ARTICLE IN PRESS

Knowledge-Based Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Citywide traffic speed prediction: A geometric deep learning approach

James J.Q. Yu

Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China

ARTICLE INFO

Article history:
Received 1 September 2020
Received in revised form 20 October 2020
Accepted 1 November 2020
Available online xxxx

Keywords:
Traffic speed prediction
Geometric deep learning
Large-scale transportation network
Intelligent transportation system
Data-driven approach

ABSTRACT

Accurate traffic speed prediction is critical to modern internet of things-based intelligent transportation systems. It serves as the foundation of advanced traffic management systems and travel services. Nonetheless, the large number of roads and sensors impose great computational burden to existing forecast approaches, most of which can only handle one or few roads at a time. In this paper, a novel data-driven deep learning-based approach is proposed for citywide traffic speed prediction. The proposed approach is grounded on recent developments of geometric deep learning techniques to fully utilize the topological information of road networks in the learning process. Specifically, the approach captures the geometric traffic data dependency with graph convolution and attention mechanisms, and the temporal data correlation is extracted and expanded using the encoder–decoder architecture within a generative adversarial learning framework. Comprehensive case studies are conducted with real-world urban road networks and respective data to evaluate its performance, where consistent improvements can be observed over baseline approaches. Lastly, an architectural study is carried out to discover the best-performing structure of the proposed approach, whose sensitivity to data noise and sample frequency is also assessed.

1. Introduction

Predicting traffic speed information timely with high accuracy is a fundamental challenge in modern intelligent transportation systems (ITS) [1], which are widely recognized as an essential component in smart cities. With the adoption of physical device inter-connections, commonly referred to as Internet of Things (IoT), a massive volume of real-time traffic data are streamlined to the advanced traffic management systems (ATMS) for traffic control and prediction, which greatly help the system operator and road users obtain the traffic dynamics in order to make travel decision, prevent traffic congestions, and improve transportation throughput [2,3], etc. For instance, online routing service adopts citywide current and future traffic speed data to provide users with real-time routes that avoid traffic jams, which in turn reduces driving time and increases transportation network capacity [4]. The predictions are considered indispensable in deploying modern ITS [5,6].

Due to the importance of speed predictions, a plethora of research effort has been dedicated in devising algorithms for providing such functionality [7]. Thanks to traditional and emerging sensor sources, e.g., inductive loop, surveillance video camera, radar sensors, and global positioning system, modern traffic management is empowered with big data. Correspondingly, data-driven techniques received wide attention in making speed and flow predictions in recent years. In general, these techniques can be classified into three categories: naïve (e.g., history average), parametric (e.g., auto-regressive integrated moving average (ARIMA) variants [8,9], Kalman filter [10]), and non-parametric models (e.g., Bayesian networks [11,12], neural networks [1,13,14]). These classes of techniques have their own merits, and their recent derived approaches can achieve satisfactory prediction accuracy [5]. In addition to the data-centric techniques, traffic simulation models are also commonly adopted to make speed predictions. Different from previous approaches, such techniques simulate the traffic in microscopic, mesoscopic, or macroscopic view with multi-agent systems to model a wide variety of traffic phenomena [15,16]. Due to the model transparency characteristic, they are better to be adopted in what–if reasoning, decision influence assessment and dynamics research. Nonetheless, data-driven approaches are generally considered more robust to noise and performant with sufficient past data [17].

In recent years, traffic speed and flow predictions are witnessing the emerge of a new data learning technique, i.e., deep learning [1,18]. As a branch of non-parametric models and neural networks, deep learning models fully utilize the enormous volume of historical traffic data with its multi-layer architecture to extract latent raw data correlations. Furthermore, the highly complicated and somehow stochastic traffic flow process can

E-mail address: yujq3@sustech.edu.cn.

https://doi.org/10.1016/j.knosys.2020.106592
0950-7051/© 2020 Elsevier B.V. All rights reserved.

be emulated or learnt by these models without prior domain-specific knowledge thanks to their huge model capacity. These properties make deep learning a good solution for transportation research, and existing results have demonstrated its outstanding performance [1,14,19–23].

