Transportation Research Part C 143 (2022) 103826

Contents lists available at ScienceDirect

Transportation Research Part C

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

Memory-augmented dynamic graph convolution networks for traffic data imputation with diverse missing patterns

Yuebing Liang [a], Zhan Zhao [a,b,*], Lijun Sun [c]

[a] Department of Urban Planning and Design, The University of Hong Kong, Hong Kong
[b] Musketeers Foundation Institute of Data Science, The University of Hong Kong, Hong Kong
[c] Department of Civil Engineering, McGill University, Montreal, QC H3A 0C3, Canada

ARTICLE INFO

Keywords:
Traffic data imputation
Spatiotemporal dependencies
Missing patterns
Graph convolutional neural networks
Memory neural networks

ABSTRACT

Missing data is an inevitable and ubiquitous problem for traffic data collection in intelligent transportation systems. Recent research has employed graph neural networks (GNNs) for spatiotemporal data imputation and achieved promising performance. However, there still exist two limitations to be addressed: first, existing approaches are generally limited in directly leveraging global spatiotemporal information from different nodes at different time; second, most of these approaches do not consider the unique characteristics of transportation systems or traffic data, including dynamic spatial dependencies and correlated missing patterns. To fill these research gaps, we propose a novel deep learning framework called Memory-augmented Dynamic Graph Convolution Networks (MDGCN) to impute missing traffic data. The model uses a recurrent layer to capture temporal information and a graph convolution layer to capture spatial information. To address the first research gap, we introduce an external memory network to store and share the global spatiotemporal information across the traffic network. For the second research gap, a graph structure estimation technique is proposed to learn dynamic spatial dependencies directly from traffic data. In addition, four types of missing patterns with various missing ratios are considered in model evaluation. Extensive experiments based on two public traffic speed datasets are conducted. The results show that our proposed model outperforms existing state-of-the-art deep learning approaches in all kinds of missing scenarios, and both the proposed external memory network and graph structure estimation technique contribute to the model performance. The model performance is competitive in most cases even without complete training data.

1. Introduction

Traffic data from real traffic systems plays an essential role in transportation research and applications, such as predicting traffic conditions, planning driving routes, and optimizing traffic flows. Traffic data is mainly collected from two types of sensors: stationary sensors (e.g., loop detectors) and mobile sensors (e.g., GPS probes). However, missing data is an inevitable problem for both stationary and mobile sensors. Stationary sensors can easily suffer from device problems such as detector malfunction, communication failure and power outage, whereas the data collected from mobile sensors is usually sparse with highly erratic spatial and temporal resolutions (Asif et al., 2013). Missing data problems seriously affect the real-time monitoring of traffic conditions and further limit other downstream applications. Therefore, how to estimate missing data, or traffic data imputation, becomes a critical issue.

* Corresponding author at: Department of Urban Planning and Design, The University of Hong Kong, Hong Kong.
E-mail address: zhanzhao@hku.hk (Z. Zhao).

https://doi.org/10.1016/j.trc.2022.103826
Received 16 September 2021; Received in revised form 10 July 2022; Accepted 25 July 2022

The traffic data imputation problem has received much attention over the years. Early studies mainly model traffic data of each location as time series and neglect the spatial correlations (Smith et al., 2003; Zhong et al., 2004). Recently, matrix (or tensor) factorization methods have been introduced for traffic data imputation, and shown to be effective in retrieving correlations across different dimensions (Chen et al., 2019; Chen and Sun, 2021). However, these methods only rely on the global low-rank structure and usually do not explicitly model the underlying local consistency such as the spatial constraints of road networks or the temporal smoothness. As a result, these models may be limited in capturing the complexity of spatiotemporal dependencies in traffic data. With the proven success of deep learning models in a wide range of tasks, neural network-based approaches have also been adopted for the data imputation problem, including Autoencoders (Duan et al., 2016), Recurrent Neural Networks (RNNs) (Berglund et al., 2015; Li et al., 2018) and Generative Adversarial Networks (GANs) (Yoon et al., 2018; Luo et al., 2018). Recent studies have shifted more focus to Graph Neural Networks (GNNs) for reconstructing multivariate time series (Cini et al., 2021; Ye et al., 2021) and demonstrated the effectiveness of GNNs in capturing spatial dependencies at the network level. Despite promising results of introducing GNNs, we argue that there are still two important research gaps.

First, existing models are generally limited in directly leveraging the global spatiotemporal information from different nodes at different time to enhance the imputation performance. Most previous graph learning approaches use two separate components to capture the localized spatial dependency and temporal smoothness. By feeding the learned representations from spatial and temporal modeling components to each other, they can only indirectly capture spatiotemporal information from other nodes at other time steps. However, we believe that directly utilizing such global spatiotemporal information can significantly enhance the imputation performance, especially when spatial or temporal information is limited.

Second, the unique characteristics of transportation systems or traffic data are often overlooked in most existing GNN-based studies for traffic imputation. One such characteristic lies in the dynamism of traffic systems. While existing approaches are usually developed for static graphs with a pre-defined adjacency matrix, the spatiotemporal dependencies across a traffic network can change dynamically and are challenging to specify due to ever-changing traffic situations. It is almost impossible to capture all the dynamics in a transportation system using a pre-defined adjacency matrix. Although some recent techniques have been proposed to model dynamic spatial dependencies across traffic networks, they are developed for prediction tasks, and it is unknown whether they can properly handle the incomplete and heterogeneous data for imputation tasks. In addition, for real-world traffic data, the missing values are often temporally or spatially correlated. For stationary sensors this may occur due to regional malfunction or long-term maintenance backlog (Li et al., 2018). For mobile sensors this often happens in urban areas with low traffic volumes. Existing research focus more on random missing scenarios in which missing data points are completely independent of each other. Whether GNN-based models can properly reconstruct traffic data for correlated missing patterns with various missing ratios is still underexplored.

To address the issues above, we propose a novel deep learning architecture, called Memory-augmented Dynamic Graph Convolution Networks (MDGCN), to provide accurate and robust imputation results for different missing patterns. It consists of several spatiotemporal blocks (ST-blocks), each comprises a bidirectional recurrent layer to capture temporal dependencies and a graph convolution layer to capture spatial dependencies. An external memory network is introduced to store and share global spatiotemporal information from different sensors at different time steps. To model dynamic spatial correlations, a graph structure estimation layer is incorporated in the ST-blocks. To validate the effectiveness of our model, we conduct experiments on two public datasets, one collected from a freeway network in Los Angeles, CA, and the other from an urban road network in Seattle, WA. The results show that our model outperforms the state-of-the-art deep learning methods significantly under diverse missing scenarios. We also test the model under different data conditions, and find competitive model performance in most situations even with only incomplete training data. Our main contributions are as follows:

- We propose a memory-augmented graph learning approach to better leverage both global and local spatiotemporal information for the traffic data imputation problem. To the best of our knowledge, we are among the first to incorporate memory neural networks with GNNs for the imputation problem.
- We introduce an external memory network to store and share the global spatiotemporal information across the traffic network. Specifically, we cluster different nodes to several groups and the shareable information of each group is stored in a learned memory matrix. The knowledge for imputation is then distilled with an attention mechanism.
- We design a graph structure estimation technique to learn dynamic spatial dependencies on the network structure directly from traffic data. This is particularly important for transportation systems because of the time-varying flow distributions and traffic conditions.
- Extensive experiments are conducted to compare our proposed model with several state-of-the-art deep learning models in four types of missing patterns with a wide range of missing ratios. The results indicate the superior performance of our model under diverse missing scenarios, and validate the effects of the proposed external memory network and dynamic adjacency matrices.

2. Literature review

Previous studies have developed a variety of imputation methods based on different missing patterns for different types of traffic data. The performance of a method can be greatly influenced by the specific missing pattern or data type. Therefore, in this section, we first review the methods for traffic data imputation, and then summarize the different missing patterns and traffic data used in the literature. In addition, we provide a review of GNN-based models for various traffic applications to better position our paper in the field.

2.1. Traffic data imputation methods

Early models for traffic data imputation mainly rely on temporal patterns and barely utilize the spatial structure of traffic data. The simplest method is Historical Average, which fills missing values based on the average values of the same time intervals in the past (Smith et al., 2003). Ni and Leonard (2005) employed a Bayesian network to learn the probability distribution from observed data and used the best fit to impute missing values. Qu et al. (2009) introduced a technique called probabilistic principle component analysis (PPCA) which utilizes daily periodicity and interval variation of traffic data. Recent research incorporates spatial information into missing data reconstruction. Aydilek and Arslan (2013) combined support vector regression (SVR) with a genetic algorithm to capture both spatial and temporal relationships in the traffic network. Laña et al. (2018) introduced a spatial context sensing model to reconstruct traffic data using information from surrounding sensors. These models demonstrate that spatial information is helpful for traffic data imputation. However, they have focused on utilizing local spatial information from neighborhood locations and failed to make full use of global spatiotemporal information.

Recently, matrix (or tensor) factorization methods have been introduced for traffic data imputation, which construct traffic data as a multi-dimensional matrix and estimate a low-rank approximation of the incomplete matrix. Chen et al. (2019) extended Bayesian probabilistic matrix factorization to high-order tensors and applied it to impute incomplete traffic data. A temporal factorization framework was proposed by Chen and Sun (2021), which combines low-rank matrix factorization with vector autoregressive process. Compared with previous models, tensor factorization is good at capturing multi-dimensional structural dependencies and thus making imputations at the system level. However, it only applies to statistical data with a low rank and needs to be learned from scratch for every new batch of incomplete data (Zhang et al., 2021b). Moreover, considering the nonlinearity and complexity of spatiotemporal dependencies in traffic data, it might be difficult for tensor factorization models to fully retrieve traffic features and provide robust imputation with diverse missing patterns and missing ratios.

With the recent advances in deep learning, a number of deep neural network models have also been developed for traffic data imputation. Compared with the tensor factorization approach, deep learning models do not make additional assumptions about the data and can be pre-trained for online application when sufficient training data is provided. Duan et al. (2016) first introduced deep learning to the traffic data imputation problem using denoising stacked autoencoders. Recent studies based on recurrent networks have achieved great success for generic imputation tasks (Lipton et al., 2016; Liu et al., 2019). In particular, Berglund et al. (2015) employed bidirectional RNNs as generative models to fill in missing gaps in text data. Cao et al. (2018) put forward a novel imputation approach for time series which utilizes feature correlations in a bidirectional recurrent dynamical system. Generative Adversarial Networks (GANs) have also been applied. For example, Yoon et al. (2018) adapted GANs to learn the distribution of data, which is further used to generate missing values. Luo et al. (2018) incorporated GANs with recurrent units for multivariate time series imputation. Although these approaches demonstrate the effectiveness of deep learning in the field of data imputation, they do not focus on traffic data and barely consider spatial information across the transportation network. To leverage spatial correlations, Li et al. (2018) developed a multi-view learning method with Long Short-Term Memory (LSTM) for temporal dependencies and SVR for spatial dependencies. Asadi and Regan (2019) proposed a convolution recurrent autoencoder for traffic data imputation, using multi-range CNNs to model spatial correlations. While CNNs work well for Euclidean correlations in grid-structured data (e.g., images), they are not well suited for the non-Euclidean relationships in irregular road networks.

Recently, GNNs have shown effectiveness in learning spatiotemporal representations through message passing (Wu et al., 2020). Based upon this, Wu et al. (2020) developed a Graph Convolutional Network (GCN)-based model to recover data for unobserved sensors (i.e., kriging). Spinelli et al. (2020) solved the data imputation problem with an adversarially-trained GNN. A GNN-based denoising autoencoder was developed by Kuppannagari et al. (2021) for the imputation of smart power grids. Cini et al. (2021) introduced a GNN-based imputation model to reconstruct generic multivariate time series. These studies do not consider the unique characteristics of transportation systems or traffic data, such as dynamic spatial dependencies and correlated missing patterns. To solve the traffic data imputation problem, Zhang et al. (2021a) designed a graph convolutional recurrent module, which however still neglects spatial dependencies evolving with time. Ye et al. (2021) used Graph Attention Networks (GATs) to adaptively learn the spatial dependencies between adjacent sensors. Nevertheless, they assume that spatial dependencies only exist between geographically adjacent locations and cannot capture the global spatiotemporal information across the traffic network.

