A Graph-Based Temporal Attention Framework for Multi-Sensor Traffic Flow Forecasting

Shaokun Zhang, Yao Guo, *Member, IEEE*, Peize Zhao, Chuanpan Zheng, and Xiangqun Chen

***Abstract*—Accurate spatio-temporal traffic forecasting serves as the basis of dynamic strategy and applications for intelligent transportation systems, which is of great practical significance for improving traffic safety and mitigating road congestion. Recently, deep learning methods such as convolutional neural networks (CNN) have been applied to traffic flow forecasting, which exhibits better performance than conventional methods. However, these CNN-based methods typically learn traffic as images to model spatial correlation, which is only applicable to Euclidean grid map data rather than non-Euclidean multi-sensor data. To address this problem, we propose a graph-based temporal attention framework GTA, which considers both spatial and temporal correlation, to forecast traffic flow based on data collected from multiple sensors. More specifically, GTA can better capture spatial dependencies leveraging graph embedding techniques on sensor networks because it preserves more details in the algorithms. We also introduce an attention mechanism to adaptively identify the relations among temporal submodules. Spatio-temporal dependencies are more effectively and comprehensively integrated due to the full use of the topological properties of transportation networks. We evaluate GTA with a large-scale traffic dataset from England and enhance it with topology information. The experimental results show that our approach outperforms several state-of-the-art baselines.**

***Index Terms*—Traffic flow prediction, graph embedding, attention mechanism, road network distance.**

I. Introduction

RECENTLY, with the acceleration of smart city construction, more and more sensor devices have been deployed in the intricate urban road network to collect real-time traffic data, laying a solid foundation for the development of intelligent transportation systems (ITS). In order to protect public traveling from congestion, it is necessary to forecast spatio-temporal traffic flow accurately, which can provide decision support for traffic management, such as traffic signal changes and temporary traffic control. Therefore, it has gradually attracted the attention of many researchers [1]–[4], with wide applications in transportation anomaly detection, optimal resource allocation, logistics supply chain, and city management [5]–[10].

Inherent patterns often exist in the traffic flow, which indicates that it is typically feasible to predict it accurately to some extent [11]–[14]. Extensive research has been carried out on traffic flow prediction during the past decades, including approaches such as autoregressive integrated moving average (ARIMA) [15], vector autoregressive (VAR) [16], and multilayer perceptron (MLP) [17]. Recently, deep models, such as long short-term memory network (LSTM) [18] and CNN [19], exhibit better predictive performance than shallow ones, because of their strong capability to model temporal or spatial dependencies [19]–[23].

For instance, Ma *et al.* [19] proposed to learn traffic as an image and leverage CNN to extract abstract features for network-wide traffic speed forecasting. The average accuracy of this method is higher than other baselines by 42.91%. In order to incorporate spatio-temporal dependencies of traffic data, many researchers developed models that combine CNN with LSTM to forecast traffic volumes, such as ConvLSTM [24], DMVST-Net [25], CLTFP [26], etc. However, these CNN-based methods are not suitable for non-Euclidean multi-sensor data, due to their inability to extract shared knowledge among the spatially correlated sensors.

Figure 1 illustrates an instance based on multiple sensors. There exists a closer Euclidean distance between Point A and Point B, which represent two sensors in a freeway network, but a farther driving distance. Besides, there are quite different variations in the magnitude of traffic flow, since these two sensors are deployed on the opposite side of the road. This instance demonstrates that spatial dependency depends to a large extent on the physical geographical conditions and the distribution of sensors. It is not supposed to be Euclidean, but to be dominated by the road topology.

Currently, there are a few studies on spatial dependency modeling using road topology [27]–[35]. For example, Li *et al.* [27] proposed a diffusion convolutional recurrent neural network (DCRNN) that integrated the scheduled sampling technique and the encoder-decoder architecture to mine spatio-temporal dependencies from historical patterns for traffic flow forecasting. However, the adjacency matrix was extracted from the pairwise road network distances by

Manuscript received 27 August 2020; revised 6 December 2020, 9 March 2021, and 27 March 2021; accepted 2 April 2021. Date of publication 16 April 2021; date of current version 8 July 2022. This work was supported in part by the National Key Research and Development Program under Grant 2017YFB1001904 and in part by the National Natural Science Foundation of China under Grant 61772042. The Associate Editor for this article was Y. Kamarianakis. *(Corresponding author: Yao Guo.)*

Shaokun Zhang, Yao Guo, and Xiangqun Chen are with the Key Laboratory on High-Confidence Software Technologies (MOE), Department of Computer Science, School of EECS, Peking University, Beijing 100871, China (e-mail: skzhang@pku.edu.cn; yaoguo@pku.edu.cn; cherry@sei.pku.edu.cn).

Peize Zhao is with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong (e-mail: eezhaopeize@ust.hk).

Chuanpan Zheng is with the Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Informatics, Digital Fujian Institute of Urban Traffic Big Data Research, Xiamen University, Xiamen 361005, China (e-mail: zhengchuanpan@stu.xmu.edu.cn).

Digital Object Identifier 10.1109/TITS.2021.3072118

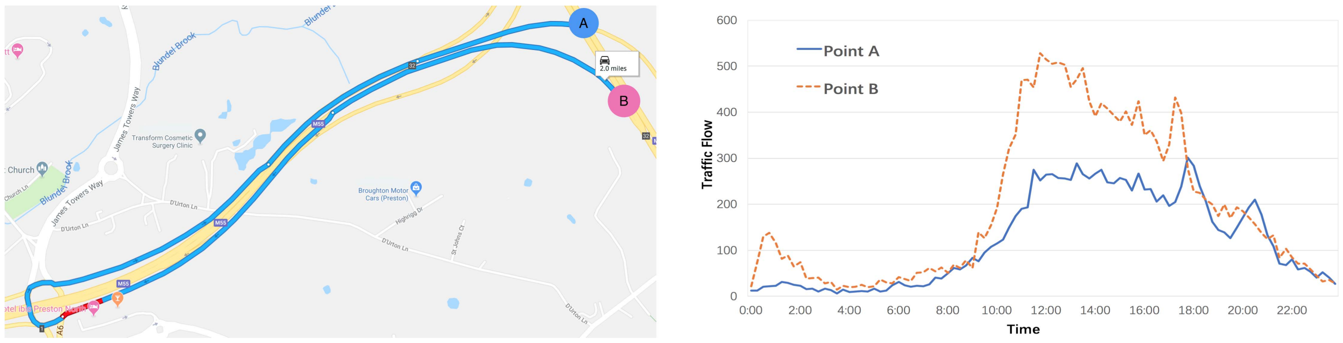


Fig. 1. Distance dependency is not symmetrical between monitoring stations.

manually setting the threshold during pre-processing, which may cause insufficient spatio-temporal correlation between the sensor and its distant neighbors in multiple time steps.

To address this problem, this paper proposes GTA, a graph-based temporal attention framework considering both spatial and temporal correlation, to forecast traffic flow based on multiple sensors. More specifically, GTA can better capture spatial dependencies leveraging graph embedding technology on sensor networks because it preserves more details in the algorithms. We also introduce an attention mechanism to adaptively identify the relations among temporal submodules. Spatio-temporal dependencies are more effectively and comprehensively integrated as we take full advantage of the topological properties of transportation networks.

Our proposed framework consists of the following components: *data preprocessing, temporal module, graph embedding, attention mechanism, and fusion*. First, the collected traffic data is normalized to [0, 1] during preprocessing. Next, the preprocessed data is divided into three portions (monthly pattern, weekly pattern, and current pattern) and fed into the temporal module to forecast affine transformation matrices of different patterns. Then, the spatial dependency is extracted by leveraging graph embedding technology on the topology of sensor networks depicted as road network distance. Next, an attention mechanism is constructed to assign different weights to the temporal submodules by using spatial embedding feature and current pattern as input. Finally, the topology is also fed into a fully connected layer to build the latent matrix, which is exploited to more comprehensively integrate spatio-temporal dependencies.

We evaluate GTA with a large-scale traffic dataset from multiple England cities. Because the data do not contain road topology information, we augment the data with road network distances between two monitoring stations, which are collected from Google services. Evaluation results show that GTA outperforms several state-of-the art prediction methods. We also perform a comparison between the variants of our proposed model and sensitivity analysis of the parameters.

Contributions. This paper makes the following main contributions:

- We propose a new deep learning framework GTA, which further incorporates the attention mechanism to adaptively identify the relations among temporal submodules, in addition to applying graph embedding on multiple sensor data. GTA also introduces a more effective and comprehensive strategy to integrate spatio-temporal dependencies, which can take full advantage of the topological properties of transportation networks.
- We conduct extensive experiments based on a large real-world traffic dataset from England, while enhancing it with road topology information. The results show that our method can yield better performance over the state-of-the-art baseline methods.
- We release our enhanced traffic dataset with road network topology information[1] to the research community under the open government license v.3.0.[2]

The remainder of this paper is organized as follows. Section II-A presents a literature review on traffic flow forecasting. Section III gives some relevant definitions used in this paper. Section IV introduces our model GTA. Section V presents and discusses the experimental results. Finally, we conclude with Section VI.

II. Related Work

A. Traffic Flow Prediction

Traffic flow forecasting has been investigated for multiple decades [36]. A significant number of prediction models have been developed to facilitate traffic control and management. Vlahogianni *et al.* [37] systematically reviewed existing methods including both computational intelligence and traditional statistical models.

1) Lacking Consideration of Spatial Dependencies: Some of these studies only took temporal dependencies into account for traffic flow forecasting. For instance, Box *et al.* [38] built an autoregressive moving average (ARMA) model, which plays a fundamental role in the area of forecasting. Taking ARMA as a basis, ARIMA [39], an extended version of ARMA, was developed for traffic flow forecasting. It requires stationary differential data and is not suitable for the highly non-linear problem due to the use of linear architecture. Later, a bunch of variations were developed, including seasonal ARIMA (SARIMA) [40] designed for capturing the common periodical features from time-series processes and Kohonen

[1]https://github.com/skzhangPKU/GTA

[2]https://www.nationalarchives.gov.uk/doc/open-government-licence/version/3/

ARIMA (KARIMA) [41] constructed for short-term traffic flow forecasting based on maps. Moreover, several studies focused on time series forecasting under missing data. For instance, Chen and Sun [42] proposed a low-rank autoregressive tensor completion (LATC-TNN) model for traffic forecasting, which can effectively integrate local temporal patterns and global trends.

To further mine the temporal correlation, some machine learning methods based on probabilistic perspectives were exploited to predict traffic flow. For instance, Qi and Ishak [43] employed the Markov chain to tackle the dynamic changes in traffic situations using state transition probabilities. Hong [44] developed a hybrid model, called SSVRCSA, that combines the seasonal SVR model with chaotic simulated annealing algorithm for inter-urban traffic volume prediction. However, they are under a naive assumption that traffic flow at one prediction interval depends only on that of a few previous intervals, which does not take periodicity into account.

Furthermore, deep learning technology, as a branch of machine learning, was also explored to mine spatio-temporal dependencies for short-term traffic flow forecasting. Several researchers attempted to utilize common deep learning models (MLP [17], LSTM [18], SAE [21], RBM [45], etc.) to forecast traffic flow, which can achieve better predictive performance by mining the temporal correlation of traffic data. However, these approaches typically ignore the spatial dependencies of traffic data.

2) Modeling Spatial Dependencies Using CNN: CNN has been widely applied in a variety of fields. For instance, Wang *et al.* [46] developed a siamesed fully convolutional network, with the embedded feature of structured contour and location prior as input, for road detection. Moreover, a joint convolutional neural network [47] that integrates context transfer was proposed to label street scenes, which significantly outperformed other methods. Inspired by this, several CNN-based methods are proposed to model the spatial dependency by learning traffic as images, which are only applicable to grid-map-based data rather than multi-sensor-based data. Ma *et al.* [19] applied a deep learning architecture of CNN to images converted from network traffic in anticipation of extracting spatio-temporal relation to predict large-scale, network-wide traffic speed.

