Transportation Research Part C 143 (2022) 103820

Contents lists available at ScienceDirect

Transportation Research Part C

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

A novel reinforced dynamic graph convolutional network model with data imputation for network-wide traffic flow prediction

Yong Chen [a], Xiqun (Michael) Chen [a,b,c,*]

[a] College of Civil Engineering and Architecture, Zhejiang University, Hangzhou 310058, China
[b] Zhejiang University/the University of Illinois Urbana-Champaign Institute (ZJU-UIUC), Zhejiang University, Haining 314400, China
[c] Alibaba-Zhejiang University Joint Research Institute of Frontier Technologies, Hangzhou 310007, China

ARTICLE INFO

Keywords:
Network-wide traffic flow prediction
Graph convolutional networks
Data imputation
Deep reinforcement learning

ABSTRACT

Traffic data missing issues due to unpredictable equipment failure, extreme weather, and other reasons have brought great challenges to traffic flow prediction modeling. In this paper, a novel reinforced dynamic graph convolutional network model is proposed to simultaneously conduct data imputation and network-wide traffic flow prediction. First, a multi-graph convolutional fusion network is proposed for data imputation by using the graph convolutional network to analyze the propagation law of traffic states between traffic flow detection stations in both time and space dimensions. Second, to enhance the robustness of network-wide traffic flow prediction, a dynamic graph learning method based on deep reinforcement learning is proposed to adaptively generate the graph adjacency matrix to represent the dynamic spatiotemporal dependencies between the stations. Finally, experimental results on two real-world traffic datasets show that the proposed method outperforms other baseline methods and can effectively extract the data missing features and spatiotemporal dependence features between the stations. The visualization results of the graph adjacency matrix indicate that the proposed method can effectively identify the influential traffic stations in the process of traffic flow prediction, and the extracted dependencies between the stations are interpretable. The proposed model has strong generalization in tackling network-wide traffic flow prediction tasks with different data missing rates and missing patterns, and can be extended to assist decision-makers in enhancing traffic management and mitigating traffic congestion.

1. Introduction

With the continuous increase of traffic flow data sources and the development of artificial intelligence technology, constructing an accurate and data-driven network-wide traffic flow prediction model becomes feasible. The core task of network-wide traffic flow prediction is to mine and analyze collected traffic data, extract temporal and spatial correlation features between various traffic flow detection stations or sensors, and predict traffic states (e.g., traffic speed, traffic volume, and time occupancy) of the road network in the future (Cui et al., 2019). The accurate prediction results can further assist decision-makers in mitigating traffic congestion and enhancing traffic management and control.

* Corresponding author at: B828 Anzhong Building, College of Civil Engineering and Architecture, Zhejiang University, 866 Yuhangtang Rd, Hangzhou 310058, China.
E-mail address: chenxiqun@zju.edu.cn (X.(M. Chen).

https://doi.org/10.1016/j.trc.2022.103820
Received 11 August 2021; Received in revised form 15 February 2022; Accepted 21 July 2022
Available online 2 August 2022

In recent years, the deep learning framework based on graph convolution network (GCN) (Kipf and Welling, 2017) has been successfully applied to traffic flow prediction modeling (Jiang and Luo, 2021). Due to the advantages of GCN in dealing with non-Euclidean or irregular data (e.g., transportation network data, and social network data), many existing studies attempt to represent the entire transportation network as a graph and then use GCN to process the graph directly. Various empirical studies (e.g., Guo et al., 2019; Lv et al., 2020) show that the GCN-based method can extract more realistic transportation network features and effectively improve network-wide traffic flow prediction accuracy compared with grid-based and image-based methods. Therefore, this study focuses on the limitations of the existing prediction model based on GCN and attempts to design a more robust and accurate traffic flow prediction model. Specifically, the limitations include: (a) Few GCN-based models are capable of tackling the issue of data missing; and (b) few studies consider the dynamic features of transportation networks in the process of modeling.

Affected by the external environment (e.g., extreme weather, and signal shielding), some traffic sensors, detectors, and other equipment in the transportation network are prone to failure, making the collected traffic data have missing values. Using incomplete data for model training will degrade the prediction performance of the model. To solve the issue of data missing, existing studies generally use the methods based on interpolation (Yang et al., 2018), tensor decomposition (Zhang et al., 2019), and deep learning (Duan et al., 2016) to predict or estimate missing data before traffic flow prediction. However, the existing data imputation methods rarely consider the transportation network topology, which leads to a lack in the reliability of imputation results. Meanwhile, the data imputation and traffic flow prediction in these studies are modeled separately, which increases the computational cost of the whole model and leads to a biased prediction result because the prediction model fails to learn the missing patterns of data effectively.

In the aspect of modeling dynamic features of transportation networks, existing GCN-based prediction models generally rely on the geographical distance (Zhu et al., 2019), correlation of traffic series (Liu et al., 2020), and functional similarity of traffic flow detection stations (Geng et al., 2019) to predefine a static graph adjacency matrix. However, the spatiotemporal dependencies between the stations can usually be affected by traffic accidents, holiday activities, and the station's surrounding environment, and constantly change over time. For example, the dependence between the stations in residential and business center areas is strong on weekdays and weak at weekends. In contrast, the dependence between the stations in residential areas and scenic spots is strong at weekends. Therefore, it is difficult to capture the dynamic and complex features of the transportation network by the existing generation methods of the graph adjacency matrix, which limits the performance of the network-wide prediction model.

Therefore, a reinforced dynamic graph convolutional network model with data imputation (RDGCNI) for network-wide traffic flow prediction is proposed in this study to solve the issues mentioned above. In particular, the main contributions are summarized as follows:

(a) A novel GCN-based deep learning framework is proposed to simultaneously conduct data imputation and network-wide traffic flow prediction. The proposed framework can extract data missing features and spatiotemporal dependence features between the stations when the model is trained with incomplete traffic data. Meanwhile, it can also effectively alleviate error accumulation from data imputation to network-wide traffic flow prediction and improve the accuracy of the prediction results.

(b) A new multi-graph convolutional fusion network is proposed to impute traffic data missing values. The network uses GCN to analyze the propagation law of traffic states between traffic flow detection stations in both time and space dimensions, which improves the accuracy of data imputation.

(c) A new dynamic graph learning network based on deep reinforcement learning (DRL) is proposed to generate a graph adjacency matrix adaptively. The network incorporates the deep deterministic policy gradient (DDPG) algorithm (Lillicrap et al., 2016) to perceive the changes in the traffic environment and generates a graph adjacency matrix representing the dynamic spatiotemporal dependencies between the stations, which enhances the robustness of the network-wide prediction model.

(d) The proposed model is verified on two real-world traffic flow datasets. The experimental results show that RDGCNI is not sensitive to missing values in training data and can effectively extract the spatiotemporal dependence features between the stations, which significantly outperforms baseline methods.

The remainder of this paper is organized as follows. Section 2 reviews previous research related to traffic flow prediction and missing data imputation. Section 3 elaborates the methodology involved in this study. Section 4 introduces experimental settings and discusses the results. Finally, Section 5 concludes the study and provides an outlook on future research.

2. Related literature

2.1. Traffic flow prediction

Accurate traffic flow prediction plays a vital role in intelligent transportation systems. With the development of computer technology, various methods have been proposed to improve the accuracy and robustness of traffic flow prediction. The existing traffic flow prediction methods can be divided into statistical methods, machine learning methods, and deep learning methods. Early statistical methods, such as exponential smoothing (Williams et al., 1998), and auto-regressive integrated moving average (Lee and Fambro, 1999), infer and predict future traffic states using fixed theoretical assumptions. Although this method is computationally simple, it is difficult to model nonlinear features of traffic data. Later, machine learning methods such as support vector machine (Hong, 2011), Bayesian network (Sun et al., 2006), and extreme learning machine (Xing et al., 2017) are proposed to map input data into a high-dimensional feature space to extract complex features in traffic data. However, the performance of these machine learning methods is usually dependent on feature engineering, which requires much experience from experts in related fields (Guo et al., 2019).

In the past few years, deep learning methods have been widely used to solve traffic flow prediction tasks, automatically extracting features layer-by-layer by stacking multi-layer neural network units. For example, Huang et al. (2014) proposed a deep learning model

combined with a deep belief network (DBN) and multi-task regression layer for traffic flow prediction, in which DBN extracted data features in an unsupervised way. Jo et al. (2018) represented the traffic state at each time point with an image, and proposed an end-to-end convolutional neural network (CNN) framework to extract spatiotemporal features and predict the traffic speed of all stations. In addition, recurrent neural networks and their variants, such as gated recurrent unit (GRU) (Du et al., 2020), long short-term memory networks (LSTM) (Yang et al., 2019), and bidirectional LSTM (Wang et al., 2019), are widely used to extract dynamic temporal features of traffic data by using ingenious gate structures to control the transmission of feature information. However, these deep learning methods are suitable for feature extraction of Euclidean data. The transportation network is similar to the social network and gene network, which belong to an irregular and non-Euclidean structure. Therefore, the performance of the above deep learning methods will be weakened when modeling transportation network data.

Recently, the deep learning method based on GCN has been successfully used to extract the spatiotemporal features of transportation network data. The GCN-based method characterizes the transportation network as a graph, and the graph's nodes represent traffic flow detection stations or road segments, and the graph's edges represent the dependencies between the stations. Representing the transportation network data in the form of a graph can effectively retain the original transportation network structure and improve traffic flow prediction accuracy. For instance, Yu et al. (2018) proposed a traffic flow prediction model based on GCN, which constructed a graph adjacency matrix based on the geographical distance between the stations. The experimental results showed that the proposed model improved the prediction accuracy and reduced the training time of the model. Lv et al. (2020) proposed a GCN-based traffic flow prediction model, which encoded the temporal, spatial, and semantic correlations between roads into multiple graphs, and extracted the spatial dependence features of the traffic flow by using GCN. Compared with CNN-based and LSTM-based methods, these GCN-based prediction methods extracted temporal and spatial dependence features of traffic data more reliably. However, they need to rely on prior knowledge to construct a fixed graph adjacency matrix representing the transportation network, which is difficult to adapt to the complex and stochastic traffic environment. Modeling the dynamic graph adjacency matrix is still in its infancy. Therefore, to fill this gap, a new dynamic graph learning method based on DRL is proposed in this study to improve the spatiotemporal feature extraction ability of GCN.

2.2. Missing data imputation

In collecting traffic data, problems such as communication failure and sensor failure will lead to data missing at some traffic flow detection stations. To alleviate the impact of incomplete data on the prediction model training, the existing studies generally use the methods, such as zero value, the mean value of historical series, and constructing regression models to impute missing data and then using the imputed traffic data to train the prediction model. The imputation method can be divided into the asynchronous data imputation method and synchronous data imputation method, according to whether data imputation and traffic flow prediction are executed separately.