Nonetheless, there exists a research gap in traffic speed prediction when considering modern urban road networks. Existing methods mostly predict speed data for a specific road by learning its traffic speed dynamics in historical data with parametric or non-parametric learning techniques, see [1,10] for some recent examples. While temporal data correlation is critical in predictions, road network topology can help improve the forecasting accuracy by incorporating spatial (geometric) data correlation among adjacent roads. Furthermore, these approaches are devised considering only one or few roads in the road network. As ATMS requires detailed traffic speed predictions to most, if not all, of the roads in the urban area, such approaches incur significant computational burden: a large number of individual models need to be developed for online data prediction, which drastically increases both offline training and online inference time. Few recent results demonstrate the possibility of producing predicted speed values in bulk by considering the spatial properties of road networks, see [13,14] for examples. Yet merely topology images or random walked traffic diffusion data are employed in the prediction, rendering a principal description to topologies — adjacency matrix — under-utilized. The presented results discuss the application of these approaches on small data sets that only cover several roads in traffic networks. Their applicability to citywide predictions with thousands of distinct roads remains unknown. While there exists efforts on employing graph neural network variants in handling the forecast task, e.g., [24,25], the respective feature extraction model exploits spatial and temporal correlations sequentially and independently. This may also lead to potential information loss and in turn performance degradation.

To fill the research gap, we propose a new data-driven citywide traffic speed prediction approach based on recent advancements in geometric deep learning theories. The proposed novel Generative Adversarial Graph Attention (GA^2) model specifically considers the practical scenarios in which accurate speed predictions for each road in an urban road network are required by the traffic management system. Different from existing work, GA^2 learns the geometric characteristics from the historical traffic data by utilizing the topology information and extracts spatial and temporal correlation simultaneously, which greatly improves the forecast accuracy. The proposed model incorporates the graph convolution principle from graph convolutional network (GCN) for geometric feature extraction, and the geometric attention mechanism from graph attention network (GAT) for model capacity augmentation. Additionally, the generative adversarial learning design paradigm is adopted to guide the design of GA^2. As far as we are concerned, this is among the pioneer work on real-time citywide traffic speed prediction based on geometric deep learning and attention mechanism. When evaluated on real-world data sets, GA^2 consistently outperforms state-of-the-art data-driven traffic forecasting baselines. To summarize:

- We propose a novel GA^2 model as a general-purpose time-series regression technique for non-Euclidean-structured data. The proposed model is capable of exploiting spatio-temporal data correlation in one go.
- We employ GA^2 to address the real-time citywide traffic speed prediction problem with tailor-made feature design and training approach. The incorporated attention mechanism significantly improves the model capability for handling large transportation networks.
- We conduct comprehensive case studies on three large-scale real-world road networks to investigate the efficacy of GA^2.

The rest of this paper is organized as follows. In Section 2, we give the mathematical formulation of the citywide traffic speed prediction problem and discuss recent related literature. In Section 3, we present the constituting component and model architecture of GA^2 with discussion on its uniqueness. In Section 4, we discuss the implementation details and improvements of GA^2 in addressing the speed prediction problem. Section 5 presents the case studies and discussions, and this paper is concluded in Section 6.

2. Citywide traffic speed prediction

In this section, we first introduce the urban road network model employed in this work with a brief formulation of the citywide traffic speed prediction problem. Existing related research is then summarized and presented.

2.1. Urban road network model

We consider the entire urban road network as a directed graph $G(\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of road intersections and $\mathcal{E}$ is the set of roads. Let $V_e^0 \in \mathcal{N}$ and $V_e^* \in \mathcal{N}$ be the starting and ending nodes of road $e \in \mathcal{E}$. Given the topology of an urban road network, graph G can be easily constructed by first creating a node for each road intersection in the topology, then creating arcs between nodes in accordance with the roads.

When predicting traffic speed data, we split the continuous time horizon into discrete time instances of length Δ. Symbols $\mathcal{T}^- = \{\cdots, -2, -1, 0\}$ and $\mathcal{T}^+ = \{1, 2, \ldots\}$ denote the past/future discrete time horizon until/from the current time instance ($t = 0$), respectively. For each time instance $t \in \mathcal{T}^- \cup \mathcal{T}^+$, symbol $v_{e,t}$ represents the average traffic speed of road $e \in \mathcal{E}$ during the time instance. Besides its traffic speed, each road is also described by a set of time-invariant attributes (e.g., speed limit and road width) denoted by $\mathcal{S}_e$.