2.2. Missing patterns and traffic data types in previous studies

Missing patterns and data types can significantly influence the imputation performance of any method. Previous studies generally classify the patterns of missing data into three classes: Missing Completely at Random, Missing at Random and Not Missing at Random (Little and Rubin, 2019). More specifically, Li et al. (2018) classified the missing patterns in intelligent transportation systems into four categories: (1) Random Missing (RM) (Fig. 1a), in which missing values are independent of each other; (2) Temporally correlated Missing (TCM) (Fig. 1b), in which missing values have temporal correlations; (3) Spatially correlated Missing (SCM) (Fig. 1c), in which missing values are spatially correlated to their neighbors' readings; (4) Block Missing (BM) (Fig. 1d), in which missing values are both temporally and spatially correlated and form blocks. This classification is also adopted in our research. Based upon missing ratios, missing patterns can also be classified as non-completely missing and completely missing. This study focuses on non-completely missing patterns, where at least one observed data exists in both spatial and temporal dimensions. Completely missing patterns include completely TCM (Wu et al., 2020) and completely SCM (Li et al., 2020b). In completely TCM, some sensors/locations are totally unobserved, whereas in completely SCM, no information is observed for some time slots. These cases are not considered in this paper. Distinction can also be made regarding the traffic data types. Due to differences in the data

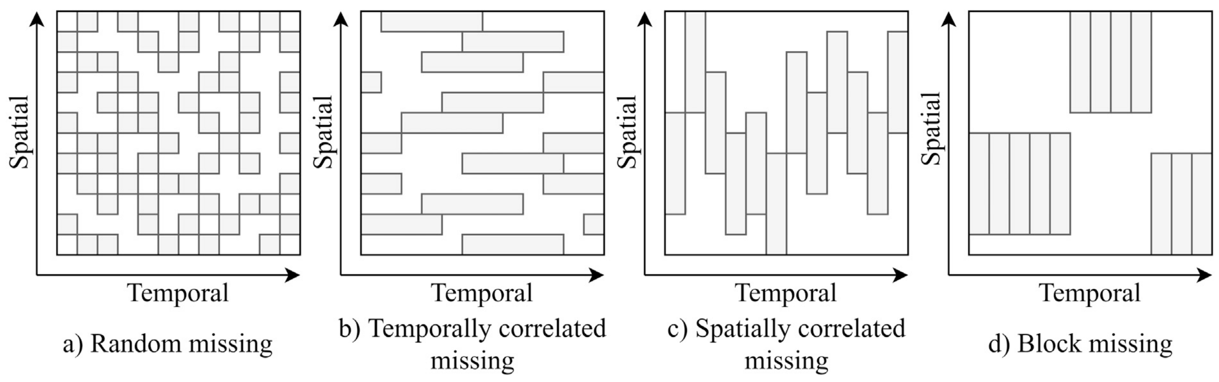
Spatial
Temporal
a) Random missing
b) Temporally correlated missing
c) Spatially correlated missing
d) Block missing

Fig. 1. Patterns of missing data.

Table 1
Methods, data and missing patterns in the literature.

Paper	Method	Data		Missing patterns				Missing ratio
		FD	UD	RM	TCM	SCM	BM	
Ni and Leonard (2005)	Bayesian Networks	Y		Y				30%
Zhang and Liu (2009)	Least Squares SVR		Y	Y				20%–50%
Qu et al. (2009)	PPCA		Y	Y		Y		0%–50%
Aydilek and Arslan (2013)	SVR+genetic algorithm	Y		Y				1%–25%
Ran et al. (2016)	tensor factorization	Y		Y				5%–80%
Duan et al. (2016)	denoising autoencoders	Y		Y	Y			5%–50%
Bae et al. (2018)	two cokriging methods	Y	Y	Y	Y		Y	10%–40%
Laña et al. (2018)	two machine learning models	Y	Y	Y	Y			0%–100%
Li et al. (2018)	LSTM + SVR	Y		Y	Y	Y	Y	5%–50%
Chen et al. (2019)	tensor factorization		Y	Y				10%–50%
Asadi and Regan (2019)	recurrent convolutional autoencoders	Y						unknown
Li et al. (2020a)	stacked autoencoders	Y		Y	Y			5%–40%
Li et al. (2020b)	prophet + Random Forest	Y		Y	Y			10%–90%
Wu et al. (2020)	GCNs	Y			Y			100%
Chen and Sun (2021)	tensor factorization	Y	Y	Y	Y			30%, 70%
Zhang et al. (2021b)	Generative Adversarial Networks	Y			Y	Y	Y	10%–80%
Zhang et al. (2021a)	GCNs		Y	Y				20%–70%
Ye et al. (2021)	GATs	Y		Y			Y	10%–90%
Cini et al. (2021)	GNNs	Y		Y			Y	25%

collection methods and underlying road networks, traffic data can be generally classified into freeway data (FD) and urban road network data (UD). FD is usually collected with stationary sensors on freeway networks while UD is collected with mobile sensors (e.g., probe vehicles) on urban road networks. Typically, the former has higher temporal granularity, while the latter has higher spatial coverage.

Table 1 summarizes the traffic data types, missing patterns and missing ratios in the literature. We can find that RM is the most commonly studied missing pattern, while the other missing patterns with temporally or spatially correlated missing values are less discussed. Compared with RM, the other patterns are more challenging due to lack of spatial or temporal adjacent information. As a result, approaches developed for random missing values might not be suitable for other missing patterns. Additionally, previous studies are usually conducted on either freeway networks or urban road networks. However, the two types of data may display different traffic characteristics due to different road design and functionality. Methods optimized for FD may not work well for UD. Missing ratios also affect the performance of models. Some previous models are developed for low missing ratios and might fail to show stable performance when the missing ratio is high. Therefore, a general approach is needed to provide accurate and robust results for different missing patterns and data types in a wide range of missing ratios.

2.3. *Graph neural networks in traffic research*

Beyond traffic data imputation, GNNs have been successfully applied to various prediction tasks in traffic research, including traffic speed prediction (Yu et al., 2017), travel demand prediction (Liang et al., 2022a,b), vehicle trajectory prediction (Liang and Zhao, 2021), and traffic accident prediction (Yu et al., 2021). To jointly extract spatiotemporal features hidden in the traffic network, researchers typically use GNNs to capture network-level spatial relations, along with RNNs or CNNs to extract temporal dependencies. By combining the recurrent architecture with diffusion graph convolution layers, Li et al. (2017) introduced a deep learning framework for traffic forecasting. Yu et al. (2017) used graph convolutions to extract spatial features and gated CNNs to extract temporal features. These approaches are all based on a fixed and pre-determined graph structure. To uncover the true dependencies hidden in the traffic network, Wu et al. (2019) developed an adaptive adjacency matrix to represent hidden spatial dependencies. Recently, attention mechanisms have been introduced to model spatiotemporal dependencies evolving with time. A transform attention mechanism was applied by Zheng et al. (2020) to adaptively learn spatial and temporal dependencies from traffic features. Park et al. (2020) developed a novel spatial attention mechanism by introducing sentinel vectors to control for irrelevant features. However, all these approaches only consider the local spatial dependency and temporal smoothness of traffic data, failing to directly leverage global spatiotemporal information from different nodes at different time. In addition, these approaches are all designed for prediction tasks and may not be suitable for traffic data imputation. Compared with prediction tasks, the imputation problem is challenging because of the limited observed data and the diversity of missing patterns. A robust method that can leverage both global and local information hidden in traffic data for the imputation problem is still needed.

3. Preliminaries

In this section, we first elaborate on how we represent the transportation network as a graph and then formulate our problem definition.

3.1. Graph representation of transportation networks

In this work, we define a traffic network as a weighted directed graph $G = (V, E, \boldsymbol{A})$, where V is a set of $N = |V|$ connected nodes (i.e., sensors or road links), E a set of edges indicating the connectivity between nodes, and $\boldsymbol{A} \in \mathbb{R}^{N\times N}$ a weighted adjacency matrix representing the proximity between nodes. Existing approaches often define the adjacency matrix as a function of distance or connectivity. However, our previous research has shown that transportation networks are sometimes heterophilous, where connected road segments or adjacent sensors can have different traffic features due to the regulating effect of traffic lights. Therefore, we determine the adjacency matrix $\boldsymbol{A}$ based on both geographic and semantic correlations. For each node pair, we first define two similarity weights, one for geographical proximity and the other for traffic pattern similarity. For sensor networks, the geographical proximity between two nodes is determined as:

$$a_{g,ij} = \exp(-(\frac{dist_{ij}}{\delta_g})^2), \tag{1}$$

where $a_{g,ij}$ is the weight for geographical proximity between sensor i and j, $dist_{ij}$ is the travel distance between i and j and δ_g is the standard deviation of travel distances. For road networks, $a_{g,ij} = 1$ if two roads are connected and 0 otherwise. For both sensor and road networks, the traffic pattern similarity between two nodes is defined as:

$$a_{s,ij} = \exp(-(\frac{\|p_i - p_j\|}{\delta_s})^2), \tag{2}$$

where $a_{s,ij}$ is the weight for pattern similarity between node i and j, p_i and p_j are the historical traffic time series of nodes i and j and δ_s is the standard deviation of time series differences. In cases of incomplete training data, the historical time series are defined based upon time periods when both node i and j are observed. Given geographical and traffic pattern similarity weights, we define the adjacency matrix $\boldsymbol{A}$ as:

$$\boldsymbol{A}_{ij} = \begin{cases} 1 & a_{g,ij} > \kappa_g \text{ and } a_{s,ij} > \kappa_s, \\ 0 & a_{g,ij} \le \kappa_g \text{ or } a_{s,ij} \le \kappa_s, \end{cases} \tag{3}$$

where $\boldsymbol{A}_{ij}$ denotes the connectivity between node i and j and κ_g, κ_s are the thresholds to control for the sparsity of the adjacency matrix. Note that if there are fewer than two time periods when node i and j are observed simultaneously, $a_{s,ij}$ is set as 1. In this way, $\boldsymbol{A}_{ij}$ depends only on geographic proximity.

3.2. Problem definition

The goal of traffic data imputation is to recover missing values given observed values from a traffic network. Note that we pre-assume the stationary distribution of the traffic data and the missing values. For each node v in the traffic network, we denote its observed value at timestamp t as $x_t^v \in \mathbb{R}$. If no data is observed, $x_t^v = 0$. At each time t, the traffic data observed on G is denoted as a graph signal $\boldsymbol{X}_t = \{x_t^1, x_t^2, \dots, x_t^N\}, \boldsymbol{X}_t \in \mathbb{R}^N$. Suppose the imputation time interval is $[1, T]$, the traffic data imputation problem aims to learn a function $F(*)$ that is able to reconstruct the complete traffic data $\widetilde{\boldsymbol{X}}_{1:T} \in \mathbb{R}^{N\times T}$ given observed data $\boldsymbol{X}_{1:T} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \dots \boldsymbol{X}_T\}, \boldsymbol{X}_{1:T} \in \mathbb{R}^{N\times T}$ and the graph structure G:

$$\widetilde{\boldsymbol{X}}_{1:T} = F(\boldsymbol{X}_{1:T}, G). \tag{4}$$

4. Methodology

In this section, we elaborate on our proposed model for traffic data imputation. As shown in Fig. 2, MDGCN is composed of S spatiotemporal blocks (ST-blocks) and an output layer. ST-blocks are used to retrieve spatiotemporal patterns from the observed

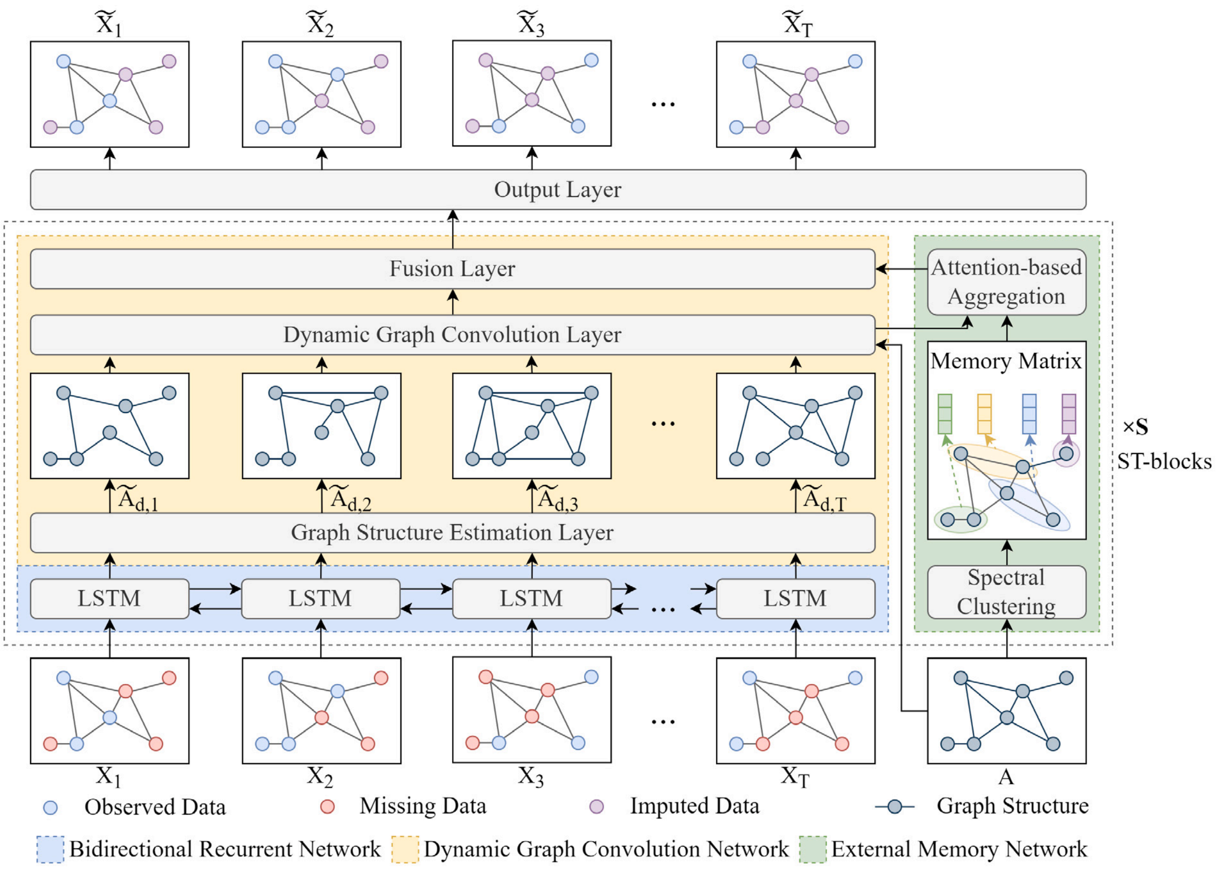

Fig. 2. The architecture of MDGCN.

traffic data. Each ST-block comprises a bidirectional recurrent network to capture temporal information (see Section 4.1) and a dynamic graph convolution network to capture spatial information (see Section 4.2). To model spatial dependencies evolving with time, a graph structure estimation layer is incorporated in the graph convolution network to learn adjacency matrices directly from traffic data. In addition, an external memory network is introduced to directly store and share global spatiotemporal information across the traffic network (see Section 4.3). The output layer is a feed-forward network which maps the output representations of the ST-blocks to the imputation results. The details of each module are described as follows.