Moreover, a series of models that integrate CNN and LSTM [45], [48]–[51] were constructed to forecast short-term traffic flow. For instance, ConvLSTM [24] based on critical road sections was built by using traffic speed as the input to forecast the traffic evolution of global networks. Yao *et al.* [25] put forward a Deep Multi-View Spatial-Temporal Network (DMVST-Net) architecture that consists of semantic view, spatial view, and temporal view for taxi demand prediction. Wu and Tan [26] developed a deep framework, named CLTFP, to forecast future short-term traffic volume. More specifically, CLTFP captured spatial correlation using 1-dimension CNN and temporal correlation including variability and periodicity using LSTMs. However, these approaches are either inadequate or unreasonable in terms of spatial dependency.

3) Modeling Spatial Dependencies Using Graph Convolution: Currently, there are a few studies incorporating road topology to forecast traffic flow [27]–[35]. Most of them adopt graph convolution methods to capture spatial dependencies. Li *et al.* [27] proposed a diffusion convolutional recurrent neural network (DCRNN) that integrated the scheduled sampling technique and the encoder-decoder architecture to mine spatial-temporal dependencies from historical patterns for traffic flow forecasting. Similarly, STGCN [35] and ASTGCN [28] were developed to model the dynamics of spatial-temporal correlations by using several individual components. In these methods, they did not consider localized spatial-temporal dependencies. To address this issue, Song *et al.* [29] proposed spatial-temporal synchronous graph convolutional networks (STSGCN), which consist of several modules designed for modeling the heterogeneities in spatial-temporal graphs.

Several attempts have also been made to incorporate the attention mechanism in graph convolution. For instance, Zhang *et al.* [32] proposed the Gated Attention Networks (GaAN) to conduct traffic speed forecasting. The GaAN can adaptively identify the importance of attention heads using a convolutional sub-network. Zheng *et al.* [34] constructed a graph multi-attention network (GMAN), which consists of various types of attention modules such as spatial attention, transform attention, and temporal attention, to model complex spatio-temporal dependencies.

Moreover, Zhao *et al.* [33] built a temporal graph convolutional network (T-GCN), which was employed to forecast traffic flow on urban road networks. Wang *et al.* [31] developed a spatial-temporal graph neural network (STGNN) model that integrates transformer layers and graph neural network layers for modeling series data. Wu *et al.* [52] proposed a novel architecture (Graph WaveNet) to model spatial correlations by developing an adaptive dependency matrix. Cui *et al.* [53] proposed a graph Markov network (GMN) and its variant that incorporates spectral graph convolution (SGMN) for spatial-temporal data forecasting with missing values. However, all of the above approaches rely on the adjacency matrix, which is extracted from the pairwise road network distances by manually setting the threshold. This may cause insufficient spatial-temporal dependencies between the sensor and its distant neighbors in multiple time steps.

In addition to the mainstream approaches mentioned above, Chen *et al.* [54] proposed to leverage the SVD-combined tensor decomposition (STD) to capture spatial-temporal traffic speed patterns from partially observed samples. Cui *et al.* [55] developed a graph wavelet gated recurrent neural network (GWGR) to forecast network-scale traffic speed. More specifically, GWGR captures spatial dependencies by incorporating the graph wavelet and temporal dependencies by leveraging a gated recurrent structure. However, they ignored the periodicity of traffic patterns in the modeling process.

B. The Attention Mechanism

The attention mechanism has been widely applied in many fields, such as natural language processing, image recognition, and speech recognition. For instance, Im and Cho [56] proposed a distance-based self-attention model, which captures the local dependency by using a simple distance mask, to understand sentences. Zhang *et al.* [57] designed a context-aware dual-attention network (CADAN) that integrates sentences and images for natural language inference. Wang *et al.* [58] developed a multi-task attention network to detect lane markings, which combines handcrafted features and deep features. Moreover, a semi-supervised learning algorithm [59], incorporating the attention mechanism and bias-variance decomposition, was built to recognize the synthetic aperture radar image. An attention recurrent convolutional network (ARCNet) [60] was proposed for remote sensing scene classification. Salazar *et al.* [61] developed a deep-learning-based framework that involves a fully self-attentional network for connectionist temporal classification. The main reason is its capability to identify local importance.

C. Graph Embedding Technology

In reality, since there exists plenty of graph-structured data, which contains latent characteristic patterns related to specific problems, how to extract and mine these patterns is a critical task. Existing classical graph embedding methods including DeepWalk [62], Node2Vec [63], SDNE [64], and LINE [65], have been applied to many systems. More specifically, Vallam *et al.* [66] proposed an effective framework that leverages the graph embedding technique to aggregate a set of incomplete ranked lists. A semi-supervised graph embedding approach (SemiGraph) [67] was built to predict dynamic links by incorporating the temporal and cross-sectional network structures. Besides, Yao *et al.* [25] designed a hybrid deep learning framework based on multiple views, where semantic views are modeled by graph embedding. One potential reason is to retain the details of the graph-structured data.

III. Preliminaries

In this section, we present the definitions of several key concepts and formulate the traffic flow forecasting problem as follows.

Definition 1 (Sensor Network): The sensor network composed of several monitoring stations is depicted as a weighted directed graph, which takes sensors as vertices and the road network distances as edge weights. We denote the directed graph as $\mathcal{G} = (\mathcal{S}, \mathcal{E}, \mathbf{D})$, in which $\mathcal{S} = \{s_i | 1 \le i \le |\mathcal{S}|\}$ is the collection of stations $|\mathcal{S}| = L$, $\mathcal{E} = \{(u, v) | u \in \mathcal{S}, v \in \mathcal{S}\}$ is the collection of edges that indicates vertex connectivity and reachability, and $\mathbf{D}_{ij}$ is the element of the weighted adjacency matrix that stands for the proximity from vertex i to vertex j.

Definition 2 (Traffic Flow): The traffic flow is defined as the number of vehicles passing through the monitoring station over a certain time interval. For instance, given the time interval Δt, $y_{t+1}^{s_i}$ indicates the traffic flow of the monitoring station s_i over the time horizon $[t, t + \Delta t)$, where t denotes the starting point of the time horizon.

Definition 3 (The Traffic Flow Forecasting Problem): The purpose of the traffic flow forecasting problem is to predict traffic volume at the $(t+1)^{th}$ time interval $y_{t+1}^{s_i}$, given the historical observation sequence $\mathcal{H}^{s_i} = \left\{ y_j^{s_i} | s_i \in \mathcal{S}, j \in \left\{ t - T' + 1, t - T' + 2, \cdots, t \right\} \right\}$, where $\mathcal{S}$ is the collection of all monitoring stations and T' is the length of historical traffic sequence. Our work aims to build a function model $\mathcal{F}(\cdot)$ that can mine the spatio-temporal correlation from historical traffic patterns to forecast future traffic flow accurately. Given the sensor network depicted as a weighted directed graph $\mathcal{G}$, we conduct multi-position global forecasting considering the interaction of monitoring stations. Historical observations of all monitoring stations are denoted as $\mathcal{H}^L = \mathcal{H}^{s_1, s_2, \cdots, s_L} = (\mathcal{H}^{s_1}, \mathcal{H}^{s_2}, \cdots, \mathcal{H}^{s_L})^{\mathrm{T}} \in \mathbb{R}^{L \times T'}$, where L is the number of monitoring stations. Therefore, the predicted traffic flow $y_{t+1}^L = y_{t+1}^{s_1, s_2, \cdots, s_L} = \left(y_{t+1}^{s_1}, y_{t+1}^{s_2}, \cdots, y_{t+1}^{s_L} \right)^{\mathrm{T}} \in \mathbb{R}^{L \times 1}$ is

$$y_{t+1}^L = \mathcal{F}\left(\mathcal{H}^L, \mathcal{G}; \Theta \right) \tag{1}$$

where Θ are all learnable parameters in the model.

IV. The GTA Framework

Our main research objective is to extract the spatio-temporal correlation from the historical traffic patterns to make accurate traffic flow prediction at different time intervals. Generally, traffic flow forecasting can be divided into long-term (over 60 minutes) and short-term (0∼60 minutes) prediction according to the time interval, and the latter is of greater practical significance in the application of ITS. We followed previous work [68]–[71] and considered the generalization performance of the model before conducting 15-min, 30-min, and 60-min traffic flow forecasting. In addition, most individuals will adopt a distance-prior strategy to drive in cities unless for vehicle accidents and emergency traffic control, resulting in the spatial (distance) correlation based on the road topology.

As the current deep models are less suitable for high-dimensional sparse data, it is desirable to embed the sparse representation of the spatial structure into a lower-dimensional space. Indeed, some graph embedding methods have been proposed and applied to various fields successfully [66], [67]. It is shown that the embedded representation not only preserves the intrinsic properties of spatial structure but also incorporates context. Inspired by these, this work attempts to investigate the effectiveness of graph embedding methods on spatial dependencies modeling.

Furthermore, considering the repetitive nature of human behavior, the temporal dependency of traffic flow is summarized into three types (monthly periodicity, weekly periodicity, and variability), which are the foundation of our sequence modeling [25], [26], [30], [72]. Currently, some work [30] only applied a simple combination of these temporal properties, and could not adequately identify the relations among them for multi-sensor traffic forecasting. Nevertheless, the attention mechanism may solve this issue because of its ability to determine the relative importance of different components. Accordingly, we consider using an attention mechanism to model the relations among temporal properties.

Therefore, we propose a graph-based temporal attention framework GTA, which incorporates the temporal and spatial correlation, for multi-sensor global traffic forecasting based on historical traffic patterns. Figure 2 presents the GTA framework, which is mainly composed of five functional components: data preprocessing, temporal module, graph embedding, attention mechanism, and fusion.

For the *preprocessing* module, we leverage max-min normalization technology to alleviate the problem of different scales of traffic data collected from monitoring stations and stacked autoencoders to perform high-level feature extraction due to the effects of exogenous variables (vehicle accidents, emergency traffic control, and extreme weather events).

The *temporal components* consist of several LSTMs, which are constructed based on multiple views (monthly pattern, weekly pattern, and current pattern). Moreover, we employ *an attention mechanism*, with spatial embedding feature and current pattern as input, assigning different weights to the temporal submodules. Each submodule forecasts an affine transformation matrix corresponding to temporal patterns, and then the weighted *fusion* of these matrices is carried out.

The *graph embedding* technique can find the latent vector representation from the topology of the sensor network, which is exploited to capture the spatial dependency from road network distance. Furthermore, a fully connected layer is exploited to mine the latent spatial matrix for traffic flow prediction.

Finally, the hybrid model is built to capture the spatio-temporal dependency using the resulting transformation matrix and current pattern as input. The prediction performance of the model is evaluated and compared by various error metrics.

A. Graph-Based Spatial Dependency Modeling

Given a sensor network $\mathcal{G} = (\mathcal{S}, \mathcal{E}, \mathbf{D})$, where $|\mathcal{S}| = L$ is the number of stations, $\mathcal{E}$ is the collection of edges and $\mathbf{D} \in \mathbb{R}^{L\times L}$ is the weight adjacency matrix depicted with road network distance. It means that our work aims to simultaneously predict the traffic volume of L positions y_{t+1}^{L} with historical traffic flow data $\mathcal{H}^L$ (denoted as $\mathcal{H}$ for simplicity) as input.

The periodicity of traffic patterns, such as the travel habits of commuters, holds only under an assumption, where the topology of the sensor network keeps unchanged in a short period. On the basis of this assumption, the distribution of sensors deployed on the urban traffic network can reveal the traffic pattern, which is fundamental to traffic flow forecasting. Road network distance, rather than Euclidean distance, can better reflect the correlation between monitoring stations. More specifically, with a closer Euclidean distance but a further driving distance, there are quite different changes in the magnitude of traffic flow between two stations in a freeway network. Therefore, it is more suitable to model the spatial correlation using road network distance.