The objective of citywide traffic speed prediction is to obtain predictions to the future averaged traffic speed data $\{v_{e,t} | \forall e \in \mathcal{E}, t \in \mathcal{T}^+\}$ by a speed predictor $\mathcal{P}(\cdots)$, which takes the past speed information and road attributes as inputs:

$$\hat{\mathcal{V}}_t = \mathcal{P}(\{\mathcal{V}_{t'} | \forall t' < t\}, \{\mathcal{S}_e | \forall e \in \mathcal{E}\}), \ \forall t \in \mathcal{T}^+, \tag{1}$$

where $\mathcal{V}_t = \{v_{e,t} | \forall e \in \mathcal{E}\}$, and $\hat{\mathcal{V}}_t = \{\hat{v}_{e,t} | \forall e \in \mathcal{E}\}$ is the prediction of $\mathcal{V}_t$ by $\mathcal{P}$. The aim of this work is to develop a geometric deep learning-based model to implement this speed predictor with high prediction accuracy. As accurate speed prediction data can greatly facilitate subsequent ITS applications, the fitness of the prediction is correspondingly measured by three common metrics, i.e., mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean square error (RMSE) for time instance $t \in \mathcal{T}^+$:

$$\text{MAE} = \frac{1}{|\mathcal{E}|} \sum_{e \in \mathcal{E}} |v_{e,t} - \hat{v}_{e,t}|, \tag{2a}$$

$$\text{MAPE} = \frac{1}{|\mathcal{E}|} \sum_{e \in \mathcal{E}} \frac{|v_{e,t} - \hat{v}_{e,t}|}{v_{e,t} + \varepsilon}, \tag{2b}$$

$$\text{RMSE} = (\frac{1}{|\mathcal{E}|} \sum_{e \in \mathcal{E}} |v_{e,t} - \hat{v}_{e,t}|^2)^{\frac{1}{2}}, \tag{2c}$$

where $\varepsilon = 10^{-6}$ is a small positive value preventing the divide-by-zero issue when $v_{e,t} = 0$.

2.2. Related research

We follow the introduction in Section 1 and discuss the three types of traffic speed and flow prediction techniques, i.e., naïve, parametric, and non-parametric [5]. Naïve models are the simplest ones, which use basic statistical assumptions to make the prediction. Among them, the instantaneous travel times (ITT) approach — use the measured value as the prediction to the next — and historical average (HA) are most widely adopted in real-world applications [26]. However, as no external properties of the traffic can be utilized to reduce the forecasting uncertainty, they are generally considered inferior than the other two types of approaches in terms of accuracy.

Parametric models emulate the traffic dynamics with a finite set of known parameters about the traffic data distribution. Time-series models such as ARIMA variants are arguably the most popular techniques besides traffic simulation and Kalman filter. ARIMA utilizes historical traffic data for regression with scalar-based models by considering data points in time-series depend on its previous values with additive noise. Ref. [8] is among the first attempts to adopt ARIMA in traffic speed prediction, in which data statistical stationarity and seasonality are employed to determine the model order and parameters. After that, results are published to extend the approach considering the strong seasonal characteristics of data [27], the spatial–temporal correlation [28,29], or with Kohonen self-organizing map [9]. Another ARIMA variant extends the original model with explanatory variables to form a vector ARIMA, which follows the principle of multivariate regression models to include other independent variables in the forecast [30]. While much effort has been devoted to this type of approaches in predicting the speed, it suffers from the internal linearized model and intrinsic difficulty in incorporating environmental data sources [5,26].