In the asynchronous data imputation method, statistical methods, such as the last observed value (Amiri and Jensen, 2016) and linear interpolation (Junninen et al., 2004), use an average value or a weighted average value of adjacent non-missing data points to infer missing data. In addition, machine learning methods are also used to construct regression models and discriminant models for data imputation. The traditional data imputation methods based on machine learning include matrix factorization (Koren et al., 2009), multivariate imputation by chain equations (MICE) (Buuren and Groothuis-Oudshoorn, 2011), SoftImpute (Mazumder et al., 2010), etc. Recently, various imputation methods based on deep learning have also been proposed to reduce imputation errors. For example, El-Fiqi et al. (2019) proposed a deep learning framework based on autoencoder with a gate mechanism for data imputation. Du et al. (2019) proposed a data imputation method based on DBN, in which DBN was used to extract the spatiotemporal correlation features of input data. Moreover, the generative adversarial network (GAN) is also used by researchers for data imputation modeling. For instance, Yoon et al. (2018) proposed a GAN-based data imputation method, which used a hint matrix to impute data in an unsupervised way. Chen et al. (2019) proposed a data imputation method based on parallel data and GAN, which improved GAN by using real traffic data as latent variables of GAN. Luo et al. (2019) proposed an end-to-end generative model to improve the accuracy of data imputation, which used GRU with data imputation unit as the generator and discriminator of GAN to improve GAN's temporal feature extraction ability.

Although the above asynchronous data imputation methods can alleviate the impact of data missing on the prediction model, its original intention is to solve the data imputation task without considering the model's prediction performance. Recently, several empirical studies (e.g., Che et al., 2018; Cui et al., 2020a) have shown that if data imputation and model prediction are executed separately, the imputation error caused by the data imputation method will make the prediction model biased to estimate model parameters in the training process. Meanwhile, the prediction model cannot thoroughly learn the missing patterns of the data in the training process, and then the trained model can only obtain a suboptimal prediction result. Therefore, several studies have been proposed to integrate the two steps of data imputation and model prediction to enable the prediction model to learn the missing patterns of input data. For example, Che et al. (2018) proposed a GRU-based prediction model, which considered the missing pattern of data (i.e., masking information and time interval) in the process of network computing, enabling the model to simultaneously achieve data imputation and prediction. Tian et al. (2018) proposed a data imputation method based on LSTM (LSTM-M) for traffic flow prediction, which combined a multi-scale temporal smoothing strategy to infer missing data. Similarly, Cui et al. (2020a) proposed a deep learning framework based on LSTM and bidirectional LSTM for data imputation and traffic flow prediction. The proposed framework embedded a data imputation unit, which could be trained and updated with LSTM. Boquet et al. (2020) proposed a prediction model with data imputation and outlier detection based on variational autoencoder, which compressed traffic data samples in a latent space by using an encoder, and then used a decoder to reconstruct data samples to achieve data imputation.

In the transportation network, the propagation of traffic states between traffic flow detection stations has a specific pattern and law in time and space dimensions. However, few studies have incorporated them into the modeling of data imputation. This study proposes a GCN-based deep learning framework to simultaneously conduct data imputation and traffic flow prediction. In particular, a multi-graph convolutional fusion network is proposed, in which the transportation network topology is taken into account during data imputation to improve the accuracy of data imputation.

3. Methodology

In this section, a novel reinforced dynamic graph convolution network model with data imputation (RDGCNI) is proposed for network-wide traffic flow prediction. The architecture of RDGCNI is shown in Fig. 1. On the left hand, a multi-graph convolutional fusion network is proposed for data imputation before network-wide traffic flow prediction, which can make full use of the propagation law of traffic states between traffic flow detection stations to improve the accuracy of data imputation. Then, the originally non-missing data and the imputed data are combined for network-wide traffic flow prediction. On the right hand, a reinforced dynamic graph learning network is proposed to enable GCN to capture dynamic transportation network features. As shown at the bottom of Fig. 1, the network can adapt to the change of traffic environment and dynamically generate a graph adjacency matrix to give feedback to the prediction model. Before introducing the details of RDGCNI, the network-wide traffic flow prediction problem solved in this study is defined as follows.

In this paper, the transportation network is represented as graph $\mathcal{G} = (\mathbb{V}, \mathbb{E}, \mathbf{A})$, where $\mathbb{V} = \{V_1, V_2, \ldots, V_N\}$ denotes a set of nodes of the graph, corresponding to N stations of the transportation network. $\mathbb{E}$ denotes a set of edges, which is used to represent connection relationships between the stations. $\mathbf{A}$ denotes the graph's adjacency matrix, which represents the strength of the dependency relationship between the stations. Meanwhile, the historical traffic data series including N stations can be represented as $\mathbf{X} = \{\mathbf{X}_{t-T}, \mathbf{X}_{t-T+1}, \ldots, \mathbf{X}_t\} \in \mathbb{R}^{T\times N}$, where T denotes the length of the series, vector $\mathbf{X}_t$ denotes the traffic information values of N stations at time t. Further, the traffic information value of the n-th station at time t can be denoted as x_t^n. In reality, traffic data may be missing at some stations due to the failure or destruction of detectors. The missing state of historical traffic data for all stations can be represented by masking matrix $\mathbf{M} = \{\mathbf{M}_{t-T}, \mathbf{M}_{t-T+1}, \ldots, \mathbf{M}_t\} \in \{0, 1\}^{T\times N}$, where vector $\mathbf{M}_t$ denotes the missing state of traffic data of N stations at time t. Further, the missing state of the n-th station at time t can be denoted as m_t^n, which is defined by Eq. (1):

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

According to the above definition, the traffic flow prediction problem can be formulated as learning function $F(\cdot)$ to predict the traffic state at the next time point based on transportation network graph $\mathcal{G}$, historical traffic data series $\mathbf{X}$, and masking matrix $\mathbf{M}$, as shown in Eq. (2):

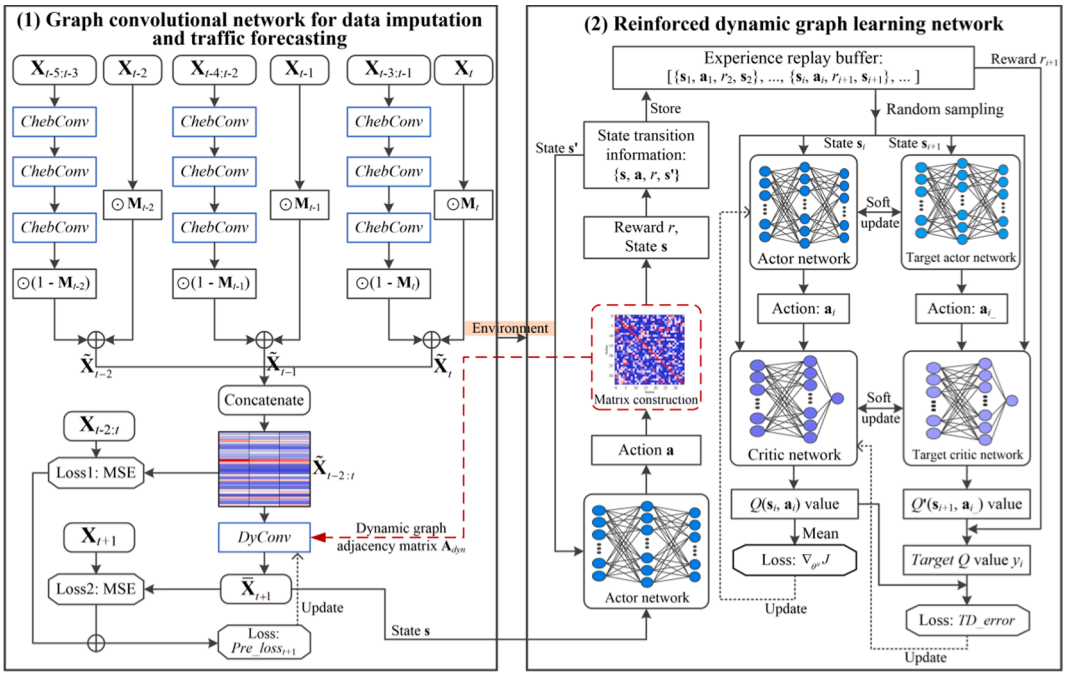


Fig. 1. Architecture of reinforced dynamic graph convolution network model with data imputation.

$$F(\mathcal{G},\ \mathbf{X},\ \mathbf{M})\ =\ \mathbf{X}_{t+1} \tag{2}$$

3.1. Network-wide traffic flow prediction with missing data using multi-graph convolutional fusion network

To alleviate the impact of incomplete data on the performance of traffic flow prediction models, this study proposes a GCN-based network-wide traffic flow prediction model with data imputation (GCNI) to simultaneously conduct data imputation and network-wide traffic flow prediction. The proposed GCNI utilizes GCN to analyze the propagation law of traffic states between stations in both time and space dimensions. Then the missing data of the target station can be inferred from the past traffic data for all stations.

As shown in the upper left side of Fig. 1, a multi-graph convolutional fusion network is constructed for data imputation. It is assumed that the traffic data of all stations at the past three time points (i.e., $\mathbf{X}_t$, $\mathbf{X}_{t-1}$, and $\mathbf{X}_{t-2}$) are used as the input of GCNI to predict the traffic state of all stations at the next time point. The traffic flow of a station is generally influenced by the past traffic flow of its neighboring or related stations. In the input data, the missing data at a time point is inferred from the traffic data of all stations at three time points before this time point. A schematic diagram of data inference is shown in Fig. 2, and the calculation method is given by:

$$\widehat{\mathbf{X}}_t = \sigma(ChebConv(\mathbf{X}_{t-3:t-1},\ \mathbf{A}_{dis})) \tag{3}$$

where σ denotes the activation function, and the rectified linear unit (relu) function (Nair and Hinton, 2010) is used in this study. $\widehat{\mathbf{X}}_t$ denotes an inference vector, which corresponds to the data inference results of all stations at time point t (including stations with no missing data). $\mathbf{X}_{t-3:t-1}$ denotes the traffic flow data of all stations at time $t-3$ through $t-1$. $\mathbf{A}_{dis}$ denotes the graph adjacency matrix. $ChebConv(\cdot)$ denotes a Chebyshev approximation based K-hop localized spectral graph convolution operation (Defferrard et al., 2016), which is defined as:

$$ChebConv(\cdot) = \sum_{k=0}^{K} \phi_k(\widetilde{\mathbf{L}})\mathbf{H}_l\alpha_k \tag{4}$$

where α_k denotes a learnable vector of parameters. K denotes the maximum order of the Chebyshev approximation. $\mathbf{H}_l$ denotes the feature input of the l-th convolution layer. $\phi_k(\widetilde{\mathbf{L}}) \in \mathbb{R}^{N\times N}$ denotes the k-th order Chebyshev polynomial with scaled graph Laplacian matrix $\widetilde{\mathbf{L}}$ as input (Defferrard et al., 2016), which is calculated as follows:

$$\begin{cases} \phi_k(\widetilde{\mathbf{L}}) &= 2\widetilde{\mathbf{L}}\phi_{k-1}(\widetilde{\mathbf{L}}) - \phi_{k-2}(\widetilde{\mathbf{L}}) \\ & \phi_0 = \mathbf{I}_N,\ \phi_1 = \widetilde{\mathbf{L}} \end{cases} \tag{5}$$

$$\begin{cases} \widetilde{\mathbf{L}} &= 2\mathbf{L}/\lambda_{max} - \mathbf{I}_N \\ \mathbf{L} &= \mathbf{I}_N - \mathbf{D}^{-1/2}\mathbf{A}_{dis}\mathbf{D}^{-1/2} \end{cases} \tag{6}$$

where $\mathbf{I}_N$ and $\mathbf{D}$ denote the identity matrix and degree matrix, respectively. λ_{max} denotes the maximum eigenvalue of normalized graph Laplacian matrix $\mathbf{L}$. Considering that the propagation of traffic states is related to the spatial distance between the stations. The traffic state of each station will first propagate to the upstream and downstream stations and gradually propagate to farther stations with the increase of time lag. Therefore, the reciprocal of the distance between two stations is used in this study to calculate graph adjacency matrix $\mathbf{A}_{dis}$ to characterize the reachability of traffic states between the stations.