For the given directed graph $\mathcal{G}$, the weight adjacency matrix, depicted with the driving distance, is denoted as

$$\mathbf{D} = \begin{bmatrix} d_{s_1s_1} & d_{s_1s_2} & \cdots & d_{s_1s_n} \\ d_{s_2s_1} & d_{s_2s_2} & \cdots & d_{s_2s_n} \\ \vdots & \vdots & & \vdots \\ d_{s_ns_1} & d_{s_ns_2} & \cdots & d_{s_ns_n} \end{bmatrix}, \tag{2}$$

where s_i denotes the monitoring station i, and $d_{s_is_j}$ is the driving distance from station i to station j.

Next, we will introduce how the topology structure of the directed graph is embedded into a low-dimensional representation to preserve vertex similarity. Specifically, LINE [65] is used to learn the embedding representations of the graph. A brief description of the graph embedding method is presented below. Its objective function is

$$O = -\sum_{(s_i,s_j)\in\mathcal{E}} d_{s_is_j} \log p\left(s_j|s_i\right), \tag{3}$$

where $p\left(s_j|s_i\right)$ is a conditional distribution that stands for the probability of station s_j under the given station s_i. The distribution is defined as follows:

$$p\left(s_j|s_i\right) = \frac{\exp\left(\vec{c}_{s_j}^{\mathrm{T}} \cdot \vec{u}_{s_i}\right)}{\sum_{k=1}^{L} \exp\left(\vec{c}_{s_k}^{\mathrm{T}} \cdot \vec{u}_{s_i}\right)}, \tag{4}$$

where L is the number of monitoring stations, and $\vec{c}_{s_i}$ and $\vec{u}_{s_i}$ are the context vector representation and vertex vector representation of s_i, respectively. By optimizing the objective function O, we extract the low-dimensional vector representation of all nodes $\mathbf{GE} \in \mathbb{R}^{L\times B}$ $(B < L)$ from the topology of sensor networks depicted with the driving distance $\mathbf{D} \in \mathbb{R}^{L\times L}$ using graph embedding technology. Here, B indicates the embedding size, which is set to 128 in the present work. Specifically, Eq. 3 is optimized with stochastic gradient descent using negative sampling [73] and edge sampling [65].

Moreover, the weight adjacency matrix is fed into a dense layer to generate the latent matrix, which is exploited to integrate the spatial correlation and temporal correlation directly. The procedure is described as follows:

$$\mathbf{LM}^d = g\left(W_{gl} * \mathbf{D} + b_{gl}\right), \tag{5}$$

where g is a linear activation function, W_{gl} is an embedding matrix, b_{gl} is a zero vector, and $*$ denotes a matrix multiplication operation.

B. Attention-Based Temporal Dependency Modeling

Considering the characteristics of the traffic data, we summarize the temporal dependency into three categories: monthly pattern, weekly pattern, and current pattern. The current pattern refers to the correlation between several recent time intervals and the target one, e.g., the traffic conditions at 07:30 am will affect those at 08:00 am. The weekly and monthly patterns refer to the repetitive nature of human activities. For instance, weekdays exhibit similar variation trends of traffic flow, including distinct morning and evening rush hour periods. Furthermore, during weekends, the morning rush hour periods may be delayed due to late wake-ups.

Accordingly, the input of the temporal module is divided into three portions:

$$\mathcal{H}^c = \begin{bmatrix} h_{s_1}(t^c - T_c' + 1) & h_{s_1}(t^c - T_c' + 2) & \cdots & h_{s_1}(t^c) \\ h_{s_2}(t^c - T_c' + 1) & h_{s_2}(t^c - T_c' + 2) & \cdots & h_{s_2}(t^c) \\ \vdots & \vdots & & \vdots \\ h_{s_L}(t^c - T_c' + 1) & h_{s_L}(t^c - T_c' + 2) & \cdots & h_{s_L}(t^c) \end{bmatrix}, \tag{6}$$

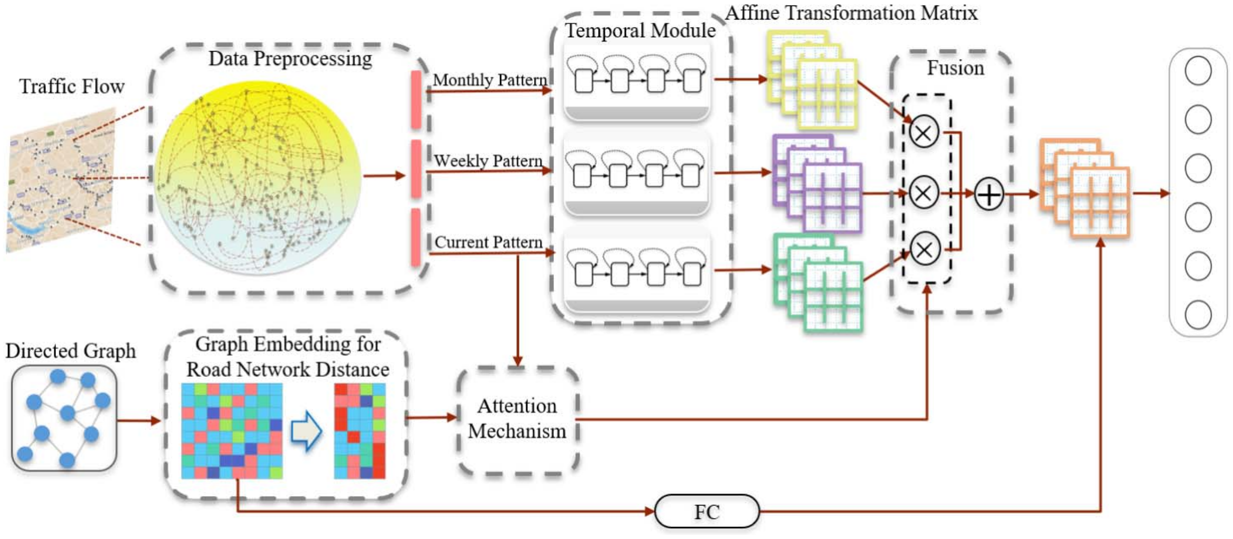


Fig. 2. The overall architecture of the GTA framework.

$$\mathcal{H}^{w}=\begin{bmatrix} h_{s_1}(t^{w}-T'_{w}+1) & h_{s_1}(t^{w}-T'_{w}+2) & \cdots & h_{s_1}(t^{w}) \\ h_{s_2}(t^{w}-T'_{w}+1) & h_{s_2}(t^{w}-T'_{w}+2) & \cdots & h_{s_2}(t^{w}) \\ \vdots & \vdots & & \vdots \\ h_{s_L}(t^{w}-T'_{w}+1) & h_{s_L}(t^{w}-T'_{w}+2) & \cdots & h_{s_L}(t^{w}) \end{bmatrix}, \tag{7}$$

$$\mathcal{H}^{m}=\begin{bmatrix} h_{s_1}(t^{m}-T'_{m}+1) & h_{s_1}(t^{m}-T'_{m}+2) & \cdots & h_{s_1}(t^{m}) \\ h_{s_2}(t^{m}-T'_{m}+1) & h_{s_2}(t^{m}-T'_{m}+2) & \cdots & h_{s_2}(t^{m}) \\ \vdots & \vdots & & \vdots \\ h_{s_L}(t^{m}-T'_{m}+1) & h_{s_L}(t^{m}-T'_{m}+2) & \cdots & h_{s_L}(t^{m}) \end{bmatrix}, \tag{8}$$

where t^{w} and t^{m} stand for the same moment of t^{c} in last week and last month separately, T'_{c}, T'_{m}, and T'_{w} stand for the number of time intervals for variability, monthly periodicity, and weekly periodicity respectively, and $h_{s_*}(t)$ indicates the historical traffic volume of the station s_* at the t^{th} time interval.

Moreover, we propose to employ LSTM to extract the temporal correlation from historical observations and forecast the affine transformation matrix of the corresponding pattern. Unlike RNN, the LSTM alleviates the problem of gradient vanish and explosion, which is capable of learning long-term dependencies [20], [74], [75]. Taking the monthly periodicity as an example, we describe in detail the realization process of the temporal submodule. The same procedure is also applicable to weekly and current patterns. The input of the submodule is denoted as $\mathrm{I}^{m}=\left(\mathrm{I}^{m}_{1},\mathrm{I}^{m}_{2},\cdots,\mathrm{I}^{m}_{T'_{m}}\right)$, where $\mathrm{I}^{m}_{p}=\mathcal{H}^{m}_{p}=[h_{s_1}(t^{m}-T'_{m}+p),h_{s_2}(t^{m}-T'_{m}+p),\cdots,h_{s_L}(t^{m}-T'_{m}+p)]^{\mathrm{T}}$. The hidden layer output at the time step t is denoted as y^{m}_{t}. The sequential representation of the traffic state can be calculated iteratively by the following equations:

$$\mathrm{i}^{m}_{t}=\sigma\left(W^{m}_{i}\cdot[\mathrm{y}^{m}_{t-1},\mathrm{I}^{m}_{t}]+\mathrm{b}^{m}_{i}\right), \tag{9}$$

$$\mathrm{f}^{m}_{t}=\sigma\left(W^{m}_{f}\cdot[\mathrm{y}^{m}_{t-1},\mathrm{I}^{m}_{t}]+\mathrm{b}^{m}_{f}\right), \tag{10}$$

$$\mathrm{o}^{m}_{t}=\sigma\left(W^{m}_{o}\cdot[\mathrm{y}^{m}_{t-1},\mathrm{I}^{m}_{t}]+\mathrm{b}^{m}_{o}\right), \tag{11}$$

$$\tilde{C}^{m}_{t}=tanh\left(W^{m}_{C}\cdot[\mathrm{y}^{m}_{t-1},\mathrm{I}^{m}_{t}]+\mathrm{b}^{m}_{C}\right), \tag{12}$$

$$C^{m}_{t}=\mathrm{f}^{m}_{t}\odot C^{m}_{t-1}+\mathrm{i}^{m}_{t}\odot\tilde{C}^{m}_{t}, \tag{13}$$

$$\mathrm{y}^{m}_{t}=\mathrm{o}^{m}_{t}\odot tanh(C^{m}_{t}), \tag{14}$$

where σ is the sigmoid activation function, $\cdot$ and $\odot$ stand for the matrix production and scalar product of two vectors respectively, and W^{m}_{*} and b^{m}_{*} are both learnable parameters in the model. $tanh$ is the hyperbolic tangent function, which contributes to the alleviation of the problem of gradient vanish due to its good gradient flow feature. Similarly, the hidden layer outputs for the weekly pattern y^{w}_{t} and current pattern y^{c}_{t} can be obtained.

Then, a fully connected layer is employed to forecast the affine transformation matrix using the hidden representation as input, which is formulated as follows:

$$\mathbf{AT}^{c}=reshape\left(f\left(W_{fc}\mathrm{y}^{c}_{t}+b_{fc}\right)\right), \tag{15}$$

$$\mathbf{AT}^{w}=reshape\left(f\left(W_{fw}\mathrm{y}^{w}_{t}+b_{fw}\right)\right), \tag{16}$$

$$\mathbf{AT}^{m}=reshape\left(f\left(W_{fm}\mathrm{y}^{m}_{t}+b_{fm}\right)\right), \tag{17}$$

where W_{fc}, W_{fw}, and W_{fm} are weight matrices, b_{fc}, b_{fw}, and b_{fm} are bias vectors, f is the activation function, and $\mathbf{AT}^{*}$ is the affine transformation matrix. In this work, the $reshape$ operation returns a $L\times L$ matrix by stacking column-wise the elements of a $L*L$ vector.