In recent years, machine learning and neural network techniques have drawn the attention of researchers [1]. Various models are utilized to construct learning approaches in the literature, e.g., stacked auto-encoder [1], convolutional long short-term memory (LSTM) network [13], diffusion convolutional recurrent network [14], etc. While these approaches demonstrate satisfactory prediction accuracy on single or few roads, they still cannot fulfill the requirements of citywide traffic speed prediction. A number of studies have been published very recently on employing graph neural networks or geometric features in traffic speed prediction. To name a few, Ref. [31] proposed a graph-convolution-only neural network for speed prediction within an area. However, the lack of recurrent units in the proposed approach weakens its capability of temporal feature extraction as the unique gating mechanism in modern recurrent neural networks are critical in the prediction [32]. Reference [33] devised a convolutional neural networks (CNN) and LSTM-based deep learning model to exploit the local geometrical features of a transportation network for speed forecast. Similarly, Ref. [24] proposed a GCN-gated-recurrent-unit network to capture first the spatial features and then the temporal ones from the historical traffic data. Nonetheless, such two-step feature extraction may not capture the exact spatiotemporal data characteristics which describes features in both domains simultaneously. While [32] and [34] are capable of doing so, the models are primarily designed for medium-sized transportation networks with several hundreds of sampling locations.

To fully address the citywide traffic speed prediction problem, we propose a novel GA^2 model that employs a new graph attention convolutional recurrent layer for latent feature extraction from historical traffic data. By integrating the graph convolution operation into the recurrent units instead of after them as did in the literature, the proposed model is capable of extracting the spatiotemporal data characteristics. Additionally, the incorporation of attention mechanism [35] enables the model to distinguish the critical/influential neighbors of network nodes from others, allowing it to handle large and complex transportation networks. Finally, the generative adversarial architecture improves the training efficacy of the model for further performance boost.

3. Generative adversarial Graph Attention (GA^2) network

In this section, we propose a new generative adversarial graph attention network for graph-based data prediction tasks. We first present some brief preliminaries on the deep learning principles employed in GA^2, then propose a graph attention convolutional recurrent layer (GAL) as the basic building block of GA^2. Finally, the network architecture of GA^2 is presented with discussion.

3.1. Preliminaries

The structural design of GAL borrows the data-flow design principle of GCN, which aims to extract latent information from non-Euclidean-structured data. GCN follows the idea of CNN and performs neighborhood information mixing on the source data. Nonetheless, different from conventional CNN which has a uniform Euclidean-space receptive field, GCN utilizes the connectivity structure of the input graph for information convolution [36]. Specifically, the convolution in GCN evolves from the graph convolution operation first given in [37]:

$$x_1 * x_2 = \mathbf{U}((\mathbf{U}^\mathsf{T} x_1) \circ (\mathbf{U}^\mathsf{T} x_2)), \tag{3}$$

where x_1 and x_2 are two signals defined on graph nodes. In the traffic prediction context, these nodal signals refer to the sensor data (speed, flow, etc.) gathered from the sensing locations in a transportation network. In (3), symbol $\mathbf{U}$ is the matrix of eigenvectors of graph Laplacian matrix $\mathbf{L}$, which represents the topology and connectivity of transportation network data sensing locations. Operator $\circ$ is element-wise multiplication. By convolution theorem [38], (3) can be extended to manipulate a single data source (nodal signal) x as:

$$x' = \mathbf{U}\Theta\mathbf{U}^\mathsf{T} x, \tag{4}$$

where x' is the output signal, $\Theta = \Theta(\Lambda)$ is a diagonal matrix of learnable filters, and Λ is the eigenvalues of $\mathbf{L}$. Adopting this nodal signal convolution operation, an lth graph convolutional layer with different filters can be defined as

$$x_j^{l+1} = \rho(\sum_{i=1}^{F^l} \mathbf{U}\Theta_{i,j}^l \mathbf{U}^\mathsf{T} x_i^l),\ \forall j = 1, \dots, F^{l+1}, \tag{5}$$

where x_j^l is the jth hidden representation for all nodes in the graph, $\rho(\cdot)$ is a non-linear activation function, F^l is the number of dimensions (filters) in layer l, and $\Theta_{i,j}^l$ is the learnable filter parameter.