Then, missing values of the input data at each time point can be updated by inference vector $\widehat{\mathbf{X}}_t$ as follows:

$$\widetilde{\mathbf{X}}_t = \mathbf{X}_t \odot \mathbf{M}_t + \widehat{\mathbf{X}}_t \odot (1 - \mathbf{M}_t) \tag{7}$$

where $\widetilde{\mathbf{X}}_t$ denotes the input data after data imputation. $\odot$ denotes the Hadamard product (Davis, 1962) between two matrices.

Further, the input data of each time point after data imputation is concatenated into a matrix. The matrix is used as the feature input of the prediction model for spatiotemporal feature extraction and network-wide traffic flow prediction. Specifically, the calculation

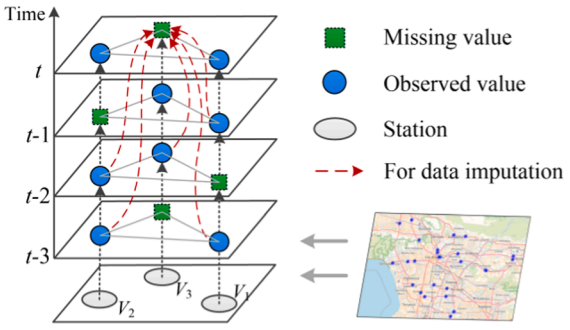


Fig. 2. Schematic diagram of data inference.

method of the prediction model is given by:

$$\overline{\mathbf{X}}_{t+1} = DyConv(\widetilde{\mathbf{X}}_{t-2:t},\ \mathbf{A}_{dyn}) = \sigma(\mathbf{A}_{dyn}\widetilde{\mathbf{X}}_{t-2:t}\mathbf{W}_g) \tag{8}$$

where $\overline{\mathbf{X}}_{t+1}$ denotes the prediction vector of the traffic state of all stations at time $t+1$. $\mathbf{W}_g$ denotes the learnable weight matrix of the graph convolution network. $DyConv(\cdot)$ denotes a dynamic graph convolution operation, which uses dynamic graph adjacency matrix $\mathbf{A}_{dyn}$ to perceive and represent the changes of the traffic environment adaptively. The adjacency matrix $\mathbf{A}_{dyn}$ is calculated by the dynamic graph learning method based on DRL.

Moreover, this study aims to make the predicted traffic state values as close as possible to the observed values. Therefore, this study uses the mean square error (MSE) metric to measure the model's prediction error. Meanwhile, we also hope that the results of data imputation can be as similar to the original observations as possible. Therefore, the data imputation error is taken into account in the loss function of the prediction model to enhance the accuracy of data imputation while ensuring the accuracy of the prediction model. The final error loss function is defined by.

$$Pre\_Loss_{t+1} = \frac{\sum_{n=1}^{N} m_{t+1}^{n}\left(\overline{x}_{t+1}^{n} - x_{t+1}^{n}\right)^2}{\sum_{n=1}^{N} m_{t+1}^{n}} + \beta \cdot \frac{\sum_{j=0}^{2}\sum_{n=1}^{N}\left(1 - m_{t-j}^{n}\right)\left(\widetilde{x}_{t-j}^{n} - x_{t-j}^{n}\right)^2}{\sum_{j=0}^{2}\sum_{n=1}^{N}\left(1 - m_{t-j}^{n}\right)} \tag{9}$$

where β denotes the penalty term. j denotes the index value of the historical time window. Finally, the obtained prediction error is fed back to update the network parameters of GCNI.

3.2. Reinforced dynamic graph learning network

Although the GCN-based network-wide traffic flow prediction model has made great progress in the feature extraction ability of transportation network data, it relies on the design and definition of the graph adjacency matrix. Previous studies generally use the geographical distance between the stations or the similarity of historical traffic series to predefine a fixed or static graph adjacency matrix. However, the dependencies between stations are related to many factors, such as weather, events, festivals, and points of interest. In particular, dependencies between stations may change over time in the transportation network. It is difficult to model these dynamic and complex dependencies using a predefined adjacency matrix. In recent years, as a sub-family of deep learning, DRL has been successfully applied in various fields, such as smart driving (Kuderer et al., 2015), robotics (Gu et al., 2017), and recommendation systems (Wei et al., 2020). DRL integrates the feature learning ability of deep learning and the decision-making ability of reinforcement learning, and can adaptively perceive complex and changeable environmental states and make corresponding actions for feedback. Therefore, a new dynamic graph learning network based on DRL is proposed in this study to adaptively learn and extract the static and dynamic spatiotemporal dependencies between the stations in the road network. The extracted features include the physical relationships of the road network and the complex nonlinear spatiotemporal dependencies caused by multiple external environmental factors (e.g., weather conditions, and traffic accidents).

In this study, the DRL algorithm based on the Actor-Critic framework (Konda and Tsitsiklis, 1999), DDPG, is used as the DRL's agent to dynamically depict the spatiotemporal dependencies between the stations in different traffic environments. Compared to Actor-Critic, DDPG uses the experience replay buffer as a memory storage device to store the state transition information (i.e., action, state, and reward) of each moment. The network is updated by randomly sampling state transition information samples from the experience replay buffer. In addition, DDPG constructs target networks for the actor network and critic network, respectively, to alleviate the problem of overestimating the action-value function (Lillicrap et al., 2016). The schematic diagram of the agent interacting with the traffic environment is shown in Fig. 1. Before introducing the interaction process between the agent and traffic environment, it is necessary to define several components of DRL according to application problems.

Definition 1. ***Action.*** *To make the agent can adaptively perceive traffic states and depict complex and dynamic spatiotemporal dependencies between stations, graph adjacency matrix* $\mathbf{A}_{dyn} \in \mathbb{R}^{N\times N}$ *is used as the agent's action. Note that* $\mathbf{A}_{dyn}$ *is an asymmetric matrix because the impact of intermediate stations on upstream and downstream stations is different. In matrix* $\mathbf{A}_{dyn}$*, the value range of elements is between zero and one, representing the nonlinear spatiotemporal dependencies between the stations. The dependencies are not limited to the physical relations of the transportation network, but include the nonlinear spatiotemporal dependencies caused by a variety of external environmental factors. At the same time, the diagonal element is one, indicating the self-impact of each station.*

Definition 2. ***State.*** *For the network-wide traffic flow prediction problem, the change of traffic environment directly affects the model's prediction result. With the different inputs of the prediction model, the prediction results are different. In this study, the prediction results of the network-wide traffic flow prediction model are regarded as the observation state from the environment. Further, the new observation state is the prediction result of the network-wide traffic flow prediction model after using the graph adjacency matrix* $\mathbf{A}_{dyn}$ *fed back by the agent.*

Definition 3. ***Reward function.*** *One way to inform the agent whether the action is reasonable is to construct a reward function. A flexible reward function can help agents learn the underlying patterns of the environment more quickly. In previous studies* (Mnih et al., 2015; Yun et al., 2017)*, an agent obtains the same reward when it takes different actions for the same observation state, making the agent confused about its decision. Therefore, a new reward function is designed in this study*:

$$Reward = \begin{cases} -\xi, & \text{if } |Pre\_Loss| > \xi \\ -|Pre\_Loss|, & \text{otherwise} \end{cases} \tag{10}$$

where *Pre_Loss* denotes the prediction error calculated by Eq. (9) in the current state. The prediction model may be difficult to adapt to some adjacency matrices, which results in extreme deviation of the prediction results from the actual observed values. In this case, we set a lower bound $-\xi$ to constrain the value of the reward function to improve the agent's learning ability, where ξ is a hyper-parameter. We set the value of ξ by observing the fluctuation range of the model's prediction error in the training process.

Based on the above definition, the agent's training process is described in detail as follows.

Step 1: The GCNI model is pre-trained and denoted as $GCNI_{base}$. Meanwhile, inspired by Bai et al. (2020), we construct a basic graph adjacency matrix $\mathbf{A}_{base}$ using the data-adaptive graph generation method as follows:

$$\mathbf{A}_{base} = softmax(relu(\mathbf{W}_1\mathbf{W}_2^{\mathrm{T}})) \tag{11}$$

where $\mathbf{W}_1$, $\mathbf{W}_2 \in \mathbb{R}^{N\times d}$ are two learnable parameter matrices. Therein d denotes the embedding dimension of the matrix. By multiplying $\mathbf{W}_1$ and $\mathbf{W}_2$, an $N \times N$ asymmetric matrix can be obtained, which is used to represent the basic spatiotemporal dependence features of the transportation network. $softmax(\cdot)$ is an activation function for normalizing the matrix.

Step 2: Four kinds of deep neural networks in DDPG are initialized, including critic network$Q(\mathbf{s}, \mathbf{a}|\theta^Q)$, actor network $\mu(\mathbf{s}|\theta^\mu)$, target critic network $Q'(\mathbf{s}, \mathbf{a}|\theta^{Q'})$, and target actor network $\mu'(\mathbf{s}|\theta^{\mu'})$. The network structures of the critic network and actor network are shown in Fig. 3, where N denotes the length of the input vector of the actor network, which is consistent with the total number of stations. 'Dense layer' denotes a fully connected layer, and the numbers in parentheses denote the size of the input and output feature vectors of the layer, respectively. The network structure of the target network is consistent with that of the corresponding actor network. θ^Q, θ^μ, $\theta^{Q'}$, and $\theta^{\mu'}$ are the parameters of the corresponding networks. The initialization weights of $\theta^{Q'}$ and $\theta^{\mu'}$ are the same as θ^Q and θ^μ, respectively.

Step 3: Each training sample is regarded as input into prediction network $GCNI_{base}$ to obtain a corresponding prediction result, which is taken as initial observation state $\mathbf{s}$. The actor network outputs action $\mathbf{a}$ based on state $\mathbf{s}$. Note that the output of the actor network is a one-dimensional vector, which needs to be rearranged into $N \times N$ two-dimensional matrix $\mathbf{A}_{dyn}$. Meanwhile, for different observation states, the actor network will output different one-dimensional vectors to represent the dynamic spatiotemporal dependencies between the stations. Then, $GCNI_{base}$ temporarily replaces the graph adjacency matrix according to the action to obtain network $GCNI_{dyn}$. $GCNI_{dyn}$ outputs a new prediction result as new observation state $\mathbf{s}'$. Meanwhile, the prediction error of the new prediction result is calculated using Eq. (9) to obtain immediate reward r of action $\mathbf{a}$. The state transition information of each moment is defined as $\{\mathbf{s}, \mathbf{a}, r, \mathbf{s}'\}$, which is stored in an experience replay buffer. Repeat Step 3 for P_1 times.