In addition, an attention mechanism, which can incorporate the temporal dependency and spatial dependency indirectly, is employed to yield weights of affine transformation matrices with the embedding representation of the directed graph and traffic flow of the current pattern as input. The reasons for using the embedding representation instead of the weighted adjacency matrix are as follows: (1) There is a large amount of redundancy caused by $d_{s_i s_j}=d_{s_j s_i}$ in the asymmetric matrix $\mathbf{D}$. (2) The scalability of the system is improved. More specifically, it is almost impossible to directly take the adjacency matrix as a feature space for input when the number of sensors is large. (3) The embedded vector, with simplicity and convenience, not only preserves the intrinsic properties of spatial structure but also incorporates context.

Next, the process of the attention mechanism will be explained in detail. The unnormalized relevance score of the k-th temporal submodule can be formulated as:

$$us^k = \tanh\left(W_{am}\left[\mathcal{H}^c_{T'_c} \| \mathbf{GE}^{\mathrm{T}}\right] + b_{am}\right), \tag{18}$$

where tanh is a hyperbolic tangent, $\|$ is a concatenation operator, $\mathcal{H}^c_{T'_c}$ denotes the input of the current pattern for all monitoring stations at time t, and W_{am} and b_{am} are the weight matrix and bias vector, respectively. Furthermore, the weights of affine transformation matrices, which can be calculated by normalizing the relevance score, is described as follows:

$$\beta^k = \frac{\exp^{us^k}}{\sum_k \exp^{us^k}}, \text{for } k = 1, \cdots, K_{ts}, \tag{19}$$

where $\sum_k \beta^k = 1$, K_{ts} is the number of temporal submodules, and β^k is the weight of the k-th temporal submodule. In this work, K_{ts} equals 3.

C. Prediction

This subsection describes the last step in the framework, which involves incorporating the spatio-temporal correlation and the formation of predicted traffic flow. Given the sensor network, the framework models the temporal dependency based on multiple views (monthly pattern, weekly pattern, and current pattern), and spatial dependency using graph embedding technology to forecast future traffic flow. For simplicity, we define a pattern array $pa = [c, w, m]$, which includes the current pattern, weekly pattern, and monthly pattern. A weighted sum of affine transformation matrices of different patterns is formulated as follows:

$$\mathbf{WM}_t = \sum_{k}^{K_{ts}} \beta^k \mathbf{AT}^{pa[k]}, \tag{20}$$

Algorithm 1: Training Procedure of GTA

```
Input: Sensor network $\mathcal{G} = (\mathcal{S}, \mathcal{E}, \mathbf{D})$;
    Historical traffic data of all monitoring stations $\mathcal{H}$;
    All hyperparameters.
Output: Learned GTA model;
1 Employ graph embedding technology on $\mathcal{G}$ to extract
  node vector representation $\mathbf{LM}^d$;
2 for $k \leftarrow 1$ to $t$ do
3     According to section IV.A, the historical temporal
      pattern is divided into three portions: $\mathcal{H}^{c,k}$, $\mathcal{H}^{w,k}$, and
      $\mathcal{H}^{m,k}$;
4     Append $\langle\{\mathcal{H}^{c,k}, \mathcal{H}^{w,k}, \mathcal{H}^{m,k}\}, \widehat{y}_{k+1}\rangle$ to $\mathcal{X}$;
5 Initialize all learnable parameters $\theta$ in GTA;
6 repeat
7     Randomly select a batch of training sample $\mathcal{X}_{bt}$ form
      $\mathcal{X}$;
8     Optimize $\theta$ by minimizing the objective function
      Eq.23 with $\mathcal{X}_{bt}$, $\mathbf{LM}^d$, and $\mathbf{D}$ as input.
9 until met model stop criteria
```

where β^k is the weight of the affine transformation matrix, K_{ts} is the number of affine transformation matrices, and $pa[k]$ is the k-th element in the pattern array pa. Besides, the resultant transformation matrix is built by using the latent spatial matrix as input, which can be formulated as follows:

$$\mathbf{RT}_t = \mathbf{LM}^d \circ \mathbf{WM}_t, \tag{21}$$

where $\circ$ denotes the element-wise product, and $\mathbf{LM}^d$ is the latent spatial matrix. Finally, based on the transformation matrix, the traffic flow at time $t+1$ can be calculated as follows:

$$y^L_{t+1} = \mathbf{RT}_t \cdot \mathcal{H}^c_{T'_c} + b_{rt}, \tag{22}$$

where $\cdot$ denotes the matrix production, b_{rt} is the learnable parameter, and $\mathcal{H}^c_{T'_c}$ stands for the input of the current pattern for all monitoring stations at time t.

Given the historical observations and sensor network, we need to minimize the following cost function for the hybrid model:

$$\mathcal{J}(\theta) = \sum_{s_k \in \mathcal{S}} \left\| y^{s_k}_{t+1} - \widehat{y}^{s_k}_{t+1} \right\|^2 + \lambda\Omega(W_*), \tag{23}$$

where θ are all parameters used in the model, $\mathcal{S}$ is the set of monitoring stations, W_* are learnable weight matrices, λ is a regularization coefficient of the penalty term Ω, and $y^{s_k}_{t+1}$ and $\widehat{y}^{s_k}_{t+1}$ are the predicted value and observed value of the monitoring station s_k at the time interval t respectively. In this work, ℓ_1 and ℓ_2 regularization penalties are attempted; the latter is chosen as per the experimental results. The regularization coefficient is determined by employing a grid search from a coarse-grained to a fine-grained manner.

Moreover, aside from square error, we also attempt to leverage absolute error as the loss term of the optimization object. The experimental results indicate that GTA can also outperform other baselines under all evaluation metrics while using the absolute error. Compared to square error, there is only a slight difference in performance. With a trade-off between performance and stability, we take square error as the loss function to optimize the model.

The training procedure of the GTA framework is summarized in Algorithm 1.

V. Experimental Results

A. Datasets

We evaluate GTA using real traffic data collected from highways in England. The traffic dataset includes average speed, traffic flow, time period, location, and date of 249 monitoring stations. The temporal granularity of the original dataset is 15 minutes, while it can be configured to 15 minutes, 30 minutes and 60 minutes, respectively, during the experiments. The sensors are located from site A414 between M1 J7 and A405, which covers several cities that include Manchester, Liverpool, and Blackburn. Figure 3 presents the distribution of monitoring stations used in the experiment. Specifically, we use a whole year of traffic data ranging from January 1st, 2014 to December 31st, 2014 for the experiments. The total number of data entries is 8,724,960, the mean value

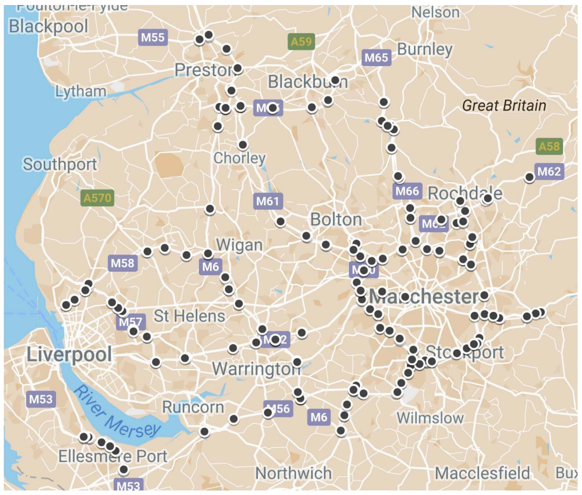


Fig. 3. The distribution of monitoring stations studied in this experiment.

of traffic volume is 466. Among the data, the last two months (11/01/2014-31/12/2014) are used as testing data, and the first 10 months (01/01/2014-10/31/2014) as training data.

We process the collected traffic data by normalizing them to [0, 1] before we feed them into the algorithm. Because the dataset does not contain road network distance between two monitoring stations, we enhance the data with these topology information based on distances collected from Google Services.[3]

B. Settings

1) Baselines: In order to evaluate the performance of our proposed model, we compare it with the following methods:

- **ESWP**: It is a model based on estimated, static weekly profiles (ESWP). The model takes the traffic of the same time period last week as the predicted value of the current time period.
- **PM**: The persistence model (PM) takes the observed value at time $t-1$ as the predicted value at time t.
- **VAR**: The vector autoregressive (VAR) for forecasting multivariate time series is widely used in statistics and econometrics, especially in sequence analysis. The model is implemented by the `statsmodels` python package.
- **MLP**: It has a multi-layer structure (one input layer, one or more hidden layers, and one output layer), which can approximate the complex nonlinear mapping with the assistance of activation functions. The hidden size is set to 200.
- **SAE** [21]: It is a deep neural network model through stacking autoencoders, which can learn the latent feature representation of traffic flow. The SAE includes three hidden layers, each of which contains 200 cells. The learning rates are set to $1e^{-3}$, $\frac{1}{3}e^{-3}$ and $1e^{-4}$ for these hidden layers, respectively. The batch size is 128.
- **LSTM** [22]: It is a variant of the RNN model with wide applications in machine translation, image annotation, and speech recognition due to its ability of mining temporal dependency. The hidden layer in LSTM holds 200 memory cells. The number of LSTM layers is 2. The L1 and L2 weight decays are $2e^{-4}$ and $2e^{-5}$, respectively.
- **T-GCN** [33]: It leverages the graph convolutional network (GCN) and gated recurrent unit (GRU) to model spatial and temporal dependencies, respectively.
- **STGCN** [35]: It extracts temporal dependencies using gated CNNs and spatial features using ChebNet.
- **ASTGCN** [28]: ASTGCN is composed of several individual components that integrate graph convolution and attention mechanisms.
- **STSGCN** [29]: It is designed for modeling the heterogeneities in spatial-temporal graphs.
- **GMAN** [34]: It consists of various types of attention modules to model complex spatio-temporal dependencies.
- **ConvLSTM** [24]: The convolutional long short-term memory (ConvLSTM) model extracts the temporal correlation using LSTM and the spatial correlation using CNN from historical traffic patterns, and thus forecasts future traffic flow. The kernel size is set to 3. Both pool size and stride are set to 2. It contains five convolution layers, five pooling layers, two LSTM layers, and two fully connected layers.
- **DCRNN** [27]: It is one of the cutting edge deep learning models for forecasting, which uses a diffusion process during the training stage to learn the representations of spatial dependency. For DCRNN, the encoder and decoder both contain two recurrent layers. The parameter settings remain the same as recommend by the authors in https://github.com/liyaguang/DCRNN.
- **Graph-WaveNet** [52]: It assembles graph convolution with dilated casual convolution to learn spatial-temporal dependencies simultaneously.

2) Setup: In this study, the historical traffic data is processed on a PC (CPU: Intel (R) Core (TM) i5-4460 @ 3.20GHz, GPU: NVIDIA GeForce GTX 750 Ti, memory: 12GB). Moreover, with the support of the Python libraries that include Scikit-learn and TensorFlow, we build and implement the deep learning models on a Graphics Processing Unit (GPU) platform using Argon with Nvidia Tesla P100 Accelerator Cards with 16GB of GPU memory. The training data is randomly divided into training sets and validation sets with a ratio of 8:2. The architecture of the GTA framework consists of six LSTM layers (three two-layer LSTMs), one embedding layer, one attention layer, and one fully connected layer. Besides, there are some hyper-parameters settings of the model as follows. An Adam optimizer with an initial learning rate of 0.001 is used to update the model parameters. The regularization coefficient of the penalty term is 0.002. The mini-batch size is set to 128. The dropout rate is set to 20%.

