition, *Transp. Res. Part C Emerging Technol.*, **86** (2018), 59–77. https://doi.org/10.1016/j.trc.2017.10.023

33. Z. Cui, R. Ke, Z. Pu, Y. Wang, Stacked bidirectional and unidirectional lstm recurrent neural network for forecasting network-wide traffic state with missing values, *Transp. Res. Part C Emerging Technol.*, **118** (2020), 102674. https://doi.org/10.1016/j.trc.2020.102674

34. X. Kong, W. Zhou, G. Shen, W. Zhang, N. Liu, Y. Yang, Dynamic graph convolutional recurrent imputation network for spatiotemporal traffic missing data, *Knowl. Based Syst.*, **261** (2023), 110188. https://doi.org/10.1016/j.knosys.2022.110188

35. G. Shen, W. Zhou, W. Zhang, N. Liu, Z. Liu, X. Kong, Bidirectional spatial–temporal traffic data imputation via graph attention recurrent neural network, *Neurocomputing*, **531** (2023), 151–162. https://doi.org/10.1016/j.neucom.2023.02.017

36. W. Zhang, P. Zhang, Y. Yu, X. Li, S. A. Biancardo, J. Zhang, Missing data repairs for traffic flow with self-attention generative adversarial imputation net, *IEEE Trans. Intell. Transp. Syst.*, **23** (2021), 7919–7930. https://doi.org/10.1109/TITS.2021.3074564

37. M. Kang, R. Zhu, D. Chen, C. Li, W. Gu, X. Qian, et al., A cross-modal generative adversarial network for scenarios generation of renewable energy,*IEEE Trans. Power Syst.*, **2023** (2023). https://doi.org/10.1109/TPWRS.2023.3277698

38. L. Li, J. Zhang, Y. Wang, B. Ran, Missing value imputation for traffic-related time series data based on a multi-view learning method, *IEEE Trans. Intell. Transp. Syst.*, **20** (2018), 2933–2943. https://doi.org/10.1109/TITS.2018.2869768

39. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., Attention is all you need, in *Advances in Neural Information Processing Systems*, (2017), 5998–6008.

40. D. P. Kingma, J. Ba, ADAM: A method for stochastic optimization, in *International Conference on Learning Representations*, 2015.

41. Z. Cui, K. Henrickson, R. Ke, Y. Wang, Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting, *IEEE Trans. Intell. Transp. Syst.*, **21** (2019), 4883–4894. https://doi.org/10.1109/TITS.2019.2950416

42. M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, in *International Conference on Machine Learning*, (2017), 214–223.

AIMS Press