Step 4: The actor network and critic network (Lillicrap et al., 2016) are updated by randomly taking B times of state transition information from the experience replay buffer. The critic network is updated by minimizing the temporal-difference error (*TD_error*) (Rolls et al., 2008), as shown in Eq. (12). The actor network is updated by the sampled policy gradient, as shown in Eq. (13):

$$\begin{cases} TD\_error = \dfrac{1}{B}\displaystyle\sum_{i=1}^{B}[y_i - Q(\mathbf{s}_i, \mathbf{a}_i|\theta^Q)]^2 \\ y_i = r_{i+1} + \gamma Q'(\mathbf{s}_{i+1}, \mu'(\mathbf{s}_{i+1}|\theta^{\mu'})|\theta^{Q'}) \end{cases} \tag{12}$$

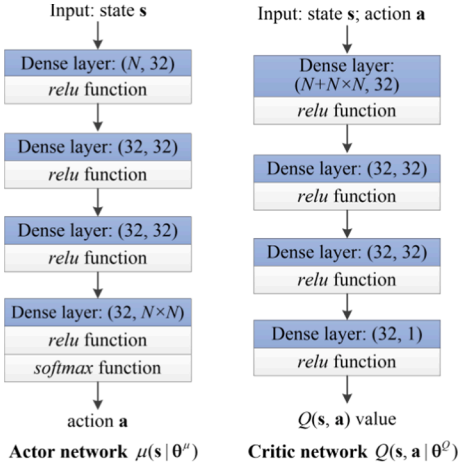


Fig. 3. Structures of actor network and critic network.

$$\nabla_{\theta^{\mu}} J = \frac{1}{B} \sum_{i=1}^{B} \nabla_{\mathbf{a}} Q(\mathbf{s},\ \mathbf{a}|\theta^{Q}) \big|_{\mathbf{s}=\mathbf{s}_i,\ \mathbf{a}_i=\mu(\mathbf{s}_i)} \nabla_{\theta^{\mu}} \mu(\mathbf{s}|\theta^{\mu}) \big|_{\mathbf{s}=\mathbf{s}_i} \tag{13}$$

where y_i denotes the target Q value of target critic network. γ denotes the discount factor. i denotes the index value of the state transition information sample. J denotes an objective function for optimizing the actor network parameters θ^{μ}.

Step 5: When the observation state changes for P_2 times, the network parameters of the target actor network and target critic network are soft updated once, as follows:

$$\theta^{\mu'} \leftarrow \tau\theta^{\mu} + (1-\tau)\theta^{\mu'} \tag{14}$$

$$\theta^{Q'} \leftarrow \tau\theta^{Q} + (1-\tau)\theta^{Q'} \tag{15}$$

where $\tau \in (0,\ 1)$ denotes the update speed. Meanwhile, after P_3 mini-batches of samples are trained each time, all the training samples are used to retrain $GCNI_{base}$. Each sample is assigned a dynamic adjacency matrix during training, which is output from the actor network. The output process of the adjacency matrix can be seen in Step 3.

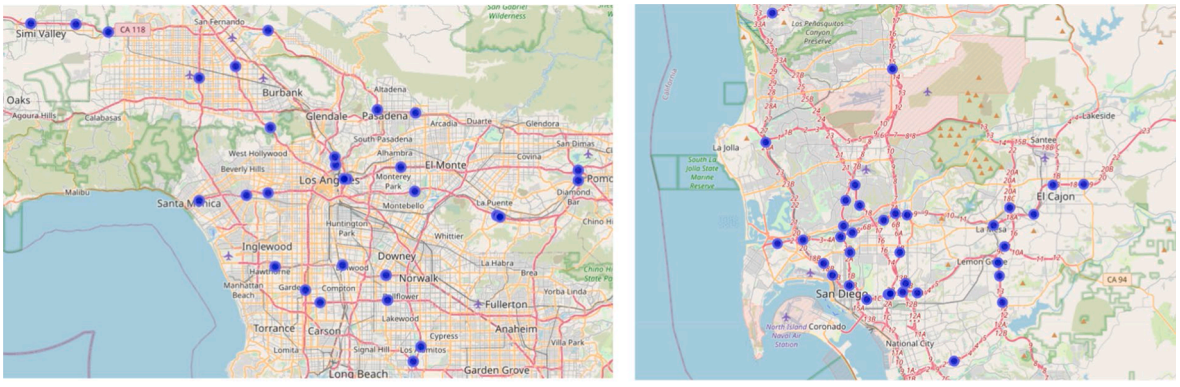
Step 6: Repeat Steps 3–5 to alternately train DDPG and GCNI until the maximum number of iterations E_p is satisfied. Then, GCNI is fine-tuned based on the trained DDPG algorithm to obtain the final RDGCNI model.

In the test phase of RDGCNI, each test sample first gets a preliminary prediction result based on $GCNI_{base}$, which is regarded as the current observation state. Then, the state is input into the actor network of DDPG to obtain the corresponding action feedback, i.e., dynamic graph adjacency matrix $\mathbf{A}_{dyn}$. Subsequently, to enable the prediction model to consider the agent's action feedback, adjacency matrix $\mathbf{A}_{base}$ in GCNI is temporarily replaced by $\mathbf{A}_{dyn}$, and the test sample is predicted again to obtain the next observation state. The new state observed by the agent fully takes into account the real-time dependencies between the stations, so this study takes this as the final prediction result of the model. Moreover, it should be noted that adjacency matrix $\mathbf{A}_{base}$ in $GCNI_{base}$ (defined in Eq. (11)) has been parameterized and learned in the training phase, which can be used to characterize the generalized static spatiotemporal dependencies between the stations, so there is no need of redefining in the test phase.

In general, the proposed dynamic graph learning network based on DDPG can automatically perceive the changes in the traffic environment and give feedback on the dynamic dependency relationship between the stations in real time. In this study, we extract the nonlinear spatiotemporal dependencies between the stations in a data-driven way to avoid relying on expert experience to simulate and construct complex transportation network graphs. Therefore, in generating the dynamic graph adjacency matrix by DDPG, this study does not take the physical relation of the road network as a priori information of the model's input. Based on the proposed data-adaptive dynamic graph learning method, the value of each element in the generated dynamic graph adjacency matrix can not only represent whether there are spatiotemporal dependencies between the stations, but also represent the dependency strength between the stations at different times, which is helpful to improve the accuracy of network-wide traffic flow prediction. At the same time, by using real-world traffic flow datasets for experimental analyses (see Section 4.3), we find that the generated dynamic graph adjacency matrix can effectively extract the general relations between the stations in the road network, and the generated nonlinear spatiotemporal dependencies are also interpretable.

4. Experiments and analyses

To verify the effectiveness of the proposed data imputation method and dynamic graph learning method, this section compares the proposed GCNI and RDGCNI with other data imputation methods and models with data imputation functions on two traffic flow datasets. The experimental configuration and analyses, and discussions on the experimental results are described in detail below.



(a) PeMSD7 dataset (b) PeMSD11 dataset

Fig. 4. Distribution of traffic detection stations for two datasets.

4.1. Datasets and data missing scenarios

In this study, two real-world traffic flow datasets are used to train and test the models. These two datasets (i.e., PeMSD7 and PeMSD11) are collected by the Performance Measurement System (PeMS) of the California Department of Transportation (Chen et al., 2001). The system can measure the traffic state of each freeway in California in real time, such as traffic speed, traffic flow, and time occupancy.

The PeMSD7 dataset contains traffic flow data for 30 stations (detectors) in Los Angeles/Ventura, California, June 1–30, 2017. The PeMSD11 dataset contains traffic speed data for 35 stations in San Diego/Imperial, California, July 1–30, 2017. In both datasets, the data are aggregated every 5 min, resulting in 288 data points per day. The distribution of the traffic detection stations in the two datasets is shown in Fig. 4.

To simulate the scenario of traffic data missing caused by detector failure in real life, a certain proportion of data points are randomly set as zero by artificially setting the missing rate. In this study, three sub-datasets with 30%, 50%, and 70% missing rates, and one sub-dataset with block missing are generated for PeMSD7 and PeMSD11 datasets to test the robustness of the proposed model under different degrees of data missing. Among them, block missing means that based on the dataset with a 50% missing rate, the data points of some detectors are missing continuously for a whole day. Fig. 5 displays examples of data matrices with different missing rates. Meanwhile, the masking matrix used to represent the missing data state of each sub-dataset is obtained accordingly. Finally, for each sub-dataset with missing data, the last seven days (i.e., $7 \times 288 = 2016$ data points) are used for model test, and the rest are used for model training.

4.2. Experimental configuration

This study aims to improve the accuracy and robustness of the GCN-based network-wide traffic flow prediction model by proposing

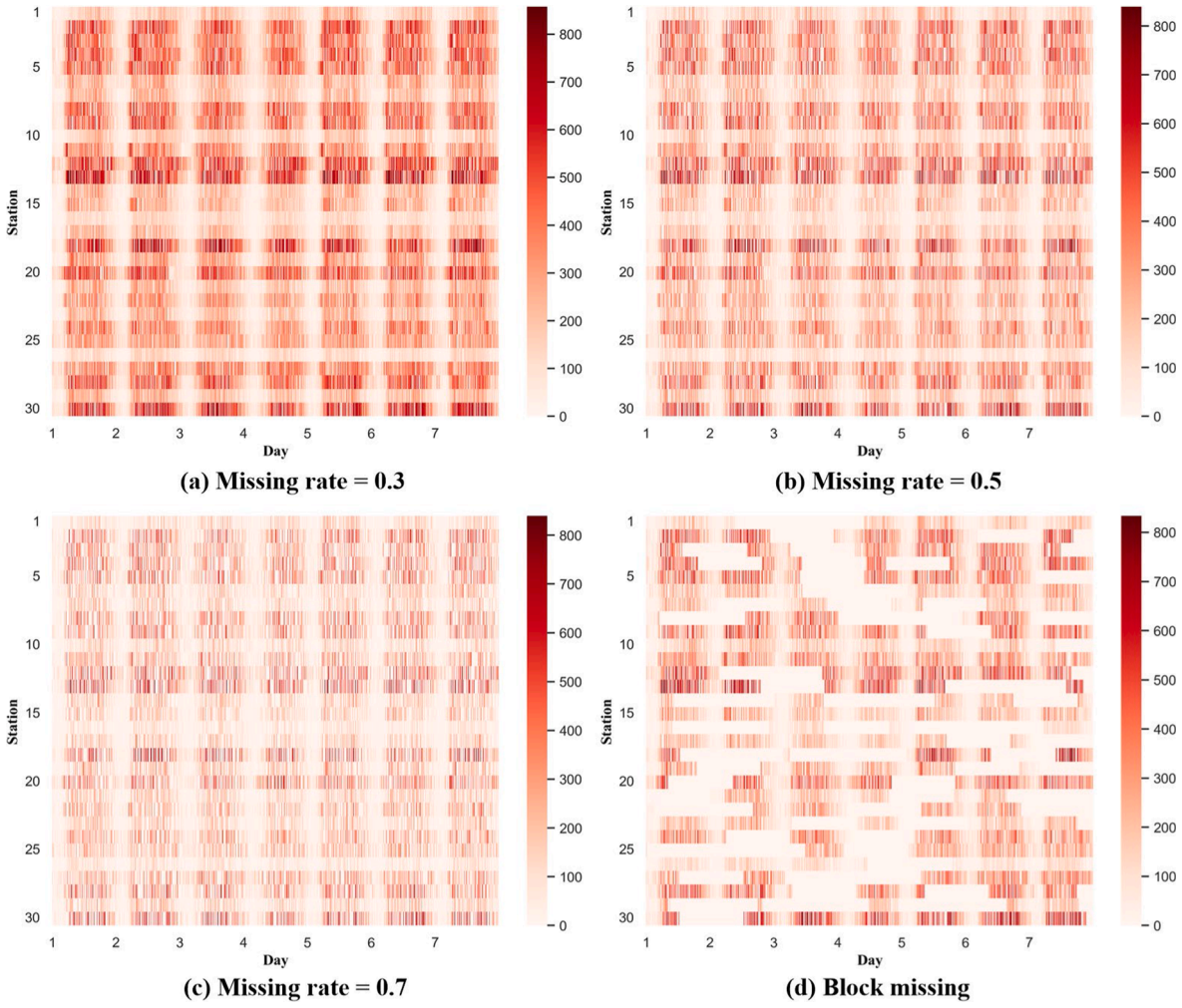


Fig. 5. Example data matrices with different missing rates and missing patterns.