4.1. Bidirectional recurrent network to capture temporal information

Existing GNN-based approaches for spatiotemporal prediction or imputation tasks typically employ variants of RNNs or CNNs to capture temporal dependencies. Through empirical comparisons, we choose LSTM, which achieves relatively good performance in our specific application. This is likely because LSTM is better at capturing long-term dependencies in time series and can benefit the imputation performance especially when adjacent temporal information is missing. A basic LSTM network is unidirectional, which is not suitable for the traffic data imputation problem since it can only utilize the temporal information prior to the missing time interval. To tackle this issue, we extend the unidirectional LSTM to bidirectional LSTM (BLSTM) by using two LSTM networks, one in the forward time direction and the other in the backward direction. In this way, the model is capable of exploiting information from both the past and the future (Berglund et al., 2015).

The BLSTM network is applied to each node separately and identically. For each node v, the input of the BLSTM network is a sequence of feature vectors, $\boldsymbol{Z}^v = \{\boldsymbol{z}_1^v, \boldsymbol{z}_2^v, \dots \boldsymbol{z}_T^v\}$, where $\boldsymbol{z}_t^v$ denotes the feature vector of node v at time t. For the BLSTM network in the first ST-block, $\boldsymbol{z}_t^v = x_t^v, \boldsymbol{z}_t^v \in \mathbb{R}^1$. If no data is observed, $x_t^v = 0$. At each time step, the BLSTM network recursively takes z_t^v as the input vector and summarizes the historical and future information into two hidden state vectors $\boldsymbol{h}_{f,t}, \boldsymbol{h}_{b,t}$ using

$$\boldsymbol{h}_{f,t}, \boldsymbol{c}_{f,t} = LSTM(\boldsymbol{z}_t^v, \boldsymbol{h}_{f,t-1}, \boldsymbol{c}_{f,t-1}), \tag{5}$$

$$\boldsymbol{h}_{b,t}, \boldsymbol{c}_{b,t} = LSTM(\boldsymbol{z}_t^v, \boldsymbol{h}_{b,t+1}, \boldsymbol{c}_{b,t+1}), \tag{6}$$

where $\boldsymbol{h}_{f,t}, \boldsymbol{c}_{f,t}$ are the cell memory state vector and hidden state vector in the forward direction, $\boldsymbol{h}_{b,t}, \boldsymbol{c}_{b,t}$ are state vectors in the backward direction. To integrate the learned features from both directions, the forward and backward hidden state vectors are concatenated and mapped to the output vector $\boldsymbol{z}^{v}_{l,t}$ using a simple linear layer, given as:

$$\boldsymbol{z}^{v}_{l,t} = \boldsymbol{W}_{l}[\boldsymbol{h}_{f,t}; \boldsymbol{h}_{b,t}] + \boldsymbol{\beta}_{l}, \tag{7}$$

where $\boldsymbol{W}_{l}$ is the parameter matrix for linear transformation, $\boldsymbol{\beta}_{l}$ is the biased term.

4.2. Dynamic graph convolution network to capture spatial information

Previous research has shown that spatial dependencies of traffic data are related to directions on the traffic network (Liang and Zhao, 2021). To capture the stochastic spatial dependencies across different directions, we introduce a Dynamic Graph Convolution Network (DDGCN) based on the Diffusion Graph Convolutional Network (DGCN) proposed by Li et al. (2017), which models traffic flow as a diffusion process. The working mechanism of the original DGCN is:

$$DGCN(\boldsymbol{H}) = \sum_{k=1}^{K} \left(F_k(\boldsymbol{A}_f)\boldsymbol{H}\boldsymbol{\theta}^{k}_{f} + F_k(\boldsymbol{A}_b)\boldsymbol{H}\boldsymbol{\theta}^{k}_{b} \right), \tag{8}$$

where $\boldsymbol{H} \in \mathbb{R}^{N\times c_{in}}$ is the input graph signals, $\boldsymbol{A}_f = \boldsymbol{A}/rowsum(\boldsymbol{A})$, $\boldsymbol{A}_b = \boldsymbol{A}^T/rowsum(\boldsymbol{A}^T)$ are the forward and backward transition matrices respectively, K is the number of diffusion steps, $\boldsymbol{\theta}^{k}_{f}, \boldsymbol{\theta}^{k}_{b} \in \mathbb{R}^{c_{in}\times c_{out}}$ are learning parameters of the kth diffusion step that defines how the received spatial information of each node is transformed, c_{in}, c_{out} are the input and output vector dimensions of each node. $F_k(\boldsymbol{A})$ is the recursive generating function for the Chebyshev polynomials given as $F_k(\boldsymbol{A}) = 2\boldsymbol{A}F_{k-1}(\boldsymbol{A}) - F_{k-2}(\boldsymbol{A})$, $F_0(\boldsymbol{A}) = \boldsymbol{I}$, and $F_1(\boldsymbol{A}) = \boldsymbol{A}$.

In our work, the input of the graph convolution network is a set of node features learned from the BLSTM network. The learned features from BLSTM at each time step t is denoted as $\boldsymbol{Z}_{l,t} = \{\boldsymbol{z}^{1}_{l,t}, \boldsymbol{z}^{2}_{l,t}...\boldsymbol{z}^{N}_{l,t}\}$. For each time step t, in addition to the fixed transition matrices $\boldsymbol{A}_f, \boldsymbol{A}_b$, we learn a dynamic adjacency matrix $\widetilde{\boldsymbol{A}}_{d,t}$ directly from $\boldsymbol{Z}_{l,t}$ using a graph structure estimation layer. The graph structure estimation layer consists of two linear transformations with a ReLU activation in between:

$$\widetilde{\boldsymbol{A}}_{d,t} = SoftMax(ReLU(\boldsymbol{Z}_{l,t}\boldsymbol{W}^{1}_{d} + \boldsymbol{\beta}^{1}_{d})\boldsymbol{W}^{2}_{d} + \boldsymbol{\beta}^{2}_{d}), \tag{9}$$

where $\boldsymbol{W}^{1}_{d} \in \mathbb{R}^{c_{in}\times c_h}, \boldsymbol{W}^{2}_{d} \in \mathbb{R}^{c_h\times N}$ are the parameter matrices for linear transformation and $\boldsymbol{\beta}^{1}_{d} \in \mathbb{R}^{c_h}, \boldsymbol{\beta}^{2}_{d} \in \mathbb{R}^{N}$ are bias terms. We apply the $Softmax$ function to normalize the learned adjacency matrix. Given the fixed transition matrices $\boldsymbol{A}_f, \boldsymbol{A}_b$ and the learned dynamic transition matrices $\widetilde{\boldsymbol{A}}_{d,t}$, we formulate the dynamic graph convolution layer at time step t as:

$$DDGCN(\boldsymbol{Z}_{l,t}) = \sum_{k=1}^{K} \left(F_k(\boldsymbol{A}_f)\boldsymbol{Z}_{l,t}\boldsymbol{\theta}^{k}_{f} + F_k(\boldsymbol{A}_b)\boldsymbol{Z}_{l,t}\boldsymbol{\theta}^{k}_{b} + F_k(\widetilde{\boldsymbol{A}}_{d,t})\boldsymbol{Z}_{l,t}\boldsymbol{\theta}^{k}_{t} \right), \tag{10}$$

where $\boldsymbol{\theta}^{k}_{t} \in \mathbb{R}^{c_{in}\times c_{out}}$ is the parameter matrix of the kth diffusion step defining how the received information is transformed based on the learned dynamic adjacency matrix.

The input of the DDGCN network is a graph-structured time series. To process multiple time steps in parallel, we extend DDGCN to multi-dimensional tensors, given as:

$$\begin{gathered}\widetilde{\boldsymbol{A}}_{d,1:T} = SoftMax(ReLU(\boldsymbol{Z}_{l,1:T}\boldsymbol{W}^{1}_{d} + \boldsymbol{\beta}^{1}_{d})\boldsymbol{W}^{2}_{d} + \boldsymbol{\beta}^{2}_{d}), \\ \boldsymbol{Z}_{g,1:T} = \sum_{k=1}^{K} \left(F_k(\boldsymbol{A}_{f,1:T}) \otimes \boldsymbol{Z}_{l,1:T}\boldsymbol{\theta}^{k}_{f} + F_k(\boldsymbol{A}_{b,1:T}) \otimes \boldsymbol{Z}_{l,1:T}\boldsymbol{\theta}^{k}_{b} + F_k \otimes (\widetilde{\boldsymbol{A}}_{d,1:T})\boldsymbol{Z}_{l,1:T}\boldsymbol{\theta}^{k}_{t} \right),\end{gathered} \tag{11}$$

where $\otimes$ denotes the operation of batch matrix multiplication, $\boldsymbol{Z}_{l,1:T} \in \mathbb{R}^{T\times N\times c_{in}}$ is the learned features from the BLSTM layer across T time slots, $\boldsymbol{A}_{f,1:T}, \boldsymbol{A}_{b,1:T}, \widetilde{\boldsymbol{A}}_{d,1:T} \in \mathbb{R}^{T\times N\times N}$ are the adjacency matrices and $\boldsymbol{Z}_{g,1:T} \in \mathbb{R}^{T\times N\times c_{out}}$ is the output of the DDGCN layer.

In the case when no pre-defined graph structure is provided, we use only the learned adjacency matrices to model spatial dependencies, given as:

$$\boldsymbol{Z}_{g,1:T} = \sum_{k=1}^{K} \left(F_k(\widetilde{\boldsymbol{A}}_{d,1:T}) \otimes \boldsymbol{Z}_{l,1:T}\boldsymbol{\theta}^{k}_{t} \right). \tag{12}$$

To integrate the spatial and temporal information learned from the DDGCN and BLSTM networks, we employ a residual connection (He et al., 2016) at the end of DDGCN, given as:

$$\boldsymbol{Z}_{r,1:T} = \boldsymbol{Z}_{l,1:T} + \boldsymbol{Z}_{g,1:T}. \tag{13}$$

4.3. External memory network to directly model global spatiotemporal information

Inspired by the use of memory neural networks to uncover shareable information across cities (Yao et al., 2019), in this section, we introduce an external memory network to capture the global spatiotemporal information from different nodes at different time steps. The main idea is to store the spatiotemporal features shared by different nodes in a global memory, which can be further transferred to missing data points to improve imputation accuracy. However, not every nodes share similar traffic patterns and it might easily introduce noise information if we simply compress the information from all nodes to a single representation. To avoid attending unrelated nodes that are not helpful for imputation, we first cluster different nodes into Q groups according to their similarity. Specifically, we utilize the spectral clustering method on the adjacency matrix defined in Section 3.1, and get a partition of the transportation network. When the graph structure is unavailable, we propose to cluster nodes based on only traffic pattern similarities. Each node v then corresponds to a group denoted as g_v. Different nodes in the same group are expected to share similar spatiotemporal patterns, and thus can transfer knowledge to each other. Then a memory matrix $\boldsymbol{M} \in \mathbb{R}^{Q\times c_m}$ is constructed, where each row $\boldsymbol{M}_g \in \mathbb{R}^{c_m}$ represents the shareable information of group g and c_m is the dimension of the representation vector. The memory matrix $\boldsymbol{M}$ is learned together with the model.