3) Evaluation Metric: The predictive performance of the model is evaluated by Mean Absolute Error (MAE), Rooted Mean Square Error (RMSE), and Mean Absolute Percentage Error (MAPE), which are the most widely reported error measures used in regression problems. The evaluation metrics

[3]https://developers.google.com/maps/documentation/distance-matrix

TABLE I
PERFORMANCE COMPARISON BETWEEN GTA AND OTHER BASELINES FOR TRAFFIC FLOW FORECASTING

Method	15 min			30 min			60 min			Average Error		
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
ESWP	69.07	137.60	45.3%	158.07	291.90	45.4%	423.50	695.11	53.3%	216.88	374.87	48.0%
PM	36.48	63.35	17.2%	106.45	181.36	20.3%	363.93	587.42	37.9%	168.95	277.38	25.1%
VAR	25.49	41.71	18.7%	54.70	89.78	18.5%	145.39	229.80	22.9%	75.19	120.43	20.0%
MLP	26.99	44.60	22.5%	55.41	93.21	21.1%	120.33	199.77	20.1%	67.49	112.53	21.2%
SAE	25.51	42.18	20.9%	49.75	85.78	19.2%	117.60	206.77	19.7%	64.29	111.58	19.9%
LSTM	24.63	41.69	19.7%	51.28	90.36	18.2%	126.95	225.37	19.5%	67.62	119.14	19.1%
ConvLSTM	25.75	43.42	18.6%	55.71	96.41	17.3%	116.38	205.56	18.3%	65.95	115.13	18.1%
T-GCN	34.72	52.62	28.9%	71.89	112.27	26.8%	151.42	231.87	30.6%	86.01	132.25	28.8%
STGCN	26.44	44.15	15.9%	60.14	98.84	22.6%	133.26	214.79	22.2%	73.28	119.26	20.2%
ASTGCN	26.47	43.32	16.7%	59.38	95.68	19.2%	173.60	258.19	35.8%	86.48	132.40	23.9%
STSGCN	23.64	40.88	15.4%	55.56	96.45	23.7%	128.04	225.11	26.3%	69.08	120.81	21.8%
GMAN	25.04	44.88	16.9%	47.88	87.72	17.3%	100.02	178.09	19.5%	57.65	103.56	17.9%
DCRNN	22.61	39.21	15.5%	44.27	78.98	16.5%	97.86	171.51	18.9%	54.91	96.57	17.0%
Graph-WaveNet	23.53	40.42	15.3%	47.99	82.59	14.3%	109.14	184.65	16.6%	59.89	102.55	15.4%
GTA	**21.86**	**36.94**	**15.3%**	**42.92**	**75.02**	**13.6%**	**90.95**	**159.93**	**13.9%**	**51.91**	**90.63**	**14.3%**

can be formulated as follows:

$$MAE = \frac{1}{N|\mathcal{S}|}\sum_{i=1}^{N}\sum_{s_k\in\mathcal{S}}\left|y_i^{s_k} - \widehat{y}_i^{s_k}\right|, \tag{24}$$

$$MAPE = \frac{1}{N|\mathcal{S}|}\sum_{i=1}^{N}\sum_{s_k\in\mathcal{S}}\frac{\left|y_i^{s_k} - \widehat{y}_i^{s_k}\right|}{\widehat{y}_i^{s_k}}, \tag{25}$$

$$RMSE = \sqrt{\frac{1}{N|\mathcal{S}|}\sum_{i=1}^{N}\sum_{s_k\in\mathcal{S}}\left(y_i^{s_k} - \widehat{y}_i^{s_k}\right)^2}, \tag{26}$$

where N is the number of samples, $|\mathcal{S}|$ is the size of the collection of stations $\mathcal{S}$, and $y_i^{s_k}$ and $\widehat{y}_i^{s_k}$ are the predicted value and ground truth of the monitoring station s_k on the i-th sample, respectively.

C. Overall Results

In this section, we evaluate and compare the prediction performance between different models (ESWP, PM, VAR, MLP, SAE, LSTM, ConvLSTM, T-GCN, STGCN, ASTGCN, STSGCN, GMAN, DCRNN, Graph-WaveNet, and GTA). Table I presents the quantitative results of these models for 15-min, 30-min, and 60-min traffic flow forecasting in terms of MAE, RMSE, and MAPE.

The results show that GTA achieves the best forecast accuracy compared with other baseline methods. More specifically, in the order listed in Table I, the average MAPE values for the other baselines decrease 235.66%, 75.52%, 39.86%, 48.25%, 39.16%, 33.57%, 26.57%, 101.17%, 41.26%, 67.13%, 52.45%, 25.17%, 18.88%, and 7.7% relative to the GTA. The main reason is that VAR, MLP, SAE, and LSTM only take temporal dependencies into account, ConvLSTM is not suitable for non-Euclidean multisensor data, and graph-convolution-based methods (DCRNN, Graph-WaveNet, STGCN, etc.) extract insufficient spatial correlations due to the neglect of details in the algorithms. The finding demonstrates that GTA can better capture spatio-temporal dependencies based on sensor networks.

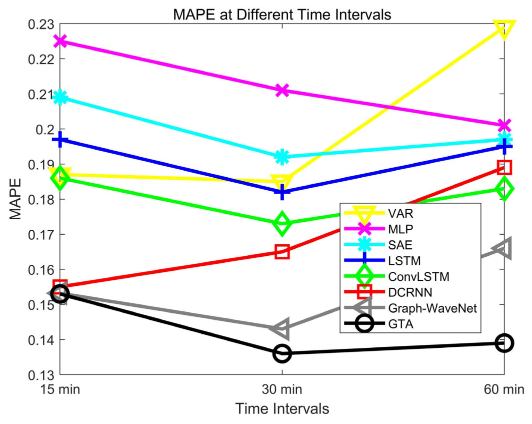


Fig. 4. The MAPE value of GTA and several selected methods on the 15-min, 30-min, and 60-min traffic flow prediction.

We can observe that GTA is robust in both evaluation metrics and time intervals. More specifically, Figure 4 shows that the slope of the black line is low, which reveals the stable performance of GTA over different time intervals. In the meantime, GTA exhibits better performance than other baselines under all evaluation metrics. The results show that GTA is insensitive to disturbances, which can better capture irregular traffic patterns.

Moreover, prediction models perform differently under different evaluation metrics. More specifically, for 60-min traffic flow forecasting, the MLP achieves a lower RMSE but a higher MAE value compared with the SAE. For ConvLSTM, it is superior to the common deep learning methods and classical time series method in terms of MAPE, but not in terms of MAE and RMSE. Even for DCRNN, it achieves a higher MAPE value compared with ConvLSTM. Therefore, it is rare that GTA can exhibit better performance under all evaluation metrics.

We also notice that common deep learning methods (MLP, SAE, and LSTM) are not significantly better than the

TABLE II
PERFORMANCE COMPARISON BETWEEN GTA AND ITS VARIANTS FOR TRAFFIC FLOW FORECASTING

Task	Metric	T	T+F	G+A+T	G+T+F	A+T+F	**GTA**
15 minutes	MAE	26.23	24.75	26.21	22.02	22.36	**21.86**
	RMSE	44.32	41.32	43.57	37.85	37.21	**36.94**
	MAPE	19.3%	18.7%	19.4%	17.8%	17.4%	**15.3%**
30 minutes	MAE	50.94	47.98	50.59	44.17	51.05	**42.92**
	RMSE	86.23	83.33	85.83	76.88	87.37	**75.02**
	MAPE	17.7%	16.2%	17.3%	16.3%	17.5%	**13.6%**
60 minutes	MAE	107.44	105.88	104.02	95.45	104.32	**90.95**
	RMSE	185.92	186.54	182.17	170.04	183.81	**159.93**
	MAPE	15.9%	15.1%	15.4%	15.1%	15.7%	**13.9%**
Average Error	MAE	61.54	59.54	60.27	53.88	59.24	**51.91**
	RMSE	105.49	103.73	103.87	94.92	102.80	**90.63**
	MAPE	17.6%	16.7%	17.4%	16.4%	16.9%	**14.3%**

traditional statistical model. For instance, the RMSE value of MLP for 15-min traffic flow forecasting is 44.60, which is slightly larger compared with VAR. However, for 60-min traffic flow prediction, its RMSE value is significantly lower than that of VAR. This reveals that GTA fails to yield poor performance like common deep models due to its reasonable structural design.

D. Additional Assessment

Apart from MAE, RMSE, and MAPE evaluation metrics, we also use the coefficient of determination (R^2) and median absolute error (MedAE) for evaluation. R^2 and MedAE are used to measure the correlation and error between observed and predicted values, respectively. The definitions are described as follows:

$$MedAE = median(|y_1 - \widehat{y}_1|, \cdots, |y_N - \widehat{y}_N|), \quad (27)$$

$$R^2 = 1 - \frac{\sum_{i=1}^{N}(y_i - \widehat{y}_i)^2}{\sum_{i=1}^{N}(\widehat{y}_i - \overline{y})^2}, \quad (28)$$

where $\overline{y} = \frac{1}{n}\sum_{i=1}^{N} y_i$, and y_i and $\widehat{y}_i$ are the predicted value and ground truth of the i-th sample, respectively. Due to the space limitation, we only report the results for 15-min traffic flow prediction (see Table III). The same conclusion holds for 30-min and 60-min traffic forecasting.

From Table III, we observe that GTA outperforms other competitive methods in terms of R^2 and MedAE. Specifically, the R^2 and MedAE values for GTA are 0.9893 and 13.34, respectively. DCRNN demonstrates the second-best performance while ESWP the worst performance, which is consistent with previous reports. Besides, Graph-WaveNet performs worse than STSGCN in terms of R^2 and MedAE. Combining the results in Table I, we find that STSGCN tends to yield some relatively large absolute errors. Overall, the results once again illustrate the effectiveness of our approach.

E. Different GTA Configurations

In order to investigate the influence of different components, we construct 5 simplified variants of GTA, which are described as follows. For simplicity, we use some notations, including graph embedding (G), attention mechanism (A), temporal module (T), and fully connected layer (F). Table II presents the prediction performance of GTA and its variants over different time horizons.

- **T**: It exploits the temporal module to forecast an affine transformation matrix. Then the traffic flow can be calculated by using the matrix and current traffic pattern as input.
- **T+F**: Taking T as a basis, we only employ a fully connected layer to generate the latent matrix, which is exploited to incorporate the temporal dependency and spatial dependency directly.
- **G+A+T**: Compared with GTA, it eliminates the fully connected layer for feature transformation of the weighted adjacency matrix, which means there is no direct spatiotemporal fusion.
- **G+T+F**: It uses graph embedding technology to extract the low-dimensional vector representation of all nodes and multiplies the representation with a learnable weight matrix $W^{B\times L}$ to form the input of building the resultant affine transformation matrix.
- **A+T+F**: It only takes the current traffic pattern as the input of attention mechanism to yield weights of affine transformation matrices, which means no graph embedding technology to incorporate the spatio-temporal correlation indirectly.

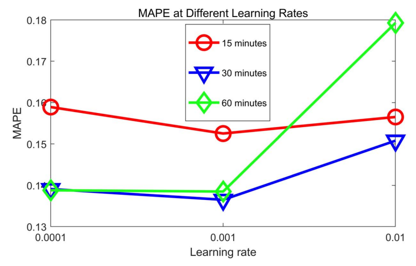
From Table II, we can see that GTA outperforms all its variants in terms of RMSE, MAE, and MAPE. More specifically, the average RMSE values for the other variants, in the order listed in Table II, decrease 16.40%, 14.45%, 14.61%, 4.73%, and 13.43% relative to the GTA. The results show that it is beneficial to improve the predictive performance of GTA by leveraging these components.

In addition, GTA performs better than G+T+F, which takes T+F as a basis and only employs graph embedding technology to yield the latent spatial matrix for spatio-temporal fusion. One explanation is that an attention mechanism can adaptively identify the relations among temporal submodules. More specifically, the prediction error of GTA under all evaluation metrics is lower than that of G+T+F, which demonstrates the validity of attention mechanism in traffic flow forecasting.