new data imputation methods and new dynamic graph learning methods. Therefore, the accuracy of missing data imputation and network-wide traffic flow prediction is used to quantify and verify the effectiveness of the proposed methods. Meanwhile, four asynchronous data imputation methods are selected to compare the performance of data imputation, including Matrix Factorization (MF) (Koren et al., 2009), MICE (Buuren and Groothuis-Oudshoorn, 2011), SoftImpute (Mazumder et al., 2010), and IterativeSVD (Troyanskaya et al., 2001). Further, the four methods are combined with LSTM and GCN respectively as baseline models for network-wide traffic flow prediction. In addition, two synchronous data imputation methods are also selected for experimental comparison, including the spectral graph Markov network (SGMN) (Cui et al., 2020b) and LSTM-M (Tian et al., 2018). All experimental comparisons are performed on a computer with 64-bit 3.7-GHz Intel Core i7 8700 K CPU and 32 GB of RAM. To ensure fairness of experimental comparison, all of the methods are trained and tested with the same hyper-parameters in all datasets. The hyper-parameters of RDGCNI are determined by the trial-and-error method (Gemperline et al., 1991), as shown in Table 1. Meanwhile, as shown in Fig. 6, by observing the prediction error distribution of RDGCNI in the training process, the lower bound of the reward function is set as -0.01.

Moreover, the performance of all methods is evaluated by three widely used evaluation metrics, including mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean square error (RMSE), which are calculated as follows:

$$\mathrm{MAE} = \frac{\sum_{t=1}^{Te}\sum_{n=1}^{N}\left|\overline{y}_t^n - y_t^n\right| m_t^n}{\sum_{t=1}^{Te}\sum_{n=1}^{N} m_t^n} \tag{16}$$

$$\mathrm{MAPE} = \frac{\sum_{t=1}^{Te}\sum_{n=1}^{N} m_t^n \frac{\left|\overline{y}_t^n - y_t^n\right|}{\left|y_t^n\right|}}{\sum_{t=1}^{Te}\sum_{n=1}^{N} m_t^n} \times 100 \tag{17}$$

$$\mathrm{RMSE\_I} = \sqrt{\frac{\sum_{t=1}^{Te}\sum_{n=1}^{N}\left(1 - m_t^n\right)\left(\widetilde{x}_t^n - x_t^n\right)^2}{\sum_{t=1}^{Te}\sum_{n=1}^{N}\left(1 - m_t^n\right)}} \tag{18}$$

$$\mathrm{RMSE\_P} = \sqrt{\frac{\sum_{t=1}^{Te}\sum_{n=1}^{N} m_t^n\left(\overline{y}_t^n - y_t^n\right)^2}{\sum_{t=1}^{Te}\sum_{n=1}^{N} m_t^n}} \tag{19}$$

where Te denotes the total number of the test dataset. y_t^n and $\overline{y}_t^n$ denote the observed and predicted values of the n-th station at time t, respectively. x_t^n and $\widetilde{x}_t^n$ denote the observed and imputed values of the n-th station at time t, respectively. RMSE_I and RMSE_P are used to evaluate the model's data imputation performance and prediction performance, respectively.

4.3. Comparison and result analysis

In this section, the effectiveness of the proposed method is verified by comparing the proposed method with other baseline methods. To ensure the reliability of the experiment results, the datasets of each missing rate are repeatedly generated three times and used to train and test each method, respectively. The average of the three test results for the proposed method and the other baseline methods on PeMSD7 and PeMSD11 datasets are shown in Table 2, where 'Improvement' represents the improvement percentage of RDGCNI compared to the best baseline method. The detailed analysis of the comparative results is described below.

4.3.1. Performance comparison of network-wide traffic flow prediction

Since the two datasets used in this study record the traffic flow status on the freeway, and the quality of traffic flow data at each

Table 1
Hyper-parameter setting of RDGCNI.

Parameter	Value
Historical time window	3
Learning rate	0.001
Maximum number of iteration E_p	5
Batch size B	288
Parameter optimizer	'Adam'
Order of Chebyshev approximation K	2
Penalty term β	0.1
Reward lower bound $-\xi$	-0.01
Matrix embedding dimension d	3
Discount factor γ	0.9
Update speed τ	0.001
Parameters for counting: P_1, P_2, P_3	5, 2500, 10

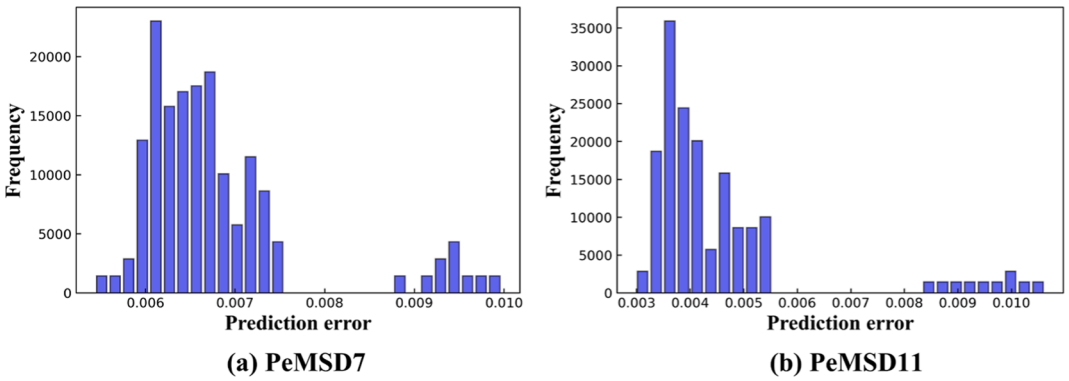


Fig. 6. Prediction error distribution of RDGCNI on two traffic datasets.

station is relatively regular, it can be seen from Table 2 that each baseline model can achieve high prediction accuracy. Nevertheless, when the data of less than one month is used for model training, the proposed GCNI and RDGCNI can further outperform other baseline methods in most of the experiments. For example, for the PeMSD7 dataset with block missing, the MAE, RMSE_P, and MAPE values of RDGCNI are decreased by 11.025%, 7.324%, 7.812%, respectively, compared to the SoftImpute-LSTM model. For the PeMSD11 dataset with a missing rate of 50%, the MAE, RMSE_P, and MAPE values of RDGCNI are decreased by 16.59%, 13.56%, and 11.24%, respectively, compared to the SGMN model. On the other hand, the training time of the RDGCNI model on the PeMSD7 and PeMSD11 datasets is about 250 min and 300 min, respectively. Although RDGCNI takes a long time to train the model, the model's training can be completed offline. When the model needs to be updated with new traffic data, we can fine-tune the parameters based on the previously trained model without re-training a completely new model. Meanwhile, the time of RDGCNI to predict the traffic flow of all stations at the next time point can be stabilized within one second. Therefore, the proposed model can meet practical application requirements.

Among all the baseline methods, the prediction performance of the synchronous data imputation method is generally better than that of the asynchronous data imputation method, which benefits from the fact that the synchronous data imputation method enables the prediction model to effectively capture the missing patterns in the traffic data, thereby alleviating the error accumulation problem caused by data imputation. Meanwhile, for the asynchronous data imputation method, the SoftImpute method has stable performance on different datasets. For the synchronous data imputation method, SGMN and LSTM-M show different performances on different missing rate datasets. For example, SGMN outperforms LSTM-M when the missing rate of the dataset is 30%. The performance of LSTM-M is gradually better than that of SGMN as the missing rate increases. When the dataset has block missing, the performance of LSTM-M is worse than that of SGMN.

Moreover, to verify the prediction effect of the proposed method, the prediction errors of RDGCNI and the other two optimal baseline models (i.e., LSTM-M, and SoftImpute-GCN) are presented in Fig. 7. The histograms, dots, and triangles represent the MAE and MAPE values of all station prediction results obtained by RDGCNI, SoftImpute-GCN, and LSTM-M, respectively, at each time point. The MAE and MAPE values of RDGCNI are significantly lower than LSTM-M and SoftImpute-GCN, especially on datasets with block missing. The reason is that compared with the other two baseline methods, RDGCNI not only effectively imputes missing data, but also timely perceives traffic state fluctuations, generates a dynamic graph adjacency matrix for feedback, and ultimately achieves accurate traffic prediction.

4.3.2. Performance comparison of data imputation

Fig. 8 shows the performance comparison between the proposed method and the other four asynchronous data imputation methods in terms of the imputation error. In general, with the increase of the missing rate, the data imputation errors of all methods show an increasing trend. Meanwhile, the data imputation method based on a multi-graph convolution fusion network proposed in this study can maintain a low error on datasets with different missing rates, which effectively guarantees the prediction accuracy of the prediction model. Although some data imputation methods (e.g., MF, and SoftImpute) can also achieve low imputation errors, these methods are independent of the prediction model. After data imputation, it is difficult for the prediction model to learn the underlying missing patterns in the traffic flow data and then produce a biased prediction result. For example, for the PeMSD11 dataset with block missing, although the imputation errors of MF and SoftImpute are lower than those of our proposed method, the MAE, MAPE, and RMSE_P values of MF-GCN (SoftImpute-GCN) are 0.458 (0.539), 1.146 (1.380), and 0.564 (0.524) higher than those of GCNI, respectively. Moreover, compared with LSTM-M and SGMN, GCNI based on a multi-graph convolutional fusion network can utilize GCN to analyze the propagation law of traffic states between traffic flow detection stations in both time and space dimensions, thus effectively improving the accuracy of data imputation. For example, for the PeMSD11 dataset with a missing rate of 70%, the MAE, RMSE_P, and MAPE values of GCNI are 2.620, 4.173, and 3.352 lower than those of SGMN, respectively.

4.3.3. Dynamic graph learning ability and interpretability analysis

By comparing RDGCNI with GCNI, the effectiveness of the proposed dynamic graph learning method based on DRL can be further

Table 2
Prediction results of different methods on two traffic datasets.