Based on the learned memory, an intuitive operation is to use the representation of the corresponding group to represent the global information for each node. However, as traffic data is dynamically changing, the relationships between nodes and different clusters also change. To capture such dynamics, we use an attention mechanism to adaptively transfer the pattern knowledge of different clusters to the target node. Specifically, for each node v at time step t, we first generate a query vector $\boldsymbol{q}_t^v \in \mathbb{R}^{c_m}$ based on $\boldsymbol{z}_{r,t}^v$ using a simple linear layer:

$$\boldsymbol{q}_t^v = \boldsymbol{z}_{r,t}^v \boldsymbol{W}_q + \boldsymbol{b}_q. \tag{14}$$

where $\boldsymbol{W}_q \in \mathbb{R}^{c_{out}\times c_m}, \boldsymbol{b}_q \in \mathbb{R}^{c_m}$ are the parameters for linear transformation. The similarity between the query vector $\boldsymbol{q}_t^v$ and the pattern representations of different clusters is then computed as:

$$\boldsymbol{s}_t^v = SoftMax(\boldsymbol{q}_t^v \boldsymbol{M}^T), \tag{15}$$

where $\boldsymbol{s}_t^v \in \mathbb{R}^Q$ is the learned attention weights with each element $s_{t,g}^v$ denoting the similarity of the patterns of node v and group g at time step t. Finally, the global spatiotemporal information for node i at time t is represented as a weighted sum of the representations of each cluster, given as:

$$\boldsymbol{g}_t^v = \sum_{g=1}^{Q} s_{t,g}^v \boldsymbol{M}_g, \tag{16}$$

where $\boldsymbol{M}_g \in \mathbb{R}^{c_m}$ is the learned representation of the gth cluster in memory $\boldsymbol{M}$ and $\boldsymbol{g}_t^v \in \mathbb{R}^{c_m}$ is the final representation of the global spatiotemporal information for node v at time t.

Finally, we use a fusion layer to integrate the learned spatiotemporal information from different modules:

$$\boldsymbol{z}_{o,t}^v = LayerNorm(ReLU(\boldsymbol{W}_g concat(\boldsymbol{g}_t^v, \boldsymbol{z}_{r,t}^v) + \boldsymbol{\beta}_g)), \tag{17}$$

where $concat(*)$ denotes the concatenation operation, $\boldsymbol{W}_g \in \mathbb{R}^{(c_m+c_{out})\times c_b}, \boldsymbol{\beta}_g \in \mathbb{R}^{c_b}$ are the learned parameters, c_b is the learned vector dimension. A layer normalization layer (Ba et al., 2016) denoted as $LayerNorm(*)$ is added to stabilize the model parameters. The output vector $\boldsymbol{z}_{o,t}^v$ will then serve as the input of the next ST-block or the output layer.

Following Yao et al. (2019), a clustering loss is defined to enforce the attention weights to be consistent with our pre-defined categories. The formulation of the clustering loss is as follows:

$$L_{cluster} = -\sum_{i=1}^{S}\sum_{t=1}^{T}\sum_{v=1}^{N} \boldsymbol{o}_v \log(\boldsymbol{s}_{t,i}^v), \tag{18}$$

where S is the number of ST-blocks, $\boldsymbol{o}_v \in \mathbb{R}^Q$ is a one-hot vector denoting the corresponding cluster of node v and $\boldsymbol{s}_{t,i}^v$ is the learned attention weights for node v at time step t in the ith ST-block.

4.4. Model setup

4.4.1. Loss function

The loss function is comprised of two parts, one is the reconstruction error of the traffic data and the other is the clustering loss defined as Eq. (18). To make our trained model more general for all nodes in the traffic network, the reconstruction error is defined on both observed and missing values following (Wu et al., 2020):

$$L_{mse} = \sum_{v=1}^{N}\sum_{t=1}^{T} (\tilde{x}_t^v - \hat{x}_t^v)^2, \tag{19}$$

where $\tilde{x}_t^v, \hat{x}_t^v$ represent the true and estimated values of node v at time t respectively. The loss function is then defined as:

$$L(\theta) = L_{mse} + \lambda L_{cluster}, \tag{20}$$

where λ is a trade-off weight to balance different loss.

4.4.2. Training data generation

To make our model more robust to different missing ratios, we use Alg. 1 to generate random training samples from training data. The key idea is to randomly generate a subset of training data $\boldsymbol{X}_{sample}$ and a binary mask matrix $\boldsymbol{E}_{sample}$ for model training. $\boldsymbol{X}_{train} = \boldsymbol{X}_{sample} \odot \boldsymbol{E}_{sample}$ so that missing values are masked as zeros. $\odot$ denotes the element-wise multiplication operation.

Algorithm 1: Generating training samples

```
Input : historical data $\boldsymbol{X} \in \mathbb{R}^{N \times P}$ over period $[1, P]$ for training, missing type $Y$, the size of imputation window $T$, batch size
        $B$, training iteration $I$
for $i$ in $\{1, 2, ...I\}$ do
    Randomly choose a time point $t$ within range $[1, P - T]$;
    Obtain sampled data $\boldsymbol{X}_{sample} = \boldsymbol{X}_{t:t+T}$ from $\boldsymbol{X}$;
    for $j$ in $\{1, 2, ...B\}$ do
        Randomly generate a missing ratio $r$ within range $(0, 1)$;
        Generate a mask matrix $\boldsymbol{E}_{sample}$ according to $Y$ and $r$;
        $\boldsymbol{X}_{train} = \boldsymbol{X}_{sample} \odot \boldsymbol{E}_{sample}$;
    end
    Use sets $\{\boldsymbol{X}^{1:B}_{train}\}$ to train DSTCGN;
end
```

5. Experiments

5.1. Data description

In this study, we conduct experiments on two public traffic datasets, one is collected on a freeway network and the other is collected on an urban road network.

METR-LA (Jagadish et al., 2014) is a traffic speed dataset[1] collected from 207 loop sensors on a freeway network of Los Angeles. The data ranges from 2012-03-01 to 2012-06-30 with a sampling rate of 5 min. To compute the adjacency matrix $\boldsymbol{A}$, we define $\kappa_g = 0.1, \kappa_s = 0.1$ through experiments.

INRIX-SEA (Cui et al., 2020) is a traffic speed dataset[2] collected from multiple data sources including GPS probes, road sensors and cell phone data on a road network in the Seattle downtown area. The data ranges from 2012-01-01 to 2012-12-31 with a sampling rate of 5 min. In this research, we choose a sample of road networks consisting of 223 connected road links for experiments. Our preliminary research shows that urban road networks are usually heterophilous, where connected road links often exhibit different patterns. Therefore, we define $\kappa_g = 0., \kappa_s = 0.8$ for the computation of $\boldsymbol{A}$ through experiments.

5.2. Missing pattern generation

Following Li et al. (2018), we define four types of missing patterns. Methods for generating different missing patterns are described in Appendix A.

Random Missing (RM) (Fig. 1a): missing values are completely independent of each other and displayed as randomly scattered points for each sensor (or road link). This may be due to temporary failure (e.g., power outage, communication error) for stationary sensors and the uncertainty of movement for mobile sensors.

Temporally Correlated Missing (TCM) (Fig. 1b): missing values are dependent in the time dimension and appear as a consecutive time interval for each sensor (or road). For stationary sensors, this can be caused by long-term physical damage and maintenance backlog (Li et al., 2018). For mobile sensors this may happen when a road has no GPS probes passing by for a long time.

Spatially Correlated Missing (SCM) (Fig. 1c): missing values are dependent in the spatial dimension and appear at neighboring sensors or connected road links for each time slot. For stationary sensors this may occur due to regional power outage or communication problems. For mobile sensors this may happen for urban areas with low traffic volumes.

Block Missing (BM) (Fig. 1d): missing values are dependent at both spatial and temporal dimensions. In this scenario, values are missing at consecutive time intervals and spatial neighboring locations. For stationary sensors this is often caused by regional long-term malfunction. For mobile sensors this is common at mid-night when few GPS probes are working on the road network.

5.3. Baselines

In the numerical experiment, we compare MDGCN with several state-of-the-art deep learning-based methods. Two groups of baselines are included. The first group contains existing imputation methods:

Denoising Autoencoder (DAE) (Duan et al., 2016): a deep learning strategy that regards the traffic states of each sensor (or road) as a vector and uses stacked DAEs to impute missing values.

[1] https://github.com/liyaguang/DCRNN
[2] https://github.com/zhiyongc/Graph-Markov-Network

Bidirectional LSTM (BiLSTM)[3] (Berglund et al., 2015): a recurrent architecture that predicts the missing values using LSTM model in the forward and backward time directions simultaneously.

Bidirectional Recurrent Imputation for Time Series (BRITS) (Cao et al., 2018): a novel imputation method for time series which utilizes inter-node feature correlations in a bidirectional recurrent system.

Convolution Bidirectional LSTM (CNN-BiLSTM) (Asadi and Regan, 2019): a convolution recurrent autoencoder that learns the spatial information using multi-range convolutions and temporal information using bidirectional LSTM layers.

Graph Attention Convolutional Network (GACN) (Ye et al., 2021): a graph learning approach using graph attention mechanisms to learn spatial correlations and standard convolution layers to learn temporal correlations.

The second group contains state-of-the-art GCN-based models in traffic research. Note that these models were designed for prediction tasks and we have made every effort to adapt these baseline models for the imputation problem.

STGCN (Yu et al., 2017): a convolution framework which uses GCNs to extract spatial features and gated CNNs to extract temporal features. In this model, the adjacency matrix is considered as prior knowledge and fixed throughout training.

GWNET (Wu et al., 2019): a graph neural network which captures spatial dependencies with generalized diffusion graph convolution layers and temporal dependencies with dilated convolution layers. An adaptive adjacency matrix is learned through node embedding to capture the hidden spatial dependencies in traffic data.

GMAN (Zheng et al., 2020): a graph multi-attention network which uses attention mechanisms to capture spatial and temporal correlations. A spatial attention mechanism is proposed to model dynamic relevance between nodes based on real-time traffic information and graph structures.

5.4. Experiment settings

All experiments are conducted on a NVIDIA 1080 Ti GPU. We use data from the first 60% time slots as the training set, the following 20% as the validation set and the last 20% as the test set. We choose $T = 72$ time steps (i.e., 5 min × 72 = 6 h) as the imputation window. During training, we randomly generate training samples from the training set using Alg. 1. For each epoch, we set the number of training iterations I as 80 and batch size B as 4. We fix the number of epochs to 400 for all experiments. All models are trained using Adam optimizer with an initial learning rate of 0.001 and we decrease the learning rate to 0.0001 after 200 epochs. During validation and test, the imputation is conducted using a sliding-window approach on: $[t, t+T), [t+T, t+2T), [t+2T, t+3T)$, etc.

The hyperparameters of our proposed model is tuned over the validation set using METR-LA data under RM scenarios. Specifically, we first fix the number of ST-blocks S as 2, the output dimension of the graph convolution layer as 64 and the hidden dimension of the output layer as 128 following Yu et al. (2017). For the BLSTM layer, we set the hidden state dimension as 128 and the output dimension as 64, which is the same as the DDGCN layer to facilitate residual connection. The diffusion step of the DDGCN layer layer K is set as 2 following the original setting of Li et al. (2017). To estimate dynamic adjacency matrices, the hidden dimension of the feed-forward network is selected from [128, 256, 512] and finally set as 256. For the external memory network, we fix the memory vector dimension c_m as 64, which is the same as the output dimension of BLSTM and DDGCN layers. The number of clusters Q is tuned from 10 to 30 with a step size of 10. We find that the number of clusters does not influence the results much and we use $Q = 30$ in our experiments. For loss construction, we choose λ from $1e-5, 1e-3, 1e-1$ and the model achieves the best performance when $\lambda = 1e-3$. For DAE, we set 3 hidden layers with a hidden dimension of 256. For BiLSTM, we use the same settings as the BLSTM layer in our proposed model. For BRITS, we set the hidden dimension of the RNN cells as 128 and use the same settings of Cao et al. (2018) for the other hyperparameters. We implement CNN-BiLSTM and GACN ourselves using the same network hyperparameters reported in their papers. For STGCN, GWNET and GWAN, we adapt them to our specific application based on their public codes. For GWNET, we replace its original temporal convolution layer with the one proposed by Yu et al. (2017), which achieves better performance in our specific application. For GWAN, we decrease the number of spatiotemporal blocks to 2 due to the spatial limits of our computing resources. The mean absolute error (MAE), the root-mean-squared error (RMSE) and the mean absolute percentage error (MAPE) are used to evaluate model performance.

6. Results

6.1. Imputation performance analysis

In this section, we compare the performance of MDGCN with the baseline models for different missing patterns and missing ratios ranging from 20% to 80%. Figs. 3 and 4 display the imputation performance of different models on METR-LA data and INRIX-SEA data respectively. All the results are averaged over 3 independent runs. For each missing pattern and missing ratio, the models are evaluated on the test dataset with 5 randomly generated missing masks. We can find that MDGCN achieves notably superior results to the baseline models in most missing scenarios. For the METR-LA data, MDGCN performs better than the baseline methods by a large margin for all missing patterns and missing ratios. For the INRIX-SEA dataset, MDGCN achieves significant improvement in the imputation performance for missing patterns RM, TCM and SCM. For BM, MDGCN shows similarly competitive performance with BiLSTM. This indicates that MDGCN can provide more accurate and robust results than existing methods for

[3] For distinction, BLSTM denotes the bidirectional recurrent layer in MDGCN and BiLSTM denotes the bidirectional recurrent baseline model.