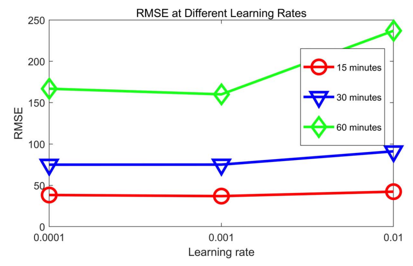
TABLE III

R^2 AND MEDAE OF GTA AND DIFFERENT MODELS FOR 15-min TRAFFIC FLOW FORECASTING. HERE, CL AND WAVENET REPRESENT CONVLSTM AND GRAPH-WAVENET, RESPECTIVELY. HIGHER R^2 AND LOWER MEDAE BOTH INDICATE BETTER PERFORMANCE

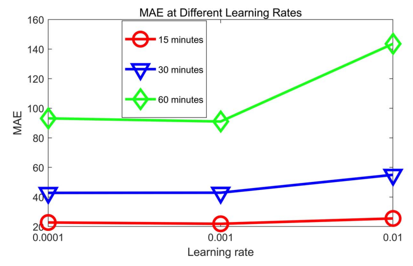
	ESWP	PM	VAR	MLP	SAE	LSTM	CL	T-GCN	STGCN	ASTGCN	STSGCN	GMAN	DCRNN	WaveNet	**GTA**
R^2	0.8494	0.9083	0.9874	0.9843	0.9859	0.9863	0.9784	0.9803	0.9853	0.9858	0.9878	0.9836	0.9889	0.9859	**0.9893**
MedAE	32.16	26.72	16.31	16.99	16.13	14.75	17.65	23.73	16.79	17.01	14.41	16.06	13.81	16.36	**13.34**



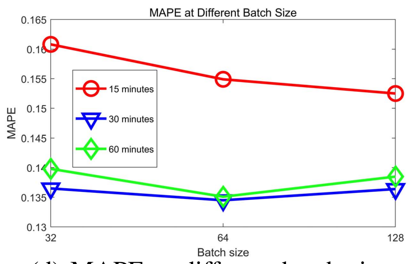
(a) MAPE at different learning rates



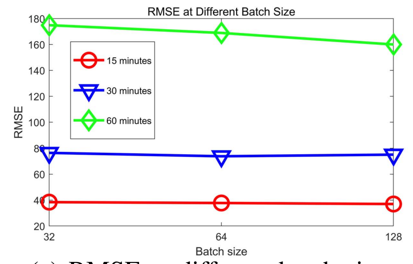
(b) RMSE at different learning rates



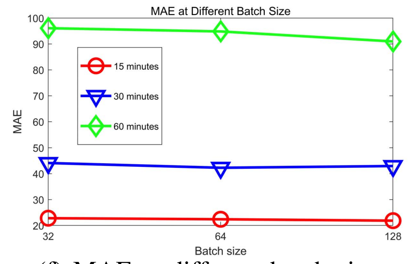
(c) MAE at different learning rates



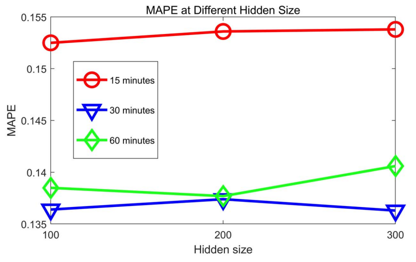
(d) MAPE at different batch size



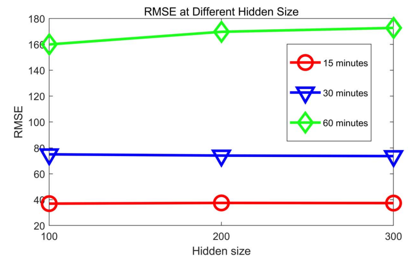
(e) RMSE at different batch size



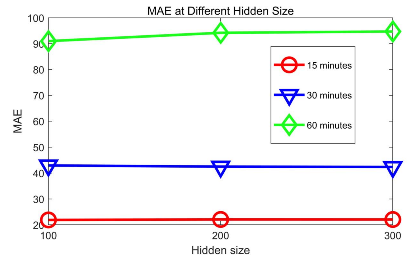
(f) MAE at different batch size



(g) MAPE at different hidden size



(h) RMSE at different hidden size



(i) MAE at different hidden size

Fig. 5. The MAPE, RMSE, and MAE value of the GTA model at different learning rates, batch size, and hidden size.

GTA exhibits better performance than A+T+F, which eliminates graph embedding technology and carries out a weighted sum of those temporal submodules based on attention mechanism. More particularly, for 60-min traffic flow forecasting, the RMSE value of GTA is dropped by 23.88 compared with A+T+F. The results imply the effectiveness of graph embedding technology in spatial dependency modeling.

Additionally, GTA outperforms both T+F and G+A+T on forecasting. One possible reason is that GTA makes full use of the topological properties of transportation networks. The results reveal the spatio-temporal dependencies can be more effectively and comprehensively integrated, resulting in lower prediction errors.

F. Sensitivity Analysis

In this section, we investigate the effect of parameters that include learning rate, batch size, and the number of hidden layer units on model performance. Figure 5 shows the prediction error of the model with respect to these parameters over various time horizons.

We can see that GTA achieves the best performance in all scenarios when the learning rate is set to 0.001. The best learning rate available by the experiment is consistent with that recommended by the optimizer algorithm. More specifically, for 60-min traffic flow prediction, the MAPE value of GTA of a 0.001 learning rate is 13.9% lower than that of a 0.01 learning rate, which indicates that there is a wide range of performance fluctuations as the learning rate changes. One possible reason is that inappropriate learning rate causes slow convergence or even failure.

For batch size, we can observe that its impact on model performance is minor relative to the learning rate. The MAE, RSME, and MAPE value of the model vary slightly but mainly remain stable as the batch size increases. More specifically, taking 30-min traffic flow forecasting as an example, the MAE value of GTA of a 64 batch size is 42.34, which is under 0.58 compared with that of a 128 batch size and under

TABLE IV

TRAINING EFFICIENCY OF GTA AND DIFFERENT MODELS FOR 15-min TRAFFIC FLOW FORECASTING

	VAR	MLP	SAE	LSTM	ConvLSTM	T-GCN	STGCN	STSGCN	ASTGCN	GMAN	DCRNN	Graph-WaveNet	GTA
# of epochs	7008	109	60	136	93	2969	49	169	79	67	100	62	87
time (hrs)	3.01	0.14	0.30	0.22	0.20	11.67	0.25	8.68	1.68	2.87	3.42	1.18	0.66

1.75 compared with that of a 32 batch size. One potential reason is that the batch size mainly affects the optimization time required for the model to reach a certain precision.

Besides, from Figure 5(g), (h), and (i), it appears that the prediction error of the model little varies as the number of hidden layer units changes. More specifically, the MAPE value of GTA of 200 hidden layer units in 60 minutes is reduced by 0.08% compared with that of 100 hidden units, and reduced by 0.29% compared with that of 300 hidden units. The results reveal that GTA is less sensitive to the number of hidden layer units, at least in the range of 100 to 300.

G. Efficiency

To investigate the training efficiency of GTA, we conduct a comparison with 11 other competitive baselines. Due to the space limitation, we only present the case for 15-min traffic flow prediction. The same conclusions are also applicable to 30-min and 60-min traffic flow forecasting. Table IV lists two attributes that include the number of epochs and time, which are used to evaluate the training efficiency of models.

From Table IV, we can observe that the training efficiency of our proposed model is considered acceptable. For DCRNN, its efficiency is lower than that of GTA. More specifically, in comparison, GTA takes less time to complete the training of 100 epochs. One possible reason is that there are many tedious and time-consuming convolution operations in DCRNN. The same holds for T-GCN, STSGCN, ASTGCN, and GMAN. The results show that a reasonable balance between accuracy and efficiency needs to be achieved.

Besides, deep learning models can dramatically improve efficiency compared to traditional models when involving extensive training data. For instance, MLP achieves faster convergence than VAR, probably due to GPU acceleration. Different deep models significantly show different efficiencies. More specifically, compared with MLP, SAE takes more time due to pre-training, but fewer epochs to reach convergence. In addition, the complexity of models may also affect training efficiencies, such as ConvLSTM and Graph-WaveNet.

H. Effect of Traffic Patterns and Their Sequence Lengths

To evaluate the effect of different traffic patterns, we perform ablation analysis using 15-min traffic forecasting as an example, as shown in Table V. NoMP, NoWP, and NoCP indicate we eliminate the monthly pattern, weekly pattern, and current pattern respectively. It is shown that the performance decreases to varying degrees without these traffic patterns. More specifically, the MAPE values for NoMP, NoWP, and NoCP increase by 15.0%, 19.6%, and 34.0% relative to GTA respectively, which indicates that the significance ranking of these traffic patterns are as follows: the current pattern, weekly pattern, and monthly pattern (from high to low). In particular,

TABLE V

THE ABLATION ANALYSIS OF TRAFFIC PATTERNS

Metric	NoMP	NoWP	NoCP	ours
MAE	22.42	23.44	25.98	**21.86**
RMSE	37.09	38.62	42.53	**36.94**
MAPE	17.6%	18.3%	20.5%	**15.3%**

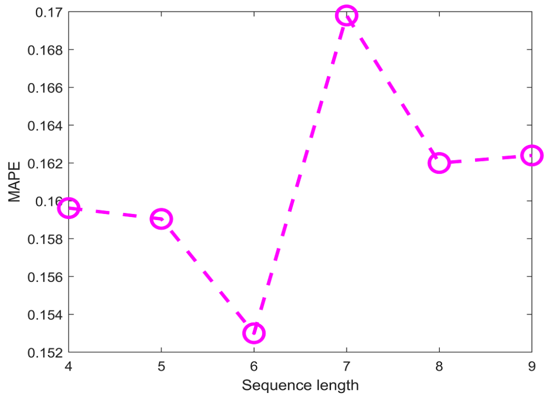


Fig. 6. MAPE with respect to input sequence lengths for GTA.

GTA performs the best, demonstrating that the integration of these patterns is effective for multi-sensor traffic forecasting.

Besides, we also investigate how the sequence length affects the performance of GTA. Figure 6 illustrates the MAPE values of GTA at different sequence lengths for 15-min traffic flow forecasting. Note that our model yields the lowest prediction error with a sequence length of 6. When the length is lower than 6, the performance improves gradually as the sequence length increases, which indicates the importance of taking account of temporal correlations. Our model shows a reduced performance when the length is increased beyond 6. One potential reason is the difficulty of modeling longer temporal dependencies.

I. Effect of Adding Fully Connected Layers

We also study the change in training costs after adding the fully connected layer (F), which combines all the temporal module, attention mechanism, and graph embedding. Contrast experiments are conducted using different quantities of fully connected layers (0 to 3). Similarly, we only depict the results for 15-min traffic flow forecasting. Figure 7 shows the training loss curves of these models after the max-min normalization.

One important observation from the results in Figure 7 is that the training costs increase apparently with increasing the number of fully connected layers. One possible reason is that the model parameter space is too large. Besides, there is no doubt that the test errors of these models show differences. More specifically, the MAE, RMSE, and MAPE values for GTA with one F are 22.19, 37.02, and 15.94%, respectively. Compared with DCRNN, it exhibits better performance in terms of MAE and RMSE, but not in terms of MAPE.