Missing Rate	Model	PeMSD7			PeMSD11		
		MAE	MAPE	RMSE_P	MAE	MAPE	RMSE_P
30 %	MF-LSTM	27.043	13.938	37.811	2.425	5.200	3.948
	MICE-LSTM	23.035	9.950	34.394	2.045	4.604	3.595
	SoftImpute-LSTM	22.661	9.756	33.756	1.913	4.375	3.478
	IterativeSVD-LSTM	23.083	10.008	34.143	1.973	4.528	3.539
	MF-GCN	24.997	14.130	35.672	1.808	4.780	3.601
	MICE-GCN	21.462	9.581	32.626	1.856	4.012	3.150
	SoftImpute-GCN	19.983	8.482	30.200	1.551	3.597	2.845
	IterativeSVD-GCN	20.574	8.995	30.874	1.513	3.304	2.806
	SGMN	19.711	8.240	30.397	1.253	2.411	2.242
	LSTM-M	21.144	9.683	31.028	2.053	4.453	3.465
	GCNI (ours)	19.203	8.331	29.142	1.301	2.799	2.549
	RDGCNI (ours)	**18.512**	**7.849**	**28.391**	**1.164**	**2.393**	**2.232**
	Improvement (%)	+6.09%	+4.74%	+5.99%	+7.12%	+0.73%	+0.47%
50%	MF-LSTM	29.209	15.161	40.257	2.550	5.491	4.163
	MICE-LSTM	24.984	11.037	36.830	2.200	4.981	3.895
	SoftImpute-LSTM	24.143	10.392	35.382	2.125	4.787	3.772
	IterativeSVD-LSTM	25.440	10.862	37.514	2.243	5.137	4.023
	MF-GCN	31.283	18.356	43.912	2.342	6.490	4.592
	MICE-GCN	23.768	10.987	35.577	2.544	5.781	4.110
	SoftImpute-GCN	23.063	9.929	34.311	1.950	4.595	3.476
	IterativeSVD-GCN	28.054	12.155	44.195	2.309	4.922	3.935
	SGMN	21.840	10.217	34.469	1.796	3.805	3.376
	LSTM-M	22.280	9.781	32.710	2.165	4.251	3.752
	GCNI (ours)	21.512	9.336	32.417	1.565	3.571	3.179
	RDGCNI (ours)	**20.472**	**8.730**	**31.148**	**1.498**	**3.289**	**2.996**
	Improvement (%)	+6.26%	+10.74%	+4.78%	+16.59%	+13.56%	+11.24%
70%	MF-LSTM	33.569	16.625	47.091	2.751	6.038	4.653
	MICE-LSTM	28.389	12.960	41.049	2.603	6.057	4.640
	SoftImpute-LSTM	27.432	11.697	40.150	2.404	5.479	4.300
	IterativeSVD-LSTM	29.301	12.781	42.589	3.043	6.907	5.301
	MF-GCN	34.029	19.134	47.901	2.951	8.723	5.768
	MICE-GCN	29.642	14.257	43.344	3.595	7.822	5.506
	SoftImpute-GCN	29.777	12.926	44.343	2.632	6.870	4.649
	IterativeSVD-GCN	48.263	22.584	74.173	5.266	11.880	7.568
	SGMN	36.416	20.132	59.287	5.005	9.767	7.796
	LSTM-M	26.516	11.588	38.376	2.728	5.504	4.762
	GCNI (ours)	26.888	11.797	40.603	2.385	5.595	4.443
	RDGCNI (ours)	**25.423**	**11.013**	**38.165**	**2.324**	**5.228**	**4.283**
	Improvement (%)	+4.12%	+4.97%	+0.55%	+3.33%	+4.58%	+0.40%
Block missing	MF-LSTM	28.654	13.728	40.480	2.739	5.974	4.473
	MICE-LSTM	26.400	11.832	37.998	2.329	5.387	4.202
	SoftImpute-LSTM	23.930	9.745	35.315	2.137	4.948	3.879
	IterativeSVD-LSTM	27.020	11.268	39.523	2.773	6.401	4.819
	MF-GCN	28.072	13.463	40.908	2.009	4.734	3.739
	MICE-GCN	26.251	12.104	38.942	2.925	6.882	4.806
	SoftImpute-GCN	26.859	11.361	40.463	2.090	4.968	3.699
	IterativeSVD-GCN	55.004	21.240	88.207	4.231	9.075	6.535
	SGMN	23.219	14.925	39.874	1.879	7.786	4.427
	LSTM-M	47.622	16.386	64.413	6.327	10.935	8.425
	GCNI (ours)	21.821	9.432	33.226	1.551	3.587	3.175
	RDGCNI (ours)	**21.291**	**9.032**	**32.556**	**1.501**	**3.392**	**3.042**
	Improvement (%)	+8.30%	+7.32%	+7.81%	+20.11%	+28.34%	+17.74%

Note: bold font represents the optimal value of each metric. Underline represents the best value obtained by the baseline methods with respect to each metric.

verified. On the basis of GCNI, RDGCNI uses the DRL algorithm (i.e., DDPG) to perceive the changes in the traffic environment and dynamically generates a graph adjacency matrix for feedback. Therefore, compared with GCNI, RDGCNI can predict each station's complex and dynamic traffic state more robustly in the transportation network. Fig. 9 shows the prediction performance of GCNI and RDGCNI on different days in the form of boxplots. Each boxplot shows the distribution of MAE values obtained by testing each station with 24 h of traffic data (i.e., 288 data points) as the test sample, where the red dot represents the average value of MAE. Meanwhile, the *p*-value represents the significance level of the difference in performance between GCNI and RDGCNI based on the *t*-test (Cressie and Whitford, 1986). A *p*-value less than 0.05 indicates a significant difference in the performance of the two methods. It can be seen from Fig. 9 that the distribution of the MAE values of RDGCN is generally better than that of GCNI in terms of the maximum, median, minimum, and mean.

Furthermore, Fig. 10 visualizes the spatiotemporal dependencies between the stations included in the dynamic graph adjacency

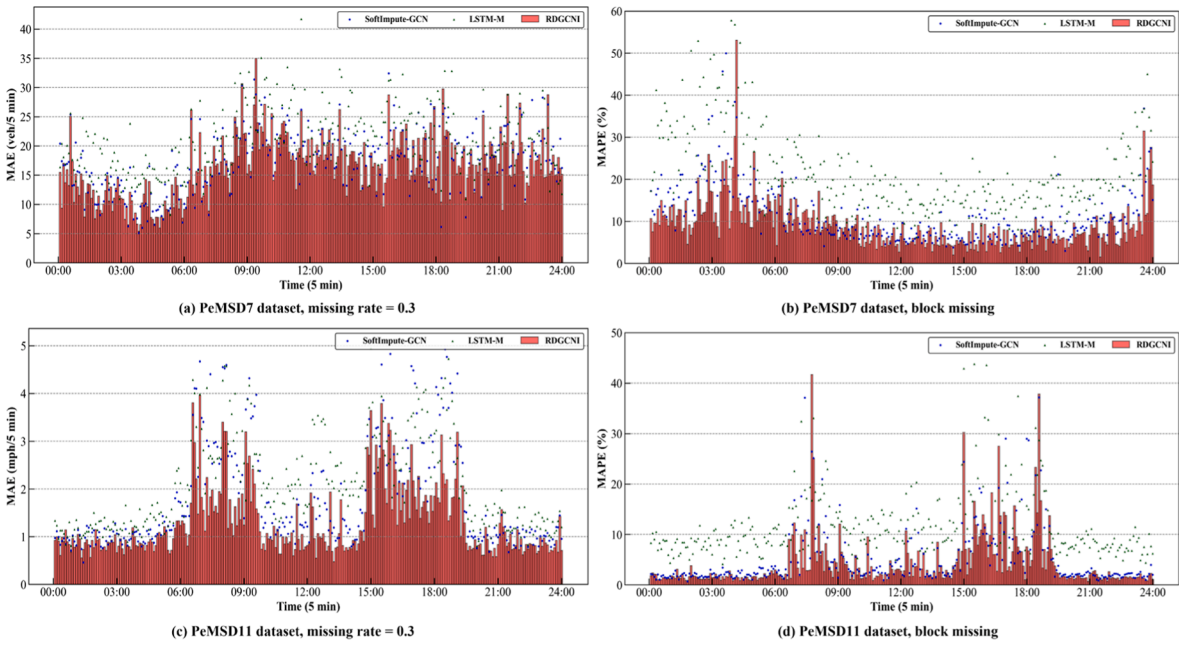


Fig. 7. Performance comparison of the three best data imputation methods on PeMSD7 and PeMSD11 datasets with different missing patterns.

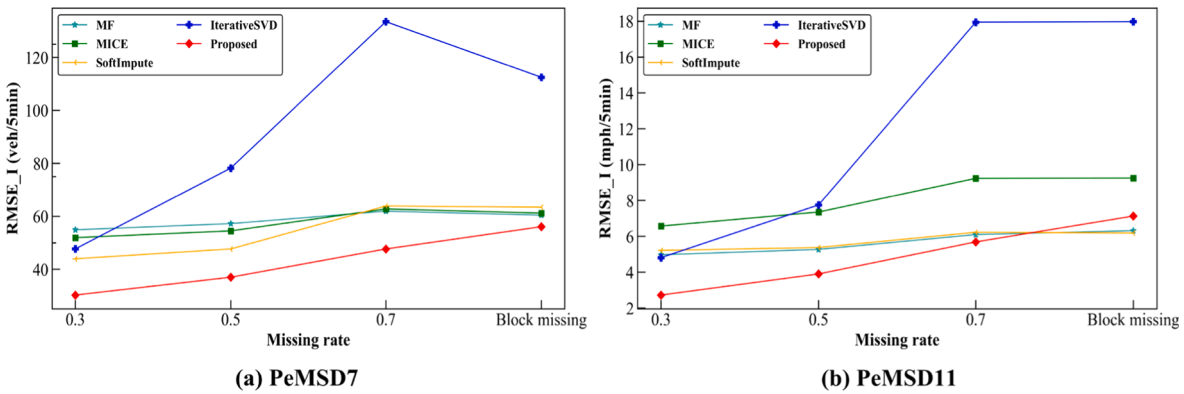


Fig. 8. Performance comparison of different imputation methods on two traffic datasets.

matrix generated by RDGCNI in the form of heatmaps and maps. Fig. 10**(a)** shows the distribution of the stations in the PeMSD11 dataset. Fig. 10**(b)** shows the traffic flow changes at all stations for seven consecutive days. Meanwhile, to better analyze the content of the figure, we only display the corresponding traffic flow values at the one-hour interval. From Fig. 10**(b)**, we can see that the traffic flow characteristics of some stations (e.g., stations 17, 18, and 19) have strong periodicity. The corresponding stations are often congested at 8:00 am and 5:00 pm, resulting in lower traffic speeds. Fig. 10**(c)-(e)** display the dynamic graph adjacency matrices generated by RDGCNI at different time points. Heatmap colors indicate the spatiotemporal dependencies between the corresponding stations, with darker colors representing stronger dependencies. In terms of spatiotemporal dependencies of the road network, as shown in the white rectangle of Fig. 10**(d)**, station 27 is strongly correlated with stations 11, 24, 29, 30, and 34. First, station 27 has an upstream and downstream relationship with stations 29, 30, and 34. The traffic flow of upstream and downstream stations not only has similar traffic flow characteristics with the intermediate station (i.e., station 27), but also affects or is affected by the future traffic flow of the intermediate station, such as the spread of traffic congestion. Therefore, these stations have strong spatial and temporal dependencies. Second, although stations 27, 11, and 24 are at different freeways, they are located near the intersection and have similar traffic flow variation characteristics. Moreover, it should be noted that the traffic flow data of all stations at multiple historical time points are used as the input of the prediction model in this study, so the adjacency matrix in Fig. 10**(c)-(e)** also contains implicit temporal dependencies between the stations. In terms of traffic flow prediction, taking adjacency matrix $\mathbf{A}_{dyn}$ 1 as an example, Fig. 10 **(f)** shows the top ten stations that have the most significant influences on predicting the traffic flow of each station at the next time point. The station influence value is calculated by summing up all the weight values of each station's corresponding rows and columns