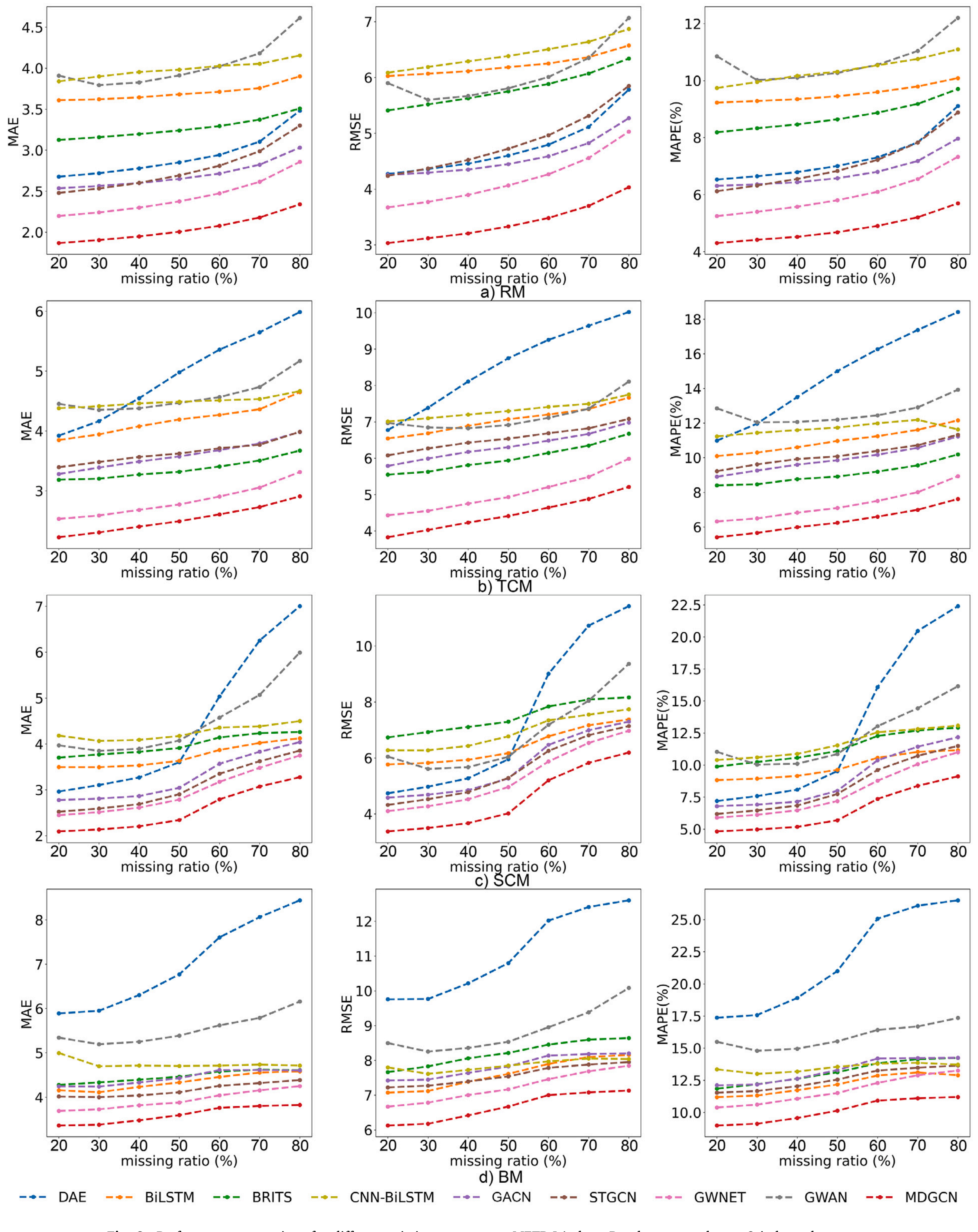

Fig. 3. Performance comparison for different missing patterns on METR-LA data. Results averaged over 3 independent runs.

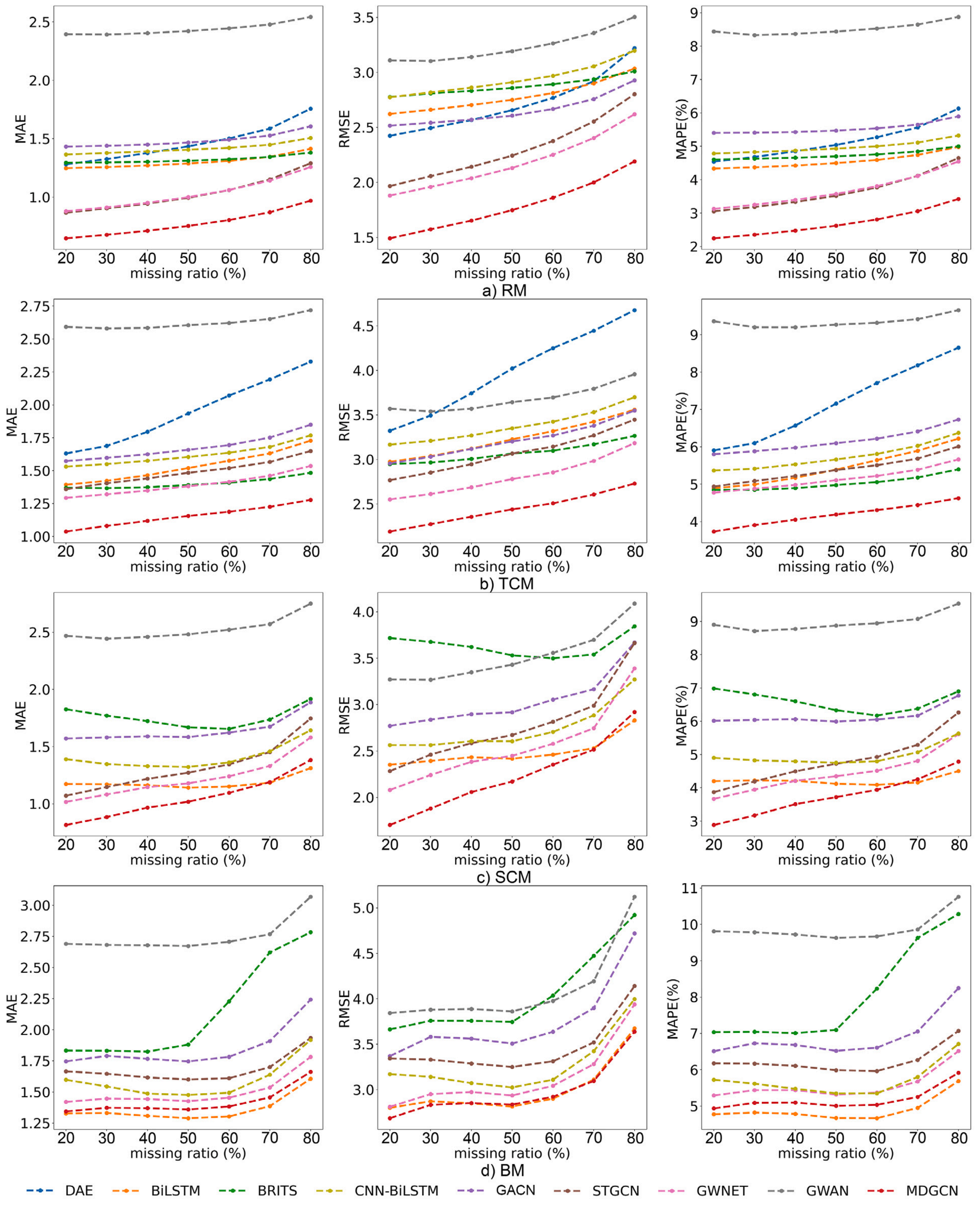

Fig. 4. Performance comparison for different missing patterns on INRIX-SEA data. Results averaged over 3 independent runs. The results of DAE for SCM and BM on INRIX-SEA data is presented separately in Appendix B.

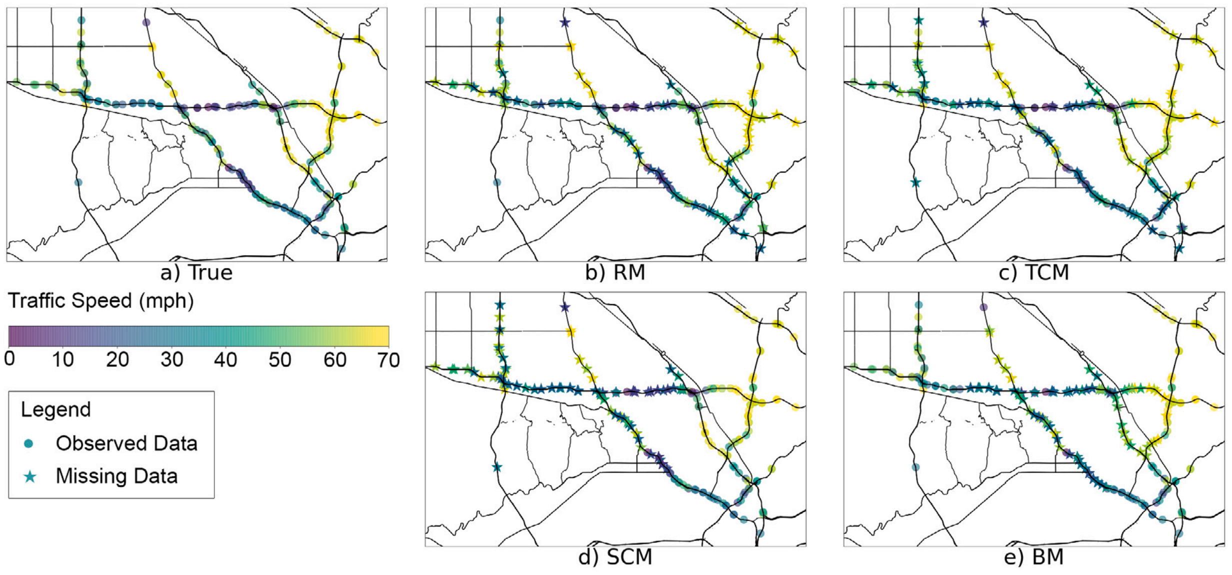

Fig. 5. The imputation result of the METR-LA data at 18:00 on 2012-05-24 using MDGCN.

diverse combinations of missing patterns and traffic data types. This is likely because our model can better leverage global and local spatiotemporal information hidden in the complete traffic data. In our model, the temporal information is captured with bidirectional recurrence, the spatial information with directed graph convolutions and the global spatiotemporal information with an external memory network. We also take advantage of the graph structure estimation technique which effectively models spatial dependencies evolving over time. It should be noted that on INRIX-SEA data, BiLSTM can slightly outperform MDGCN when the missing pattern is BM or when the missing ratio reaches 80% for SCM. This can be potentially explained that when spatially adjacent information is largely missing, it is difficult for GCNs to uncover meaningful spatial information and may even introduce noise to the imputation results. The effect seems to be more pronounced in urban road networks (e.g., INTRIX-SEA) where the presence of traffic signals can further disrupt the spatial dependencies between links. Fig. 5 provides an example of the imputation result of MDGCN in different missing scenarios with the missing ratio of 50%. It can be seen that MDGCN is capable of reconstructing missing values in all kinds of missing patterns.

Apart from our proposed model, we also examine the performance of baseline models in different missing patterns, which has not been fully discussed in literature.

DAE: Although DAE performs well in RM, in the other missing patterns, its performance degrades tremendously with the increase of missing ratios. The poor performance of DAE in the TCM scenario is potentially because DAE regards the data of each node as a vector, which is suitable to repair isolated missing data points but fails to recover consecutive missing intervals. Additionally, DAE barely leverages spatial information, thus performs poorly in SCE and BE scenarios.

BiLSTM: Although BiLSTM provides competitive results on INRIX-SEA data, it performs relatively poorly on METR-LA data. This suggests that traffic conditions on urban road networks might display strong temporal dependencies, whereas traffic conditions on freeway networks have lower regularities on time axis.

BRITS: BRITS provides better performance than BiLSTM for RM and TCM on both datasets, validating its effectiveness in capturing feature correlations in these scenarios. However, it fails to provide competitive performance in other scenarios, suggesting that it cannot work well for all missing patterns.

CNN-BiLSTM: The performance of CNN-BiLSTM is consistently inferior to BiLSTM. This indicates that CNNs cannot capture the spatial dependencies on traffic networks effectively and even have a negative effect on the model performance.

GACN: The performance of GACN is relatively poor compared with other GCN-based approaches. This is potentially because it uses a standard convolution layer to capture temporal dependencies, which can only leverage adjacent temporal information. In addition, it only considers spatial dependencies between pre-defined adjacent nodes.