TABLE VI
THE MAPE, RMSE, AND MAE VALUE OF GTA AND SEVERAL BASELINES AT DIFFERENT MISSING RATES

Task	Metric	LATC-TNN	GMN-6	GMN-8	GMN-10	SGMN-6	SGMN-8	SGMN-10	**GTA**
Missing Rate = 10%	MAE	69.34	42.08	42.47	42.27	39.35	39.29	39.34	**32.08**
	RMSE	122.68	72.28	73.09	72.58	66.94	66.82	66.91	**53.29**
	MAPE	35.3%	23.8%	23.8%	23.7%	16.5%	16.5%	**16.5%**	21.3%
Missing Rate = 20%	MAE	70.34	45.43	46.81	47.22	42.04	42.02	41.98	**40.06**
	RMSE	124.21	75.58	79.84	80.35	72.25	72.17	72.12	**65.41**
	MAPE	35.8%	26.6%	26.5%	26.3%	18.2%	18.2%	**18.1%**	23.2%
Missing Rate = 40%	MAE	73.44	**47.48**	50.56	54.48	51.94	49.98	49.72	53.13
	RMSE	128.94	**77.76**	81.30	88.79	91.89	87.19	86.28	83.54
	MAPE	37.3%	28.1%	29.3%	30.9%	23.2%	21.9%	**21.6%**	28.2%

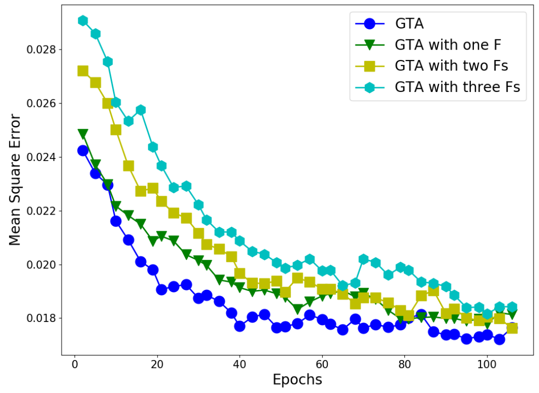


Fig. 7. Training cost of GTA with different quantities of fully connected layers.

TABLE VII
THE PERFORMANCE OF GTA BEFORE AND AFTER REPLACING ITS TEMPORAL MODULE

Metric	T-CNN	T-MLP	T-GRU	ours
MAE	22.12	22.23	**21.84**	21.86
RMSE	37.17	37.45	36.97	**36.94**
MAPE	15.5%	15.7%	15.4%	**15.3%**

TABLE VIII
THE PERFORMANCE OF GTA BEFORE AND AFTER REPLACING ITS GRAPH EMBEDDING TECHNOLOGY

Metric	G-RS	G-NDR	ours
MAE	21.96	22.09	**21.86**
RMSE	37.12	37.47	**36.94**
MAPE	17.7%	17.9%	**15.3%**

However, for GTA with two or three Fs, they yield higher prediction error under all evaluation metrics. These results also demonstrate the rationality of our model architecture design.

J. Prediction Under Missing Data

As discussed in [42], [53], [54], there are often data quality issues in real-world scenarios, such as missing data caused by sensor damage. Therefore, we further investigate how our model performs under missing data. It is compared with three competitive baselines (LATC-TNN, GMN, and SGMN). Due to space limitations, we only present the results for 15-min traffic flow forecasting. Table VI illustrates the prediction errors of GTA and several baselines at the missing rates of 10%, 20%, and 40%. Here, GMN-n and SGMN-n stand for the GMN and SGMN models with n-steps of historical data, respectively.

TABLE IX
THE PERFORMANCE OF GTA BEFORE AND AFTER REPLACING ITS ATTENTION MODULE

Metric	A-FLR	ours
MAE	22.02	**21.86**
RMSE	37.85	**36.94**
MAPE	17.8%	**15.3%**

As shown in Table VI, in terms of MAE and RMSE, GTA yields the lowest prediction error at the missing rates of 10% and 20%. More specifically, in the order listed in Table VI, the MAE values for other baselines decrease by 116.1%, 31.2%, 32.4%, 31.8%, 22.7%, 22.5%, and 22.6% relative to the GTA at the missing rates of 10%. When the missing rate is 40%, GMN-6 achieve better prediction performance. The results reveal that GTA is less suitable for those sensors with high missing rates. In terms of MAPE, SGMN-10 outperforms other methods for all missing rates. A higher MAPE value and lower MAE and RMSE values indicate that our model tends to yield smaller absolute errors at higher traffic volumes than at lower ones. Moreover, LATC-TNN exhibits the worst performance with different missing rates under all evaluation metrics. One possible reason is that this method ignores the spatial dependencies of traffic patterns.

K. Module Replacement Analysis

In this section, we explore whether our framework design is separate and replaceable using 15-min traffic flow forecasting as an example. Several variants for components such as the temporal module, attention mechanism, and graph embedding technology are constructed for experiments.

More specifically, for the temporal module, we attempt to replace the LSTM with other common deep learning methods such as CNN, MLP, and GRU. The corresponding overall models are abbreviated to T-CNN, T-MLP, and T-GRU, respectively. In terms of graph embedding technology, there are two variants; one abbreviated to G-RS is a random selection (RS) from the adjacency matrix, the other abbreviated to G-NDR is nonlinear dimensionality reduction (NDR) of the matrix using MLP. As for the attention mechanism, we only design a variant abbreviated to A-FLR, which assigns a free learnable parameter (FLR) for each affine transformation matrix. Specific analyses of these models are as follows.

Table VII demonstrates that T-GRU may constitute an efficient alternative for the temporal model when focusing on MAE. More particularly, T-GRU yields a lower prediction error than GTA in terms of MAE, but not in terms of RMSE and MAPE. Besides, we can also observe that T-MLP performs worst among all variants of the temporal module, which implies MLP extracts insufficient temporal correlation. For T-CNN, it is also inferior to GTA under all evaluation metrics. These results indicate that LSTM is more suitable to model temporal dependency compared with other common deep models.

One observation in Table VIII is that graph embedding technology plays a role in spatial dependency modeling. More specifically, GTA exhibits better performance than G-RS and G-NDR in terms of MAE, RMSE, and MAPE. The main reason is that graph embedding technology not only preserves the intrinsic properties of spatial structure but also incorporates context.

Besides, from Table IX, it can be observed that A-FLR exhibits lower performance than GTA in terms of MAE, RMSE, and MAPE. More specifically, the MAE, RMSE, and MAPE values of A-FLR are 22.02, 37.85, and 17.8%, respectively, which are 0.16, 0.91, and 2.5% respectively higher than that of GTA. One potential reason is that spatio-temporal dependencies are effectively and comprehensively integrated.

VI. Discussions

A. Deployment Considerations

Training and evaluation of deep models are typically carried out on a large number of historical samples. However, their performance might degrade in a production environment since the data distribution used by the model in the prediction phase is different from that used in the training phase. Therefore, the quality of the deployed model needs to be monitored to detect problems quickly and take corrective actions. One routine action is to retrain the model regularly, but it comes with high costs.

The combination of incremental learning and retraining may offer a valid and feasible solution in practical applications. More specifically, model weights are estimated from historical samples and then trained with a smaller learning rate based on incremental traffic data. When the prediction errors such as MAE, RMSE, and MAPE, are greater than the acceptance threshold, models will be retrained on all samples from certain time periods. Indeed, this strategy has been already widely applied in recommendation systems.

Moreover, another important consideration is the real-time requirements of deployed models. It can be optimized by 1) using TensorRT to speed up inference, 2) using a dedicated GPU card for inference, and 3) using the model compression method to reduce the computational load. In terms of our proposed model, its inference time (0.018 seconds) is much lower than the time interval of traffic flow prediction, which shows it is well suitable for real environments.

B. Key Benefit of GTA

Traffic flow forecasting has been investigated for many years. Current state-of-the-art approaches rely on the adjacency matrix, which is extracted from the pairwise road network distances by manually setting the threshold. This may cause insufficient spatial-temporal dependencies between the sensor and its distant neighbors in multiple time steps. In contrast, GTA introduces a combination of direct and indirect ways, which can integrate spatial-temporal dependencies more comprehensively. Graph embedding technology is employed to extract latent vector representations from the topology of transportation networks, which can better model spatial dependencies between sensors. Besides, the integration of multiple patterns (monthly periodicity, weekly periodicity, and variability) can also lead to better performance.

C. Future Direction

In the future, we believe these two research directions are important for traffic forecasting.

1) Traffic Prediction in Extreme Cases: Compared with normal conditions, it is more challenging to predict traffic flow in extreme cases such as peak-hour and post-accident scenarios. In [76], the authors employed SAE to extract abstract representations from static accident features and then combined with LSTM to capture traffic patterns of extreme conditions. However, the proposed method ignores the spatial dependencies between sensors, and its performance needs to be further improved.

2) Long-Term Temporal Dependency Modeling: Periodic patterns often exist in the traffic flow. Currently, most studies adopted RNNs with GRU or LSTM to capture the long-term dependency from the time sequences with periodic features. However, they are still insufficient for long sequential data. Therefore, there is a need for further research in long-term temporal dependency learning.

VII. Concluding Remarks

In this paper, we have proposed GTA, a graph-based temporal attention framework that incorporates the spatio-temporal dependencies, to conduct multi-sensor traffic flow forecasting over different time horizons. Inspired by representation learning of social networks, we propose to employ graph embedding technology on sensor networks, which can better extract the spatial dependency. Moreover, an attention mechanism is employed to adaptively identify the relations among temporal submodules (monthly pattern, weekly pattern, and current pattern). Spatio-temporal dependencies are more effectively and comprehensively integrated due to the full use of the topological properties of transportation networks. Experiments on a large-scale traffic dataset of England show that our approach significantly outperforms the state-of-the-art baselines and has strong stability and robustness.

In the future, we will focus on the following aspects to extend our work: (1) conduct multi-step traffic forecasting; (2) perform experiments on more traffic data from different cities; (3) introduce other exogenous variables (holidays, accidents, weather, etc.) into the model for more accurate forecasting.

References

[1] L. Wang, X. Geng, X. Ma, F. Liu, and Q. Yang, "Cross-city transfer learning for deep spatio-temporal prediction," 2018, *arXiv:1802.00386*. [Online]. Available: http://arxiv.org/abs/1802.00386

[2] B. Çetiner, M. Sari, and O. Borat, "A neural network based traffic-flow prediction models," *Math. Comput. Appl.*, vol. 15, no. 2, pp. 269–278, 2010.

[3] Y. Kamarianakis, W. Shen, and L. Wynter, "Real-time road traffic forecasting using regime-switching space-time models and adaptive LASSO," *Appl. Stochastic Models Bus. Ind.*, vol. 28, no. 4, pp. 297–315, Jul. 2012.

[4] N. G. Polson and V. O. Sokolov, "Deep learning for short-term traffic flow prediction," *Transp. Res. C, Emerg. Technol.*, vol. 79, pp. 1–17, Jun. 2017.

[5] X. Kong, X. Song, F. Xia, H. Guo, J. Wang, and A. Tolba, "LoTAD: Long-term traffic anomaly detection based on crowdsourced bus trajectory data," *World Wide Web*, vol. 21, no. 3, pp. 825–847, May 2018.

[6] A. A. Cook, G. Misirli, and Z. Fan, "Anomaly detection for IoT time-series data: A survey," *IEEE Internet Things J.*, vol. 7, no. 7, pp. 6481–6494, Jul. 2020.

[7] S. Dafermos and F. T. Sparrow, "Optimal resource allocation and toll patterns in user-optimised transport networks," *J. Transp. Econ. Policy*, vol. 5, no. 2, pp. 184–200, 1971.

[8] A. Bhattacharya, S. A. Kumar, M. K. Tiwari, and S. Talluri, "An intermodal freight transport system for optimal supply chain logistics," *Transp. Res. C, Emerg. Technol.*, vol. 38, pp. 73–84, Jan. 2014.

[9] M. Morioka, K. Kuramochi, Y. Mishina, T. Akiyama, and N. Taniguchi, "City management platform using big data from people and traffic flows," *Hitachi Rev.*, vol. 64, no. 1, p. 53, 2015.

[10] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, "Internet of Things for smart cities," *IEEE Internet Things J.*, vol. 1, no. 1, pp. 22–32, Feb. 2014.

[11] J. Tang, F. Liu, Y. Zou, W. Zhang, and Y. Wang, "An improved fuzzy neural network for traffic speed prediction considering periodic characteristic," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2340–2350, Sep. 2017.

[12] Z. Hou and X. Li, "Repeatability and similarity of freeway traffic flow and long-term prediction under big data," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 6, pp. 1786–1796, Jun. 2016.