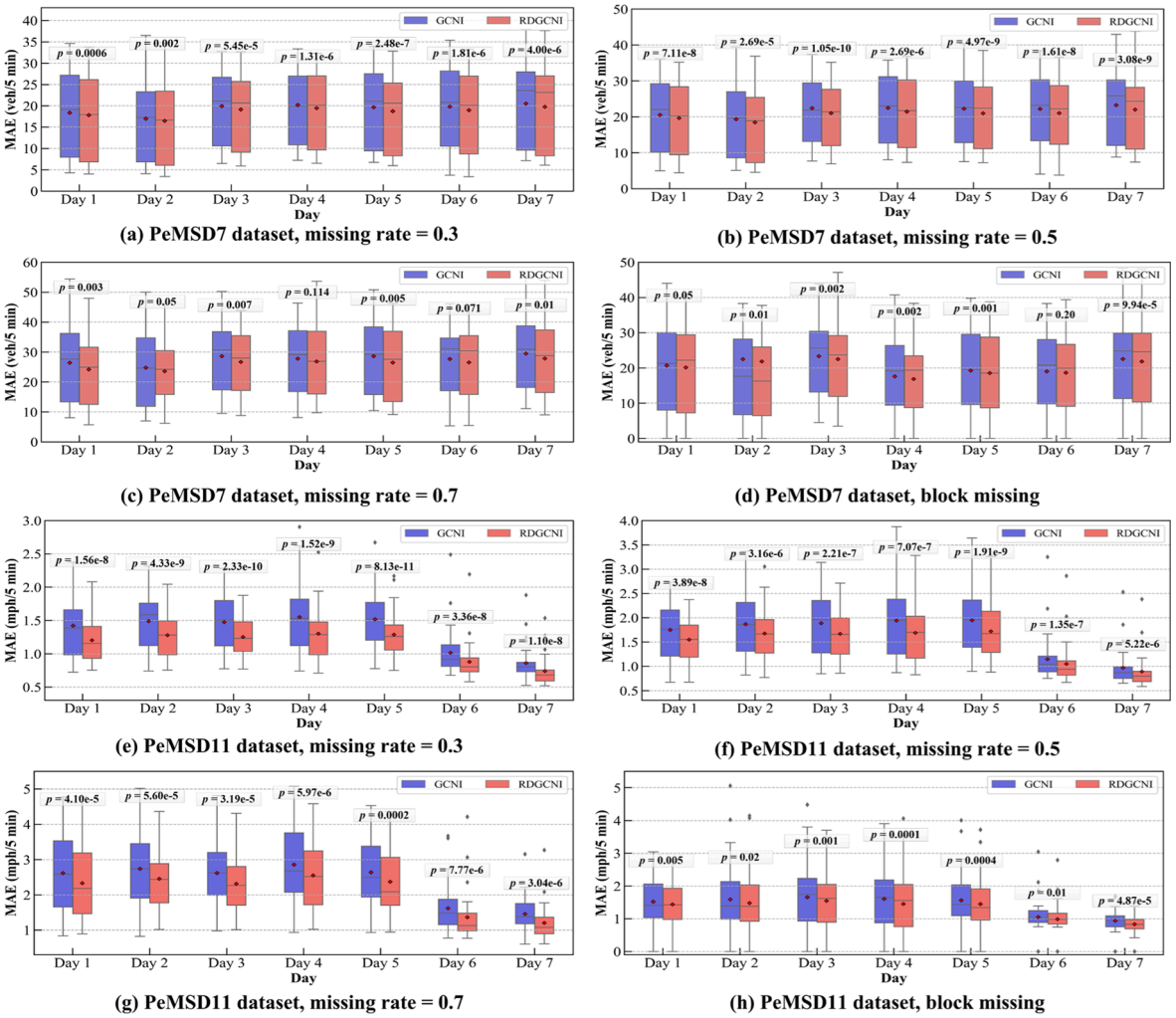


Fig. 9. Statistical test results of GCNI and RDGCNI on PeMSD7 and PeMSD11 datasets.

in$\mathbf{A}_{dyn}$ 1. By sorting the influence values of all stations in the descending order, we selected the top ten stations for visualization. As shown in Fig. 10**(f)**, most of the influential stations are distributed around in the intersection and its upstream and downstream areas, and their traffic state has a strong temporal and spatial correlation with the traffic flow of surrounding stations.

To more clearly display the variation of the graph adjacency matrix values at different time points, Fig. 10**(g)-(h)** show the residual matrix (denoted as **Res_M1** and **Res_M2**) of the dynamic graph matrix sequence (i.e., $\mathbf{A}_{dyn}$ 1,$\mathbf{A}_{dyn}$ 2, and $\mathbf{A}_{dyn}$ 3). From Fig. 10**(g)-(h)**, it can be seen that the graph adjacency matrix generated by RDGCNI at different time points has a significant change. The reason is that RDGCNI can make timely feedback according to the traffic state observed at different time points, that is, to extract the complex nonlinear spatiotemporal dependencies between the stations, which is not limited to the road topology structure. For example, in residual matrix **Res_M1**, both stations 15 and 17 are at a regular freeway (i.e., non-intersection) and have similar speeds at 8:00 am on July 24. However, at 5:00 pm on July 26, there was a slight increase in traffic speed at station 15, but station 17 was congested for some reason. Thereby, as shown in the black squares of Fig. 10**(g)**, the correlation between station 15 and station 17 is significantly reduced. In addition, station 13 and station 8 are in a similar situation. As shown in residual matrix **Res_M2**, since each road section is relatively smooth in the early morning, the traffic flow of each station has similar characteristics. Therefore, the nonlinear spatiotemporal dependencies between the stations will be significantly weakened, resulting in negative values of most elements of **Res_M2**.

5. Conclusions and future work

In this study, a reinforced dynamic graph convolution network model with data imputation is proposed to improve the accuracy and robustness of network-wide traffic flow prediction in the situation of traffic station data missing. Specifically, the proposed model combines a multi-graph convolutional fusion network to impute the missing data and a DRL technology to perceive the changes in the traffic environment. The proposed method is tested and compared with other baseline methods on two real-world traffic datasets. The

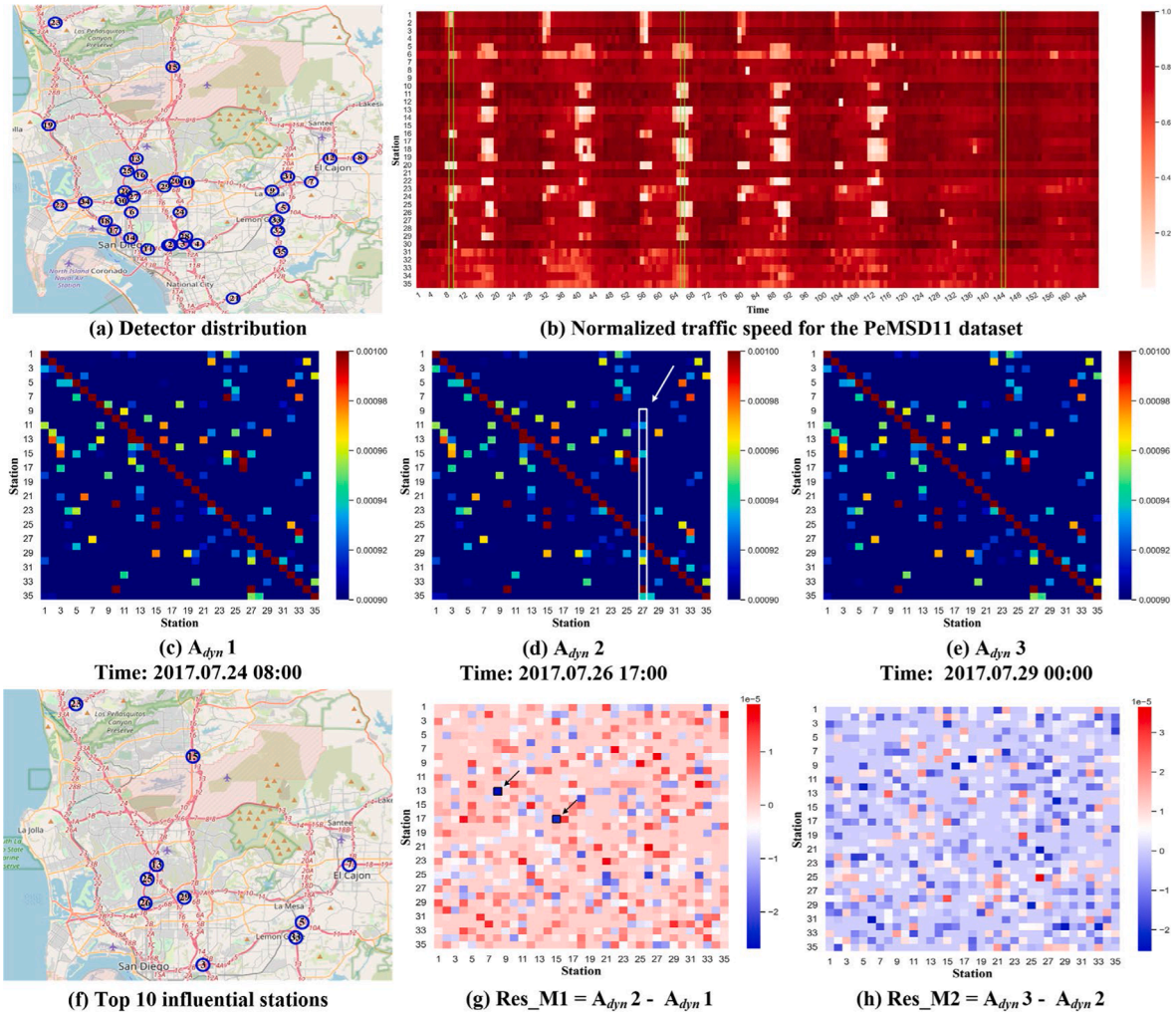


Fig. 10. Visualization of dynamic graph adjacency matrices and residual matrices of RDGCNI.

three conclusions are drawn as follows: (a) The proposed method is competent to handle network-wide traffic flow prediction problems with different data missing rates and is superior to other baseline methods in different network-wide traffic flow prediction tasks. (b) The proposed data imputation method based on a multi-graph convolutional fusion network can utilize GCN to effectively analyze the propagation law of traffic states between traffic flow detection stations and then enhance the accuracy of data imputation. (c) The proposed dynamic graph learning method based on DRL can generate a graph adjacency matrix in real time to feedback the changes of the traffic environment and then improve the robustness and accuracy of network-wide traffic flow prediction. The generated adjacency matrix can effectively represent the spatiotemporal dependencies between the stations and has good interpretability.

This study has some limitations worthy of exploring and solving in future work. First, this study provides a new perspective to employ DRL technology to optimize the adjacency matrix of GCN dynamically. A fully connected structure is used in RDGCNI, resulting in high computational costs. Therefore, future research work may design a more lightweight network structure in RDGCNI to reduce the model complexity and improve the accuracy of network-wide traffic flow prediction. At the same time, the fully connected structure limits the input of the actor network to be a one-dimensional vector. Therefore, this study focuses on one-step traffic flow prediction, using the predicted traffic flow values of each station at the next time point as the observation state of the DRL model, and taking them as the input of the action network to obtain the corresponding action feedback. In future research, we will design more robust actor networks and extend the proposed model to carry out multi-step traffic flow prediction tasks.