STGCN: STGCN performs well on METR-LA data, but it fails to provide satisfying results for INRIX-SEA data. This can be explained that STGCN relies on a fixed adjacency matrix pre-determined by road connectivity to model spatial correlations and fails to capture the true spatial dependencies in urban road networks.

GWNET: GWNET shows better performance than STGCN in all missing scenarios, validating the effectiveness of the adaptive adjacency matrix learned through node embedding. Compared with GWNET, our proposed model performs better, suggesting that our graph structure estimation technique can more effectively uncover nonlinear features from traffic data and model true and dynamic spatial dependencies.

GMAN: Although GMAN was reported to outperform STGCN and GWNET in the task of traffic prediction, its performance is worse than STGCN and GWNET in all kinds of missing patterns for both data sets. This indicates that although the attention mechanism

Table 2
Performance (RMSE/MAPE) comparison of different variants of the external memory network on METR-LA data. Results averaged over 3 independent runs.

Model		Missing ratios						
		20%	30%	40%	50%	60%	70%	80%
RM	-Memory	3.072/4.37	3.157/4.49	3.246/4.60	3.373/4.76	3.527/4.98	3.753/5.30	4.096/5.80
	-Cluster	3.062/4.35	3.147/4.47	3.235/4.57	3.360/4.73	3.513/4.95	3.735/5.26	4.075/5.75
	-Attn	3.056/4.33	3.142/4.45	3.229/4.55	3.355/4.72	3.507/4.94	3.729/5.25	4.066/5.75
	MDGCN	*3.033/4.30*	*3.120/4.42*	*3.207/4.52*	*3.331/4.68*	*3.483/4.90*	*3.700/5.20*	*4.035/5.69*
TCM	-Memory	3.963/5.83	4.161/5.96	4.384/6.20	4.576/6.47	4.819/6.83	5.057/7.23	5.388/7.84
	-Cluster	3.878/5.39	4.064/*5.63*	4.271/*5.96*	4.448/*6.20*	4.680/*6.55*	4.919/*6.95*	5.240/*7.55*
	-Attn	3.858/*5.38*	4.041/5.64	4.245/5.98	4.420/6.23	4.651/6.59	4.890/6.98	5.218/7.58
	MDGCN	*3.828*/5.36	*4.024*/5.61	*4.231*/5.94	*4.410*/6.19	*4.643*/6.55	*4.879*/6.94	*5.212*/7.55
SCM	-Memory	3.401/4.91	3.521/5.06	3.695/5.27	4.073/5.86	5.483/8.12	6.087/9.13	6.419/9.83
	-Cluster	3.394/4.85	3.513/5.00	3.683/5.20	4.050/5.76	5.330/7.64	5.873/8.45	6.194/*9.10*
	-Attn	3.380/4.87	*3.500*/5.02	3.672/5.23	4.027/5.77	5.209/7.47	*5.806*/8.46	*6.189*/9.24
	MDGCN	*3.379/4.83*	*3.500/4.97*	*3.671/5.18*	*4.023/5.69*	*5.205/7.36*	5.826/*8.38*	6.195/*9.12*
BM	-Memory	6.253/9.49	6.347/9.64	6.592/10.13	6.833/10.75	7.231/11.73	7.289/11.84	7.333/11.90
	-Cluster	6.222/9.38	6.296/9.52	6.521/9.96	6.776/10.63	7.150/11.62	7.175/11.64	7.243/11.73
	-Attn	*6.109*/9.03	*6.172*/9.17	*6.409*/9.59	*6.645*/10.18	*6.987*/11.02	*7.067*/11.26	7.148/11.40
	MDGCN	6.119/*9.01*	6.173/*9.13*	6.414/*9.56*	6.660/*10.13*	7.001/*10.92*	7.072/*11.10*	*7.132/11.20*

is capable of modeling spatiotemporal dependencies of complete traffic data, it may not be adequate to provide robust results for incomplete and heterogeneous traffic data.

In summary, none of the baseline models can provide competitive results for all missing patterns in both freeway and urban road networks. DAE and BRITS are not competitive for spatially correlated missing patterns whereas STGCN cannot achieve superior results when missing values are temporally correlated. For different data types, BiLSTM and CNN-BiLSTM perform relatively poorly for freeway networks, whereas existing GCN-based models, including GACN, STGCN, GWNET and GMAN, do not provide competitive results for urban road networks. Relative to the baseline, our proposed model achieves high accuracy across different missing scenarios for both freeway and urban road networks. This validates the robustness of our proposed model in capturing both global and local spatiotemporal dependencies with limited observed data across different scenarios.

6.2. Effect of external memory network

There are two main components of the external memory network: first, different nodes are clustered to several groups according to the pre-defined adjacency matrix to avoid information sharing between unrelated nodes; second, we use an attention mechanism to softly match each node to different clusters. To verify the effectiveness of the external memory network and its different components, we implement three variants of DMGCN on METR-LA data for ablation tests: one drops the entire external memory network and the other two employed a simplified version of the external memory network:

- **External memory network** ($-Memory$): without the external memory network, the variant model only utilizes temporal information with the BLSTM layer and spatial information with the DDGCN layer.
- **Spectral clustering** ($-Cluster$): without spectral clustering, each node is not assigned with a pre-defined cluster category. Each node is matched with different clusters using only the attention network.
- **Attention mechanism** ($-Attn$): without the attention mechanism, we use the representation of the corresponding category of each node to represent the global spatiotemporal information.

Table 2 displays the performance comparison of our proposed model and the three variants of MDGCN. To evaluate whether the improvement brought by the external memory network is statistically significant, we conduct a t-test based on 6 repetitive experiments and the result can be found in Appendix C. It is found that the external memory network contributes to the model performance for all missing patterns and missing ratios significantly. The improvement brought by the memory network is obvious for TCM and BM with all missing ratios. This indicates that the memory network can effectively capture shareable spatiotemporal information across different nodes. In addition, the memory network can also store long-term temporal information, which is beneficial to the imputation of temporal correlated missing patterns. For SCM, the contribution of the external memory network is large when the missing ratio is high. This is potentially because graph convolutions do not work well when spatially adjacent information is largely missing, and the memory network can utilize spatial information from different time slots to enhance the imputation performance. Removing either spectral clustering or the attention mechanism leads to worse performance for most missing scenarios, validating the importance of both components. Between the two, spectral clustering has a greater impact on different missing patterns. For BM, the variant model with only spectral clustering even achieves slightly better performance than MDGCN in most scenarios. This suggests that it is necessary to consider the pattern difference across different nodes for knowledge transfer.

Table 3
Performance (RMSE/MAPE) Comparison of Different Adjacency Matrix Configurations on METR-LA data. Results averaged over 3 independent runs.

Model		Missing ratios						
		20%	30%	40%	50%	60%	70%	80%
	Fix-only	3.102/4.41	3.191/4.53	3.283/4.64	3.418/4.81	3.582/5.05	3.822/5.38	4.194/5.92
	Fix-SAAM	3.090/4.39	3.176/4.50	3.269/4.62	3.397/4.78	3.554/5.01	3.785/5.34	4.131/5.87
RM	GAT	3.473/5.00	3.546/5.10	3.631/5.20	3.759/5.35	3.917/5.57	4.160/5.89	4.571/6.48
	Dynamic-only	3.108/4.41	3.195/4.53	3.285/4.64	3.410/4.80	3.565/5.03	3.785/5.33	4.119/5.83
	Fix-dynamic	*3.033/4.30*	*3.120/4.42*	*3.207/4.52*	*3.331/4.68*	*3.483/4.90*	*3.700/5.20*	*4.035/5.69*
	Fix-only	4.154/5.86	4.447/6.29	4.745/6.78	4.992/7.19	5.254/7.61	5.510/8.10	5.850/8.79
	Fix-SAAM	3.970/5.51	4.188/5.79	4.431/6.18	4.635/6.49	4.887/6.87	5.139/7.31	5.486/7.95
TCM	GAT	4.832/6.93	5.194/7.52	5.501/8.11	5.779/8.61	6.017/9.02	6.262/9.55	6.553/10.20
	Dynamic-only	3.968/5.54	4.135/5.77	4.350/6.12	4.534/6.38	4.746/6.71	4.980/7.12	5.304/7.73
	Fix-dynamic	*3.828/5.41*	*4.027/5.65*	*4.231/5.98*	*4.410/6.24*	*4.643/6.59*	*4.879/6.99*	*5.212/7.61*
	Fix-only	3.420/4.96	3.545/5.12	3.719/5.34	4.120/6.00	5.773/8.78	6.570/10.42	6.903/11.14
	Fix-SAAM	3.383/4.86	3.502/5.01	3.675/5.22	4.053/5.80	5.421/7.93	6.101/9.06	6.482/9.90
SCM	GAT	3.665/5.37	3.782/5.51	3.958/5.73	4.368/6.38	5.989/9.21	6.793/10.87	7.177/11.86
	Dynamic-only	3.483/5.02	3.609/5.17	3.783/5.39	4.202/6.00	5.350/7.66	5.878/8.54	*6.180*/9.19
	Fix-dynamic	*3.379/4.83*	*3.500/4.97*	*3.671/5.18*	*4.023/5.69*	*5.205/7.36*	*5.826/8.38*	6.195/*9.12*
	Fix-only	6.662/10.41	6.730/10.55	6.947/11.01	7.262/11.85	7.707/13.18	7.836/13.49	7.948/13.72
	Fix-SAAM	6.264/9.54	6.359/9.67	6.559/10.08	6.781/10.68	7.244/11.93	7.374/12.22	7.465/12.37
BM	GAT	6.962/11.14	7.044/11.28	7.327/12.02	7.527/12.71	8.060/14.23	8.186/14.43	8.214/14.45
	Dynamic-only	6.350/9.63	6.394/9.69	6.625/10.08	6.835/10.67	7.108/11.40	7.170/11.58	7.222/11.64
	Fix-dynamic	*6.119/9.01*	*6.173/9.13*	*6.414/9.56*	*6.660/10.13*	*7.001/10.92*	*7.072/11.10*	*7.132/11.20*

6.3. *Effect of dynamic adjacency matrices*

Recent studies have proposed different techniques to model the true and dynamic spatial dependencies across traffic networks, which can be mainly divided into two streams: one is to learn an adaptive adjacency matrix together with the model (Wu et al., 2019; Ye et al., 2020), and the other is to employ attention mechanisms to capture spatial dependencies evolving with time (Zheng et al., 2020; Ye et al., 2021). However, most of these approaches are developed for prediction tasks, and it is unknown whether they can properly handle the incomplete and heterogeneous data for imputation tasks. To compare the performance of our proposed graph structure estimation technique with existing approaches, we use the self-adaptive adjacency matrix ($SAAM$) proposed by Wu et al. (2019) as a representative of the first stream of studies, and the Graph attention networks (GAT) in Ye et al. (2021) as a representative of the second stream of studies. In addition, we conduct experiments using different variants of our proposed adjacency matrix configurations: fix-only ($[\boldsymbol{A}_f, \boldsymbol{A}_b]$), dynamic-only ($[\widetilde{\boldsymbol{A}}_d]$) and fix-dynamic ($[\boldsymbol{A}_f, \boldsymbol{A}_b, \widetilde{\boldsymbol{A}}_d]$). For fair comparison, we use the same structure of our proposed model and only replace the graph convolution layer with different alternatives.

Table 3 shows the performance comparison of different adjacency matrices on the METR-LA data. It can be found that the fix-dynamic model works better than the fix-only model under all missing scenarios. For all missing patterns, the impact of the dynamic adjacency matrices becomes greater as the missing ratio gets higher, suggesting that dynamic adjacency matrices play a more important role for higher missing ratios. The contribution brought by the dynamic adjacency matrices is especially significant for correlated missing patterns: without A_d, the RMSE increases by at most 13.2% for TCM, 12.8% for SCM and 11.4% for BM. We also find that the dynamic-only model works better than the fix-only model under most scenarios. This suggests that our proposed graph structure estimation technique can effectively uncover the true and dynamic spatial dependencies even when no pre-defined graph structure is provided. For SCM, the dynamic-only model performs even better than the fix-dynamic model when the missing ratio is 80%. This is potentially because when spatial information is largely missing, the fix adjacency matrices cannot pass useful messages to neighborhood nodes and even introduce some noise information. Compared with SAAM and GAT, our proposed graph structure estimation technique also performs best under all missing scenarios. This is potentially because the feed-forward network can effectively uncover nonlinear features from traffic data and model true and dynamic spatial dependencies, while SAAM fails to directly leverage real-time traffic information and does not change over time. GAT performs even worse than the fix-only model in our application. Recall that among our baselines, GMAN (Zheng et al., 2020) and GACN (Ye et al., 2021), which also employ attention mechanisms to capture spatial dependencies across the traffic network, do not perform as well as the other GCN-based models in our specific application. This indicates that although the attention mechanism is capable of modeling spatiotemporal dependencies of complete traffic data, it might fail to provide robust results for incomplete and heterogeneous traffic data.