[13] W. Zeng, C.-W. Fu, S. M. Arisona, S. Schubiger, R. Burkhard, and K.-L. Ma, "Visualizing the relationship between human mobility and points of interest," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 8, pp. 2271–2284, Aug. 2017.

[14] C. Song, Z. Qu, N. Blumm, and A.-L. Barabasi, "Limits of predictability in human mobility," *Science*, vol. 327, no. 5968, pp. 1018–1021, Feb. 2010.

[15] C. K. Moorthy and B. G. Ratcliffe, "Short term traffic forecasting using time series methods," *Transp. Planning Technol.*, vol. 12, no. 1, pp. 45–56, Jul. 1988.

[16] S. R. Chandra and H. Al-Deek, "Predictions of freeway traffic speeds and volumes using vector autoregressive models," *J. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 53–72, May 2009.

[17] S. Innamaa, "Short-term prediction of traffic situation using MLP-neural networks," in *Proc. 7th World Congr. Intell. Transp. Syst.*, Turin, Italy, 2000, pp. 6–9.

[18] Z. Zhao, W. Chen, X. Wu, P. C. Y. Chen, and J. Liu, "LSTM network: A deep learning approach for short-term traffic forecast," *IET Intell. Transp. Syst.*, vol. 11, no. 2, pp. 68–75, Mar. 2017.

[19] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, and Y. Wang, "Learning traffic as images: A deep convolutional neural network for large-scale transportation network speed prediction," *Sensors*, vol. 17, no. 4, p. 818, Apr. 2017.

[20] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. 31st Youth Academic Annu. Conf. Chin. Assoc. Autom. (YAC)*, Nov. 2016, pp. 324–328.

[21] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

[22] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 187–197, May 2015.

[23] X. Song, H. Kanasugi, and R. Shibasaki, "Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level," in *Proc. IJCAI*, vol. 16, 2016, pp. 2618–2624.

[24] G. Yang, Y. Wang, H. Yu, Y. Ren, and J. Xie, "Short-term traffic state prediction based on the spatiotemporal features of critical road sections," *Sensors*, vol. 18, no. 7, p. 2287, Jul. 2018.

[25] H. Yao *et al.*, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–8.

[26] Y. Wu and H. Tan, "Short-term traffic flow forecasting with spatial-temporal correlation in a hybrid deep learning framework," 2016, *arXiv:1612.01022*. [Online]. Available: http://arxiv.org/abs/1612.01022

[27] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, Vancouver, BC, Canada, 2018, pp. 1–33.

[28] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 922–929.

[29] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 1, pp. 914–921.

[30] S. Zhang, Z. Kang, Z. Zhang, C. Lin, C. Wang, and J. Li, "A hybrid model for forecasting traffic flow: Using layerwise structure and Markov transition matrix," *IEEE Access*, vol. 7, pp. 26002–26012, 2019.

[31] X. Wang *et al.*, "Traffic flow prediction via spatial temporal graph neural network," in *Proc. Web Conf.*, Apr. 2020, pp. 1082–1092.

[32] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "GaAN: Gated attention networks for learning on large and spatiotemporal graphs," 2018, *arXiv:1803.07294*. [Online]. Available: http://arxiv.org/abs/1803.07294

[33] L. Zhao *et al.*, "T-GCN: A temporal graph convolutional network for traffic prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 9, pp. 3848–3858, Sep. 2020.

[34] C. Zheng, X. Fan, C. Wang, and J. Qi, "GMAN: A graph multi-attention network for traffic prediction," in *Proc. 34th AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 1, pp. 1234–1241.

[35] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," 2017, *arXiv:1709.04875*. [Online]. Available: http://arxiv.org/abs/1709.04875

[36] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, "Short-term traffic forecasting: Where we are and where we're going," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 3–19, Jun. 2014.

[37] E. I. Vlahogianni, J. C. Golias, and M. G. Karlaftis, "Short-term traffic forecasting: Overview of objectives and methods," *Transp. Rev.*, vol. 24, no. 5, pp. 533–557, Sep. 2004.

[38] G. E. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. Hoboken, NJ, USA: Wiley, 2015.

[39] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traffic volume forecasting," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1678, no. 1, pp. 179–188, Jan. 1999.

[40] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov. 2003.

[41] M. Van Der Voort, M. Dougherty, and S. Watson, "Combining Kohonen maps with ARIMA time series models to forecast traffic flow," *Transp. Res. C, Emerg. Technol.*, vol. 4, no. 5, pp. 307–318, Oct. 1996.

[42] X. Chen and L. Sun, "Low-rank autoregressive tensor completion for multivariate time series forecasting," 2020, *arXiv:2006.10436*. [Online]. Available: http://arxiv.org/abs/2006.10436

[43] Y. Qi and S. Ishak, "A hidden Markov model for short term prediction of traffic conditions on freeways," *Transp. Res. C, Emerg. Technol.*, vol. 43, pp. 95–111, Jun. 2014.

[44] W.-C. Hong, "Traffic flow forecasting by seasonal SVR with chaotic simulated annealing algorithm," *Neurocomputing*, vol. 74, nos. 12–13, pp. 2096–2107, Jun. 2011.

[45] H. Tan, X. Xuan, Y. Wu, Z. Zhong, and B. Ran, "A comparison of traffic flow prediction methods based on DBN," in *Proc. CICTP*, Jul. 2016, pp. 273–283.

[46] Q. Wang, J. Gao, and Y. Yuan, "Embedding structured contour and location prior in siamesed fully convolutional networks for road detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 1, pp. 230–241, Jan. 2018.

[47] Q. Wang, J. Gao, and Y. Yuan, "A joint convolutional neural networks and context transfer for street scenes labeling," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 5, pp. 1457–1470, May 2018.

[48] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," 2017, *arXiv:1705.02699*. [Online]. Available: http://arxiv.org/abs/1705.02699

[49] Y. Liu, H. Zheng, X. Feng, and Z. Chen, "Short-term traffic flow prediction with conv-LSTM," in *Proc. 9th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2017, pp. 1–6.

[50] S. Du, T. Li, X. Gong, Y. Yang, and S. J. Horng, "Traffic flow forecasting based on hybrid deep learning framework," in *Proc. 12th Int. Conf. Intell. Syst. Knowl. Eng. (ISKE)*, Nov. 2017, pp. 1–6.

[51] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[52] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 1907–1913.

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

[54] X. Chen, Z. He, and J. Wang, "Spatial-temporal traffic speed patterns discovery and incomplete data recovery via SVD-combined tensor decomposition," *Transp. Res. C, Emerg. Technol.*, vol. 86, pp. 59–77, Jan. 2018.

[55] Z. Cui, R. Ke, Z. Pu, X. Ma, and Y. Wang, "Learning traffic as a graph: A gated graph wavelet recurrent neural network for network-scale traffic prediction," *Transp. Res. C, Emerg. Technol.*, vol. 115, Jun. 2020, Art. no. 102620.

[56] J. Im and S. Cho, "Distance-based self-attention network for natural language inference," 2017, *arXiv:1712.02047*. [Online]. Available: http://arxiv.org/abs/1712.02047

[57] K. Zhang, G. Lv, E. Chen, L. Wu, Q. Liu, and C. P. Chen, "Context-aware dual-attention network for natural language inference," in *Proc. PAKDD*. Cham, Switzerland: Springer, 2019, pp. 185–198.

[58] Q. Wang, T. Han, Z. Qin, J. Gao, and X. Li, "Multitask attention network for lane detection and fitting," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Dec. 8, 2020, doi: 10.1109/TNNLS.2020.3039675.

[59] F. Gao, W. Shi, J. Wang, A. Hussain, and H. Zhou, "A semi-supervised synthetic aperture radar (SAR) image recognition algorithm based on an attention mechanism and bias-variance decomposition," *IEEE Access*, vol. 7, pp. 108617–108632, 2019.

[60] Q. Wang, S. Liu, J. Chanussot, and X. Li, "Scene classification with recurrent attention of VHR remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 57, no. 2, pp. 1155–1167, Feb. 2019.

[61] J. Salazar, K. Kirchhoff, and Z. Huang, "Self-attention networks for connectionist temporal classification in speech recognition," in *Proc. ICASSP-IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 7115–7119.

[62] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 701–710.

[63] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 855–864.

[64] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1225–1234.

[65] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, May 2015, pp. 1067–1077.

[66] R. D. Vallam *et al.*, "Deepaggregation: A new approach for aggregating incomplete ranked lists using multi-layer graph embedding," in *Proc. 18th Int. Conf. Auton. Agents Multiagent Syst.*, 2019, pp. 2235–2237.

[67] R. Hisano, "Semi-supervised graph embedding approach to dynamic link prediction," in *Proc. Int. Workshop Complex Netw.* Cham, Switzerland: Springer, 2018, pp. 109–121.

[68] H. Yao, X. Tang, H. Wei, G. Zheng, Y. Yu, and Z. Li, "Modeling spatial-temporal dynamics for traffic prediction," 2018, *arXiv:1803.01254*. [Online]. Available: https://arxiv.org/abs/1803.01254

[69] W. Huang, G. Song, H. Hong, and K. Xie, "Deep architecture for traffic flow prediction: Deep belief networks with multitask learning," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 5, pp. 2191–2201, Oct. 2014.

[70] K. Tang, S. Chen, and Z. Liu, "Citywide spatial-temporal travel time estimation using big and sparse trajectories," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 12, pp. 4023–4034, Dec. 2018.

[71] S. Zhang, Z. Kang, Z. Hong, Z. Zhang, C. Wang, and J. Li, "Traffic flow prediction based on cascaded artificial neural network," in *Proc. IGARSS-IEEE Int. Geosci. Remote Sens. Symp.*, Jul. 2018, pp. 7232–7235.

[72] J. Zhang, Y. Zheng, D. Qi, R. Li, X. Yi, and T. Li, "Predicting citywide crowd flows using deep spatio-temporal residual networks," *Artif. Intell.*, vol. 259, pp. 147–166, Jun. 2018.

[73] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," 2013, *arXiv:1310.4546*. [Online]. Available: http://arxiv.org/abs/1310.4546

[74] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," in *Proc. 13th Annu. Conf. Int. Speech Commun. Assoc.*, 2012, pp. 194–197.

[75] M. Tan, C. D. Santos, B. Xiang, and B. Zhou, "LSTM-based deep learning models for non-factoid answer selection," 2015, *arXiv:1511.04108*. [Online]. Available: http://arxiv.org/abs/1511.04108

[76] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, and Y. Liu, "Deep learning: A generic approach for extreme condition traffic forecasting," in *Proc. Int. Conf. Data Mining*. Philadelphia, PA, USA: SIAM, 2017, pp. 777–785.

Shaokun Zhang is currently pursuing the Ph.D. degree with the Department of Computer Science, Peking University, China. His research interests include intelligent transportation systems, mobile application testing, mobile energy modeling, and machine learning.

Yao Guo (Member, IEEE) received the Ph.D. degree in computer engineering from the University of Massachusetts at Amherst in 2007. He is currently a Full Professor with the Department of Computer Science, Peking University. He has been serving as the Vice Chair of Computer Science, since 2013. His general research interests include operating systems, mobile computing and applications, low-power design, and software engineering.

Peize Zhao received the B.Eng. degree in Internet of Things engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2019. He is currently pursuing the Ph.D. degree with the Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology, Hong Kong. His current research interests include bioinformatics and computational biology.

Chuanpan Zheng received the B.Sc. degree in applied physics from Shandong University, Jinan, China, in 2012. He is currently pursuing the Ph.D. degree in computer science and technology with the Fujian Key Laboratory of Sensing and Computing for Smart Cities, School of Informatics, Xiamen University, China. His research interests include big data analytics and machine learning.

Xiangqun Chen received the B.S. and M.S. degrees in computer science from Peking University. She is currently a Professor with the Department of Computer Science, Peking University. Her research interests include operating systems and software engineering.