CRediT authorship contribution statement

Yong Chen: Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft. **Xiqun (Michael) Chen:** Conceptualization, Methodology, Funding acquisition, Project administration, Resources, Supervision, Writing – review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research is financially supported by the National Natural Science Foundation of China (71922019, 72171210), National Key Research and Development Program of China (2020AAA0107400), and joint project of the National Natural Science Foundation of China and Joint Programming Initiative Urban Europe (NSFC – JPI UE) ('U-PASS', 71961137005).

References

Amiri, M., Jensen, R., 2016. Missing data imputation using fuzzy-rough methods. Neurocomputing 205, 152–164.
Bai, L., Yao, L., Li, C., Wang, X., Wang, C., 2020. Adaptive graph convolutional recurrent network for traffic forecasting. In: Proceedings of the 34th International Conference on Advances in Neural Information Processing Systems, Vancouver, Canada, 33, pp. 17804–17815.
Boquet, G., Morell, A., Serrano, J., Vicario, J.L., 2020. A variational autoencoder solution for road traffic forecasting systems: Missing data imputation, dimension reduction, model selection and anomaly detection. Transportation Research Part C: Emerging Technologies 115, 102622.
Buuren, S.V., Groothuis-Oudshoorn, K., 2011. Mice: Multivariate imputation by chained equations in R. J. Stat. Softw. 45 (3), 1–67.
Che, Z.P., Purushotham, S., Cho, K., Sontag, D., Liu, Y., 2018. Recurrent neural networks for multivariate time series with missing values. Sci. Rep. 8, 6085.
Chen, Y.Y., Lv, Y.S., Wang, F.Y., 2019. Traffic flow imputation using parallel data and generative adversarial networks. IEEE Trans. Intell. Transp. Syst. 21 (4), 1624–1630.
Chen, C., Petty, K., Skabardonis, A., Varaiya, P., Jia, Z., 2001. Freeway performance measurement system: Mining loop detector data. Transp. Res. Rec. 1748 (1), 96–102.
Cressie, N.A.C., Whitford, H.J., 1986. How to use the two sample *t*-test. Biometrical Journal 28 (2), 131–148.
Cui, Z.Y., Henrickson, K., Ke, R.M., Wang, Y.H., 2019. Traffic graph convolutional recurrent neural network: A deep learning framework for network-scale traffic learning and forecasting. IEEE Trans. Intell. Transp. Syst. 21 (11), 4883–4894.
Cui, Z., Ke, R., Pu, Z., Wang, Y., 2020a. Stacked bidirectional and unidirectional LSTM recurrent neural network for forecasting network-wide traffic state with missing values. Transportation Research Part C: Emerging Technologies 118, 102674.
Cui, Z., Lin, L., Pu, Z., Wang, Y., 2020b. Graph Markov network for traffic forecasting with missing data. Transportation Research Part C: Emerging Technologies 117, 102671.
Davis, C., 1962. The norm of the Schur product operation. Numer. Math. 4 (1), 343–344.
Defferrard, M., Bresson, X., & Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. In *Proceedings of the 30th International Conference on Advances in Neural Information Processing Systems*, Barcelona, Spain, pp. 3844–3852.
Du, J.H., Chen, H.Y., Zhang, W.N., 2019. A deep learning method for data recovery in sensor networks using effective spatio-temporal correlation data. Sensor Review 39 (2), 208–217.
Du, S.D., Li, T.R., Gong, X., Horng, S.J., 2020. A hybrid method for traffic flow forecasting using multimodal deep learning. International Journal of Computational Intelligence Systems 13 (1), 85–97.
Duan, Y.J., Lv, Y.S., Liu, Y.L., Wang, F.Y., 2016. An efficient realization of deep learning for traffic data imputation. Transportation Research Part C: Emerging Technologies 72, 168–181.
El-Fiqi, H., Kasmarik, K., Bezerianos, A., Tan, K. C., & Abbass, H. A. (2019). Gate-layer autoencoders with application to incomplete EEG signal recovery. In *Proceedings of the 2019 International Joint Conference on Neural Networks*, Budapest, Hungary, pp. 1–8.
Gemperline, P.J., Long, J.R., Gregoriou, V.G., 1991. Nonlinear multivariate calibration using principal components regression and artificial neural networks. Anal. Chem. 63 (20), 2313–2323.
Geng, X.u., Li, Y., Wang, L., Zhang, L., Yang, Q., Ye, J., Liu, Y., 2019. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. AAAI 33, 3656–3663.
Gu, S. X., Holly, E., Lillicrap, T., & Levine, S. (2017). Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *Proceedings of the 2017 IEEE International Conference on Robotics and Automation*, Singapore, Singapore, pp. 3389–3396.
Guo, S. N., Lin, Y. F., Feng, N., Song, C., & Wan, H. Y. (2019). Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the 33rd AAAI Conference on Artificial Intelligence*, Hawaii, USA, pp. 922–929.
Hong, W.C., 2011. Traffic flow forecasting by seasonal SVR with chaotic simulated annealing algorithm. Neurocomputing 74 (12–13), 2096–2107.
Huang, W.H., Song, G.J., Hong, H.K., Xie, K.Q., 2014. Deep architecture for traffic flow prediction: Deep belief networks with multitask learning. IEEE Trans. Intell. Transp. Syst. 15 (5), 2191–2201.
Jiang, W., & Luo, J. (2021). Graph neural network for traffic forecasting: A survey. *arXiv preprint arXiv:2101.11174*.
Jo, D., Yu, B., Jeon, H., Sohn, K., 2018. Image-to-image learning to predict traffic speeds by considering area-wide spatio-temporal dependencies. IEEE Trans. Veh. Technol. 68 (2), 1188–1197.
Junninen, H., Niska, H., Tuppurainen, K., Ruuskanen, J., Kolehmainen, M., 2004. Methods for imputation of missing values in air quality data sets. Atmos. Environ. 38 (18), 2895–2907.
Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *Proceedings of the 5th International Conference on Learning Representations*, Toulon, France, pp. 1–14.
Konda, V. R., & Tsitsiklis, J. N. (1999). Actor-critic algorithms. In *Proceedings of the Advances in Neural Information Processing Systems*, Denver, USA, pp. 1008–1014.
Koren, Y., Bell, R., Volinsky, C., 2009. Matrix factorization techniques for recommender systems. Computer 42 (8), 30–37.
Kuderer, M., Gulati, S., & Burgard, W. (2015). Learning driving styles for autonomous vehicles from demonstration. In *Proceedings of the 2015 IEEE International Conference on Robotics and Automation*, Seattle, USA, pp. 2641–2646.
Lee, S., Fambro, D.B., 1999. Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting. Transp. Res. Rec. 1678 (1), 179–188.
Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., & Tassa, Y., et al. (2016). Continuous control with deep reinforcement learning. In *Proceedings of the 4th International Conference on Learning Representations*, San Juan, USA, pp. 1–14.
Liu, L., Chen, J., Wu, H., Zhen, J., Li, G., Lin, L., 2020. Physical-virtual collaboration modeling for intra-and inter-station metro ridership prediction. IEEE Trans. Intell. Transp. Syst. 23 (4), 3377–3391.
Luo, Y. H, Zhang, Y., Cai, X. R., & Yuan, X. J. (2019). E^2GAN: End-to-end generative adversarial network for multivariate time series imputation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, Macao, China, pp. 3094–3100.
Lv, M.Q., Hong, Z.X., Chen, L., Chen, T.M., Zhu, T.T., Ji, S.L., 2020. Temporal multi-graph convolutional network for traffic flow prediction. IEEE Trans. Intell. Transp. Syst. 22 (6), 3337–3348.
Mazumder, R., Hastie, T., Tibshirani, R., 2010. Spectral regularization algorithms for learning large incomplete matrices. Journal of Machine Learning Research 11 (80), 2287–2322.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, pp. 807–814.

Rolls, E.T., McCabe, C., Redoute, J., 2008. Expected value, reward outcome, and temporal difference error representations in a probabilistic decision task. Cereb. Cortex 18 (3), 652–663.

Sun, S.L., Zhang, C.S., Yu, G.Q., 2006. A Bayesian network approach to traffic flow forecasting. IEEE Trans. Intell. Transp. Syst. 7 (1), 124–132.

Tian, Y., Zhang, K.L., Li, J.Y., Lin, X.X., Yang, B.L., 2018. LSTM-based traffic flow prediction with missing data. Neurocomputing 318, 297–305.

Troyanskaya, O., Cantor, M., Sherlock, G., Brown, P., Hastie, T., Tibshirani, R., Botstein, D., Altman, R.B., 2001. Missing value estimation methods for DNA microarrays. Bioinformatics 17 (6), 520–525.

Wang, J.W., Chen, R.X., He, Z.C., 2019. Traffic speed prediction for urban transportation network: A path based deep learning approach. Transportation Research Part C: Emerging Technologies 100, 372–385.

Wei, P., Xia, S., Chen, R., Qian, J., Li, C., Jiang, X., 2020. A deep-reinforcement-learning-based recommender system for occupant-driven energy optimization in commercial buildings. IEEE Internet Things J. 7 (7), 6402–6413.

Williams, B.M., Durvasula, P.K., Brown, D.E., 1998. Urban freeway traffic flow prediction: Application of seasonal autoregressive integrated moving average and exponential smoothing models. Transp. Res. Rec. 1644 (1), 132–141.

Xing, Y., Ban, X., Guo, C., 2017. Probabilistic forecasting of traffic flow using multikernel based extreme learning machine. Sci. Program. 2017, 1–12.

Yang, B.L., Sun, S.L., Li, J.Y., Lin, X.X., Tian, Y., 2019. Traffic flow prediction using LSTM with feature enhancement. Neurocomputing 332, 320–327.

Yang, H., Yang, J., Han, L.D., Liu, X., Pu, L.i., Chin, S.-M., Hwang, H.-L., Ma, X., 2018. A Kriging based spatiotemporal approach for traffic volume data imputation. PLoS ONE 13 (4), e0195957.

Yoon, J., Jordon, J., & Schaar, M. (2018). Gain: Missing data imputation using generative adversarial nets. In *Proceedings of the 35th International Conference on Machine Learning*, Stockholm, Sweden, pp. 5689–5698.

Yu, B., Yin, H. T., & Zhu, Z. X. (2018). Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, pp. 3634–3640.

Yun, S., Choi, J., Yoo, Y., Yun, K., & Young Choi, J. (2017). Action-decision networks for visual tracking with deep reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, USA, pp. 1349–1358.

Zhang, H., Chen, P., Zheng, J.F., Zhu, J.Q., Yu, G.Z., Wang, Y.P., Liu, H.X., 2019. Missing data detection and imputation for urban ANPR system using an iterative tensor decomposition approach. Transport. Res. Part C: Emerg. Technol. 107, 337–355.

Zhu, H., Luo, Y.i., Liu, Q., Fan, H., Song, T., Yu, C.W., Du, B., 2019. Multistep flow prediction on car-sharing systems: A multi-graph convolutional neural network with attention mechanism. Int. J. Software Eng. Knowl. Eng. 29 (11n12), 1727–1740.