To intuitively illustrate the dynamic adjacency matrices learned from the DDGCN layer, we visualize the learned transition weights between sensor A and the other sensors in METR-LA data across different time in Fig. 6. Each colored dot represents a sensor, and the lighter the color of a dot is, the higher the transition weight between A and the sensor is. Recall that transition weights represent the diffusion likelihood between sensors and higher transition weights indicate stronger spatial correlations. As displayed in Fig. 6, spatial correlations are not strictly determined by distance. For example, at 20:00, sensor B which is not adjacent to A displays high dependencies with sensor A. Comparing the transition weights across different time, we can find that sensor B has different correlations with A at different time, indicating that the spatial dependencies are dynamically changing over time. This confirms the necessity of capturing dynamic spatial dependencies.

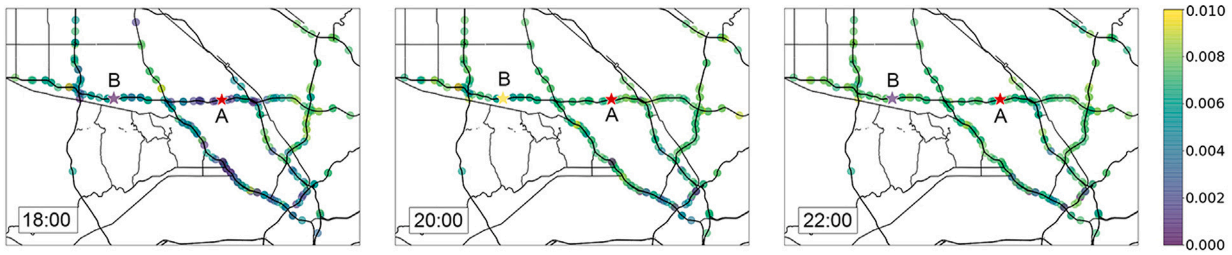

Fig. 6. Transition weights between sensor *A* and the other sensors in METR-LA data from 18:00 to 22:00 on 2012-05-24. Sensor B is marked as an example to highlight the dynamic transition weights between sensors A and B.

Table 4
Computation cost of GCN-based models on METR-LA data.

Model	Time		Space	
	Training (s/epoch)	Inference (s)	RAM used (GB)	GPU used (GB)
GACN	17.78	5.24	2.77	8.65
STGCN	4.57	1.43	2.80	1.38
GWNET	18.39	1.67	2.81	1.79
GMAN	34.09	5.04	2.79	10.06
MDGCN	16.81	4.71	2.88	3.71

6.4. Computation cost

In this section, we compare the time and space cost of MDGCN with GCN-based baselines, including GACN, STGCN, Graph WaveNet and GMAN. The results are displayed in Table 4. The inference time is measured as the total time cost on the test set. The space cost is measured as the RAM and GPU usage for running one epoch. It is shown that STGCN is the most efficient for both training and inference due to its simple architecture. Our proposed model runs two times faster than GMAN and similar to GACN and GWNET. For space consumption, the RAM usage of different models are similar, while the GPU usage of GACN and GMAN is much larger than the other models. This is potentially because they both employed attention mechanisms to capture spatial dependencies, which require node-to-node computation and consume much GPU space. These results show that our proposed model can achieve more accurate imputation performance with reasonable computation cost.

6.5. Experiments with and without complete training data

In prior sections, we use complete historical data for model training. Note that the missing ratio of traffic data can often vary over time. For example, when new traffic sensors are installed, the data is close to complete, but its missing ratio may increase as the sensors degrade over time. In such cases, we may take advantage of the (near)complete data in the beginning to train a deep learning model for imputation later on. However, it is not always possible to obtain sufficient complete data to train deep learning models. In cases when traffic sensors are damaged for a long period (e.g., several months) or few vehicles are equipped with GPS devices, only incomplete traffic data is available. In this section, we compare the performance of our proposed model with two selected baseline models under different data conditions, based on whether complete training data is available or not.

In recent years, tensor factorization has emerged as a popular approach to traffic data imputation. Generally, tensor factorization methods assume that the multivariate and multidimensional time series can be characterized by a low-rank structure with shared latent factors (Chen and Sun, 2021). It is expected that, when sufficient complete data is provided, deep learning models have a clear advantage over tensor factorization methods: first, they can capture the complex spatiotemporal dependencies hidden in the road network, and second, they can be pre-trained for online application. In cases with only incomplete data, tensor factorization models may offer more robust results, as they directly learn the low-rank structure from the incomplete data without the need for separate model training. To compare the performance of our proposed model and other methods under different data conditions, we use GWNET (Wu et al., 2019) as a representative of baseline deep learning models, and BGCP (Chen et al., 2019) as an example of tensor factorization models. Experiments are conducted on the METR-LA dataset for different missing patterns with missing ratios ranging from 20% to 60%.

Because deep learning and tensor factorization methods are fitted in very different ways, we design our experiment settings as follows. When there is complete training data, we train MDGCN and GWNET using the method described in Section 5.4. For BGCP, the entire data set (concatenated training, validation and test set) is used as input and the recovery results are generated at once. When only incomplete data is provided, we assume the entire data set has uniform missing distributions. For example, in the RM 60% scenario, both the training set and the test set have 60% randomly missing values. For the training of MDGCN and GWNET, the incomplete training set with missing values is fed to Alg. 1 to generate training samples. It should be noted that the missing values are kept as unobserved during the whole training process, so that MDGCN can only use the observed information for model

Table 5
Performance (RMSE/MAPE) comparison of BGCP, GWNET and BGCP with complete or incomplete training data on METR-LA dataset. Results averaged over 3 independent runs.

Missing ratio		Complete training data			Incomplete training data		
		20%	40%	60%	20%	40%	60%
	BGCP	5.497/8.69	5.559/8.89	5.640/8.99	5.449/8.59	5.447/8.66	5.462/8.65
RM	GWNET	3.673/5.25	3.897/5.57	4.266/6.10	3.667/5.25	3.911/5.59	4.335/6.28
	MDGCN	*3.033/4.30*	*3.207/4.52*	*3.483/4.90*	*3.161/4.51*	*3.294/4.70*	*3.645/5.21*
	BGCP	5.669/8.95	5.825/9.29	5.974/9.62	5.618/8.80	5.800/9.29	6.031/9.69
TCM	GWNET	4.430/6.32	4.752/6.82	5.208/7.50	4.521/6.38	4.927/7.15	5.470/8.18
	MDGCN	*3.828/5.41*	*4.231/5.98*	*4.643/6.59*	*3.921/5.42*	*4.424/6.17*	*5.077/7.31*
	BGCP	5.441/8.53	5.698/9.02	6.928/11.28	5.410/8.49	5.636/8.93	*7.410*/12.87
SCM	GWNET	4.103/5.90	4.527/6.47	5.876/8.78	4.123/5.98	4.589/6.71	7.980/12.84
	MDGCN	*3.379/4.83*	*3.671/5.18*	*5.205/7.36*	*3.304/4.69*	*3.592/5.05*	8.619/*12.81*
	BGCP	6.419/9.80	7.382/11.60	8.834/14.02	6.550/9.98	7.376/11.64	*9.011/15.76*
BM	GWNET	6.669/10.38	7.002/11.07	7.459/12.30	6.765/10.54	7.235/11.39	9.428/16.31
	MDGCN	*6.119/9.01*	*6.414/9.56*	*7.001/10.92*	*6.305/9.55*	*6.557/9.86*	9.355/16.37

training. For BGCP, the entire data set with the same missing distribution is used as input. The tensor rank for the BGCP model is set as 40 for METR-LA data and 45 for INRIX-SEA data through experiments.

Table 5 displays the imputation performance of BGCP, GWNET and MDGCN with complete or incomplete training data. It can be found that when complete training data is provided, MDGCN outperforms BGCP and GWNET by a large margin in all kinds of missing patterns and missing ratios, indicating the effectiveness of our proposed model with sufficient training data. In the absence of complete training data, MDGCN still achieves notably superior performance to the two baseline models in the RM and TCM patterns for all missing ratios. In SCM and BM, MDGCN performs better than GWNET and BGCP with low missing ratios, whereas BGCP shows slightly better performance when the missing ratio reaches 60%. This result shows that our proposed model can still provide competitive imputation performance in most scenarios with only incomplete training data, while tensor factorization methods show advantage in dealing with incomplete training data with complex missing patterns and high missing ratios. This is reasonable as deep learning models are more data-driven while tensor factorization does not require a large amount of high-quality training data. In summary, deep learning models and tensor factorization models are applicable to different cases: deep learning models can perform better when complete training data is provided or when the missing ratio is relatively low, whereas tensor factorization methods may be preferred in complex missing scenarios with high missing ratios.

7. Conclusion

Missing values in traffic data are an inevitable and important problem for intelligent transportation systems. Recent research have employed GNNs for multivariate time series imputation and achieved state-of-the-art performance. Despite these studies, there exist two important limitations: first, existing approaches cannot directly leverage global spatiotemporal information from different nodes at different time; second, these research mostly do not properly handle dynamic spatial dependencies and correlated missing patterns of traffic data. To fill these research gaps, this paper introduces a novel deep learning-based framework called MDGCN to reconstruct missing traffic data. The proposed method comprises several spatiotemporal blocks to capture spatiotemporal information coherently. Each block contains a bidirectional recurrent layer to capture temporal correlations and a diffusion graph convolution layer to capture spatial correlations. Moreover, we introduce an external memory network to store and share global spatiotemporal information across different nodes. In addition, a graph structure estimation technique is proposed to learn dynamic spatial dependencies directly from traffic data. Extensive experiments are conducted to compare our proposed model with several state-of-the-art deep learning models in four types of missing patterns using two public traffic speed datasets. The results show that our proposed model achieves superior performance to existing methods in different missing patterns and provides robust results with a wide range of missing ratios. Further ablation analysis validates the contribution of our proposed external memory network and graph structure estimation technique. In addition, we compare our proposed model with a tensor factorization method under different training data availability. It is found that the our model significantly outperforms the tensor factorization method with high-quality training data whereas the tensor factorization method is more suitable to deal with incomplete training data with high missing ratios.

There are several directions for future works. First, for complex missing scenarios without sufficient training data, our model needs further improvement. It would be meaningful to incorporate transfer learning techniques in our proposed method, so that we can leverage models trained in data-rich environments for scenarios without adequate training data. Second, the spectral clustering method currently used in our model requires the historical traffic data of nodes and might not work well when the missing ratio is high. A more robust clustering method can be introduced to enhance the performance of our model in the case of incomplete data with high missing ratios. A potential method is to use a Bayesian approach, where the geographic proximity is the prior and the traffic pattern similarity is the posterior. When traffic data is limited, we rely more on the geographic proximity; when the traffic data is abundant, we rely more on the traffic pattern similarity. Third, existing imputation researches are usually based on traffic data from either loop sensors or GPS probes. Loop sensors provide traffic data with high sampling rates yet limited spatial coverage, whereas GPS probes cover a wide spatial range with low temporal resolutions. By considering traffic data from GPS probes and loop sensors simultaneously, it is possible to recover traffic data with high spatial and temporal resolutions.

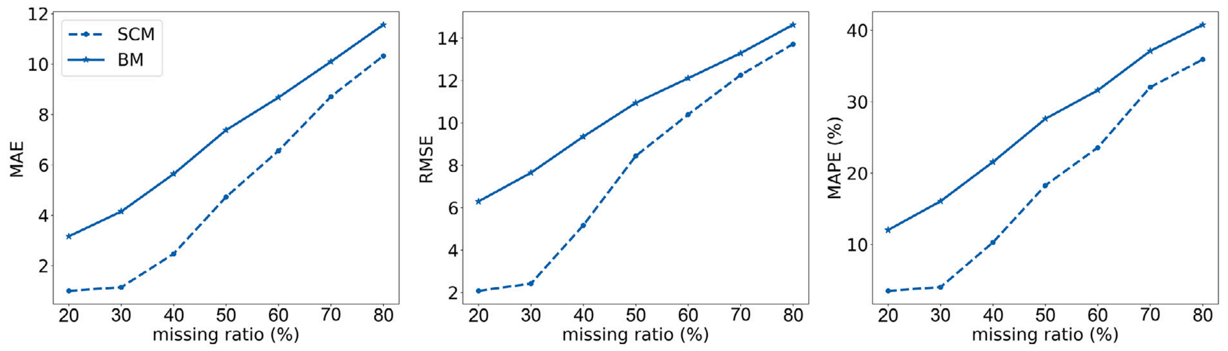

Fig. B.7. The performance of DAE for SCM and BM with INRIX-SEA data.

CRediT authorship contribution statement

Yuebing Liang: Conceptualization, Methodology, Software, Formal analysis, Writing – original draft. **Zhan Zhao:** Conceptualization, Methodology, Writing – review & editing. **Lijun Sun:** Methodology, Writing – review & editing.

Acknowledgments

The authors would like to thank the anonymous reviewers for their constructive and detailed comments.

Appendix A. Methods for missing pattern generation

Suppose the missing ratio is r and the imputation time window is $[1, T]$, we generate the missing patterns using the following methods:

RM: A random integer a within range $[0, 1)$ is generated for each data point. If $a < r$, the data point is masked.

TCM: For each node in the traffic network, randomly choose a time point t within $[1, T]$. If $t \leq T - \lfloor T * r \rfloor$, data points within $[t, t + \lfloor T * r \rfloor]$ are masked. If not, data points within range $[1, \lfloor T * r \rfloor - (T - t))$ and $[t, T]$ are masked.

SCM: For each time slot in the imputation window, randomly choose a node v in the traffic network. If the traffic network contains a set of sensors, the $\lfloor N * r \rfloor$ sensors closest to v are searched and masked. If the traffic network contains a set of road links, we use the Breadth-first Search Algorithm to iterate over the road network from v and the first $\lfloor N * r \rfloor$ road links to be discovered are masked.

BM: BM contains missing data points which are both temporally and spatially correlated. We use Alg. 2 to generate the BM pattern.

Algorithm 2: Generating the BM pattern on a data matrix S

```
Input : missing ratio $r$, imputation window $[1, T]$, the input data matrix $S$
$i = 1$;
while $i \leq T$ do
    Randomly generate an integer $a$ within range $[1, T - i]$;
    Randomly choose a node $v$ from the traffic network;
    Select the $\lfloor N * r \rfloor$ closest nodes $N_v$ to $v$;
    Mask the data points of $N_v$ within time range$[i, i + a]$ in $S$;
    $i = i + a$;
end
```

Appendix B. The imputation performance of DAE for SCM and BM with INRIX-SEA data

The imputation performance of DAE in SCM and BM patterns for INRIX-SEA data is displayed in Fig. B.7.

Appendix C. The t-test result of the contribution of the external memory networks

To verify the contribution of our proposed external memory network, we repeat the experiments with and without the external memory network six times and use a t-test to evaluate whether the performance improvement provided by the external memory network is significant. The result is displayed in Table C.6. It can be found that the external memory network contributes to the model performance significantly for all missing patterns and missing ratios.

Table C.6
t-test result of performance improvement brought by the external memory network on METR-LA data.

Missing patterns			Missing ratios						
			20%	30%	40%	50%	60%	70%	80%
RM	RMSE	Improvement %	0.92	0.81	0.85	0.87	0.87	1.03	1.11
		t-value	−2.279	−2.024	−2.091	−2.023	−1.920	−2.100	−1.924
		p-value	0.023**	0.035*	0.032**	0.035**	0.042**	0.039**	0.048**
	MAPE	Improvement %	1.23	1.13	1.18	1.16	1.14	1.20	1.26
		t-value	−2.295	−2.035	−2.119	−2.024	−1.948	−1.924	−1.753
		p-value	0.022**	0.035**	0.030**	0.035**	0.040**	0.042**	0.055*
TCM	RMSE	Improvement %	2.92	2.89	3.14	3.34	3.23	3.11	3.02
		t-value	−2.620	−2.244	−2.142	−2.120	−2.103	−2.125	−2.194
		p-value	0.013**	0.024**	0.029**	0.030**	0.031**	0.030**	0.026**
	MAPE	Improvement %	3.22	3.59	4.00	4.32	4.12	3.96	3.73
		t-value	−2.693	−2.429	−2.271	−2.238	−2.187	−2.132	−2.016
		p-value	0.011**	0.018**	0.023**	0.025**	0.027**	0.029**	0.036**
SCM	RMSE	Improvement %	0.57	0.56	0.55	0.99	4.97	4.69	3.69
		t-value	−3.370	−3.824	−3.641	−3.276	−4.436	−3.938	−3.068
		p-value	0.004***	0.002***	0.002***	0.004***	0.001***	0.001***	0.006***
	MAPE	Improvement %	1.57	1.55	1.55	2.46	8.65	8.38	7.00
		t-value	−5.714	−5.596	−5.212	−4.028	−3.902	−3.510	−2.826
		p-value	0.000***	0.000***	0.000***	0.001***	0.001***	0.003***	0.009***
BM	RMSE	Improvement %	2.16	2.74	2.70	2.52	3.17	2.98	2.70
		t-value	−5.194	−7.767	−7.579	−5.243	−6.172	−5.868	−7.143
		p-value	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***
	MAPE	Improvement %	5.11	5.30	5.63	5.76	6.71	6.27	5.85
		t-value	−6.850	−8.151	−7.472	−7.267	−7.290	−5.919	−5.894
		p-value	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***	0.000***

*Represent the confidence level at 90%.
**Represent the confidence level at 95%.
***Represent the confidence level at 99%.

References

Asadi, R., Regan, A., 2019. A convolution recurrent autoencoder for spatio-temporal missing data imputation. arXiv preprint arXiv:1904.12413.

Asif, M.T., Mitrovic, N., Garg, L., Dauwels, J., Jaillet, P., 2013. Low-dimensional models for missing data imputation in road networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, pp. 3527–3531.

Aydilek, I.B., Arslan, A., 2013. A hybrid method for imputation of missing values using optimized fuzzy c-means with support vector regression and a genetic algorithm. Inform. Sci. 233, 25–35.

Ba, J.L., Kiros, J.R., Hinton, G.E., 2016. Layer normalization. arXiv preprint arXiv:1607.06450.

Bae, B., Kim, H., Lim, H., Liu, Y., Han, L.D., Freeze, P.B., 2018. Missing data imputation for traffic flow speed using spatio-temporal cokriging. Transp. Res. C 88, 124–139.

Berglund, M., Raiko, T., Honkala, M., Kärkkäinen, L., Vetek, A., Karhunen, J.T., 2015. Bidirectional recurrent neural networks as generative models. Adv. Neural Inf. Process. Syst. 28, 856–864.

Cao, W., Wang, D., Li, J., Zhou, H., Li, L., Li, Y., 2018. Brits: Bidirectional recurrent imputation for time series. arXiv preprint arXiv:1805.10572.

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

Chen, X., Sun, L., 2021. BayesIan temporal factorization for multidimensional time series prediction. IEEE Trans. Pattern Anal. Mach. Intell..

Cini, A., Marisca, I., Alippi, C., 2021. Multivariate time series imputation by graph neural networks. arXiv preprint arXiv:2108.00298.

Cui, Z., Lin, L., Pu, Z., Wang, Y., 2020. Graph Markov network for traffic forecasting with missing data. Transp. Res. C 117, 102671.

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

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

Jagadish, H.V., Gehrke, J., Labrinidis, A., Papakonstantinou, Y., Patel, J.M., Ramakrishnan, R., Shahabi, C., 2014. Big data and its technical challenges. Commun. ACM 57 (7), 86–94.

Kuppannagari, S.R., Fu, Y., Chueng, C.M., Prasanna, V.K., 2021. Spatio-temporal missing data imputation for smart power grids. In: Proceedings of the Twelfth ACM International Conference on Future Energy Systems. pp. 458–465.

Laña, I., Olabarrieta, I.I., Vélez, M., Del Ser, J., 2018. On the imputation of missing data for road traffic forecasting: New insights and novel techniques. Transp. Res. C 90, 18–33.

Li, L., Du, B., Wang, Y., Qin, L., Tan, H., 2020a. Estimation of missing values in heterogeneous traffic data: Application of multimodal deep learning model. Knowl.-Based Syst. 194, 105592.

Li, H., Li, M., Lin, X., He, F., Wang, Y., 2020b. A spatiotemporal approach for traffic data imputation with complicated missing patterns. Transp. Res. C 119, 102730.

Li, Y., Yu, R., Shahabi, C., Liu, Y., 2017. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. arXiv preprint arXiv:1707.01926.

Li, L., Zhang, J., Wang, Y., Ran, B., 2018. Missing value imputation for traffic-related time series data based on a multi-view learning method. IEEE Trans. Intell. Transp. Syst. 20 (8), 2933–2943.

Liang, Y., Huang, G., Zhao, Z., 2022a. Bike sharing demand prediction based on knowledge sharing across modes: A graph-based deep learning approach. arXiv preprint arXiv:2203.10961.

Liang, Y., Huang, G., Zhao, Z., 2022b. Joint demand prediction for multimodal systems: A multi-task multi-relational spatiotemporal graph neural network approach. Transp. Res. C 140, 103731.

Liang, Y., Zhao, Z., 2021. NetTraj: A network-based vehicle trajectory prediction model with directional representation and spatiotemporal attention mechanisms. IEEE Trans. Intell. Transp. Syst..

Lipton, Z.C., Kale, D.C., Wetzel, R., et al., 2016. Modeling missing data in clinical time series with rnns. Mach. Learn. Healthc. 56.

Little, R.J., Rubin, D.B., 2019. Statistical Analysis with Missing Data. Vol. 793. John Wiley & Sons.

Liu, Y., Yu, R., Zheng, S., Zhan, E., Yue, Y., 2019. Naomi: Non-autoregressive multiresolution sequence imputation. arXiv preprint arXiv:1901.10946.

Luo, Y., Cai, X., Zhang, Y., Xu, J., Yuan, X., 2018. Multivariate time series imputation with generative adversarial networks. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. pp. 1603–1614.

Ni, D., Leonard, J.D., 2005. Markov chain Monte Carlo multiple imputation using Bayesian networks for incomplete intelligent transportation systems data. Transp. Res. Rec. 1935 (1), 57–67.

Park, C., Lee, C., Bahng, H., Tae, Y., Jin, S., Kim, K., Ko, S., Choo, J., 2020. ST-GRAT: A novel spatio-temporal graph attention networks for accurately forecasting dynamically changing road speed. In: Proceedings of the 29th ACM International Conference on Information & Knowledge Management. pp. 1215–1224.

Qu, L., Li, L., Zhang, Y., Hu, J., 2009. PPCA-based missing data imputation for traffic flow volume: A systematical approach. IEEE Trans. Intell. Transp. Syst. 10 (3), 512–522.

Ran, B., Tan, H., Wu, Y., Jin, P.J., 2016. Tensor based missing traffic data completion with spatial–temporal correlation. Physica A 446, 54–63.

Smith, B.L., Scherer, W.T., Conklin, J.H., 2003. Exploring imputation techniques for missing data in transportation management systems. Transp. Res. Rec. 1836 (1), 132–142.

Spinelli, I., Scardapane, S., Uncini, A., 2020. Missing data imputation with adversarially-trained graph convolutional networks. Neural Netw. 129, 249–260.

Wu, Z., Pan, S., Long, G., Jiang, J., Zhang, C., 2019. Graph wavenet for deep spatial-temporal graph modeling. arXiv preprint arXiv:1906.00121.

Wu, Y., Zhuang, D., Labbe, A., Sun, L., 2020. Inductive graph neural networks for spatiotemporal kriging. arXiv preprint arXiv:2006.07527.

Yao, H., Liu, Y., Wei, Y., Tang, X., Li, Z., 2019. Learning from multiple cities: A meta-learning approach for spatial-temporal prediction. In: The World Wide Web Conference. pp. 2181–2191.

Ye, J., Sun, L., Du, B., Fu, Y., Xiong, H., 2020. Coupled layer-wise graph convolution for transportation demand prediction. arXiv preprint arXiv:2012.08080.

Ye, Y., Zhang, S., Yu, J.J.Q., 2021. Spatial-temporal traffic data imputation via graph attention convolutional network. In: Farkaš, I., Masulli, P., Otte, S., Wermter, S. (Eds.), Artificial Neural Networks and Machine Learning – ICANN 2021. Springer International Publishing, Cham, pp. 241–252.

Yoon, J., Jordon, J., Schaar, M., 2018. Gain: Missing data imputation using generative adversarial nets. In: International Conference on Machine Learning. PMLR, pp. 5689–5698.

Yu, L., Du, B., Hu, X., Sun, L., Han, L., Lv, W., 2021. Deep spatio-temporal graph convolutional network for traffic accident prediction. Neurocomputing 423, 135–147.

Yu, B., Yin, H., Zhu, Z., 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. arXiv preprint arXiv:1709.04875.

Zhang, Z., Lin, X., Li, M., Wang, Y., 2021a. A customized deep learning approach to integrate network-scale online traffic data imputation and prediction. Transp. Res. C 132, 103372.

Zhang, Y., Liu, Y., 2009. Missing traffic flow data prediction using least squares support vector machines in urban arterial streets. In: 2009 IEEE Symposium on Computational Intelligence and Data Mining. IEEE, pp. 76–83.

Zhang, W., Zhang, P., Yu, Y., Li, X., Biancardo, S.A., Zhang, J., 2021b. Missing data repairs for traffic flow with self-attention generative adversarial imputation net. IEEE Trans. Intell. Transp. Syst..

Zheng, C., Fan, X., Wang, C., Qi, J., 2020. Gman: A graph multi-attention network for traffic prediction. In: Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 34. (01), pp. 1234–1241.

Zhong, M., Lingras, P., Sharma, S., 2004. Estimation of missing traffic counts using factor, genetic, neural, and regression techniques. Transp. Res. C 12 (2), 139–166.