However, there is a fundamental limitation for this convolution, i.e., the full eigenvectors of the Laplacian matrix Λ are required per each forward and backward pass, rendering $O(N^2)$ complexity [39]. This limits the scalability of the model. GCN addresses the issue by first adopting a polynomial filter and then transform the filter with Chebyshev expansion:

$$\Theta(\Lambda) = \sum_{k=0}^{K-1} \theta_k \Lambda^k = \sum_{k=0}^{K-1} \theta_k \mathcal{T}_k(\tilde{\Lambda}), \tag{6}$$

where K is the polynomial order, θ_k is a new learnable parameters, $\tilde{\Lambda} = 2\Lambda/\max\{\Lambda\} - \mathbf{I}$ are the re-scaled eigenvalues, and $\mathcal{T}_k(\cdot)$ is the Chebyshev polynomial of order k. Consequently, by using only the first-order approximation, (5) can be written as follows [36]:

$$x_j^{l+1} = \rho(\sum_{i=1}^{F^l}\sum_{k=0}^{K-1}\theta_k \mathbf{U}\mathcal{T}_k(\tilde{\Lambda})\mathbf{U}^\top x_i^l) = \rho(\sum_{i=1}^{F^l}\sum_{k=0}^{K-1}\theta_k\mathcal{T}_k(\tilde{\mathbf{L}})x_i^l)$$
$$\approx \rho(\sum_{i\in\mathcal{N}(j)}[\tilde{\mathbf{D}}(i,i)\tilde{\mathbf{D}}(j,j)]^{-\frac{1}{2}}\Theta^l x_i^l), \tag{7}$$

where $\tilde{\mathbf{L}} = 2\mathbf{L}/\max\{\Lambda\} - \mathbf{I}$, $\tilde{\mathbf{D}} = \mathbf{D} + \mathbf{I}$, $\mathbf{D}$ is the diagonal degree matrix of adjacency matrix $\mathbf{A}$ of the graph, and $\mathcal{N}(i)$ returns the neighborhood nodes of node i in the graph.

Based on GCN, GAL incorporates the attention mechanism [40] on graphs as proposed in GAT [35]. One of the most promising benefit of attention is that it enable the neural networks to deal with variable sized inputs and to focus on the most relevant ones instead of treating them equally [35]. This is achieved by modifying the convolution in (7) and introduce an additional attention term as follows:

$$x_j^{l+1} = \rho(\sum_{i\in\mathcal{N}(j)}\alpha_{ij}^l\Theta^l x_i^l), \tag{8}$$

where the attention α_{ij}^l is defined as

$$\alpha_{ij}^l = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^\top[\Theta^l x_j^l \| \Theta^l x_i^l]))}{\sum_{k\in\mathcal{N}(j)}\exp(\text{LeakyReLU}(\mathbf{a}^\top[\Theta^l x_k^l \| \Theta^l x_i^l]))}, \tag{9}$$

in which $\text{LeakyReLU}(x) = \max\{0, x\} - 0.2\max\{0, -x\}$ [39], $\mathbf{a}$ is the attention parameter, and $[\cdot\|\cdot]$ is the concatenation operation. From (7) to (8), the information presented by $\mathbf{D}$ is retained by $\mathcal{N}(j)$ (topology information) and α_{ij}^l (nodal data dependency).

This attention mechanism has two properties that are critical in citywide traffic speed prediction. Computationally, GAT is highly efficient due to its full parallelizability across all edges and nodes. No eigen-decompositions or expensive matrix computations are required. Furthermore, the model allows neural networks to assign different importance to nodes of a same neighborhood, rendering a massive model capacity improvement. This also accords with the intuition in transportation that specific incoming and outgoing flows are more influential to traffic than others.

3.2. Graph attention convolutional recurrent layer

As introduced in Section 3.1, GCN and GAT are capable for feature extraction from non-Euclidean-structured data, e.g., graphs. While this property fits well in urban transportation networks, traffic speed data are typically time series which require additional data processing mechanisms for temporally-correlated feature extraction. To achieve this, we follow the state-of-the-art time-series learning technique in deep learning, i.e., LSTM, to design GAL for simultaneous non-Euclidean-spatially- and temporally-correlated feature extraction. Specifically, each time-series is divided into multiple graphs, each of which corresponds to one time instance. These graph data are input into GAL sequentially, and the layer holds an internal state that stores latent time-series information from all graph data already processed. Finally, GAL outputs processed data features based on the final state information.

To construct this special finite state machine [41], we first formulate an internal cell state for GAL as follows[1]:

$$\mathbf{C}_t = \mathbf{f}_t \circ \mathbf{C}_{t-1} + \mathbf{i}_t \circ \tilde{\mathbf{C}}_t, \tag{10a}$$

[1] We discuss the propagation rule of the lth layer in a neural network. The superscript "l" is omitted when there is no ambiguity.

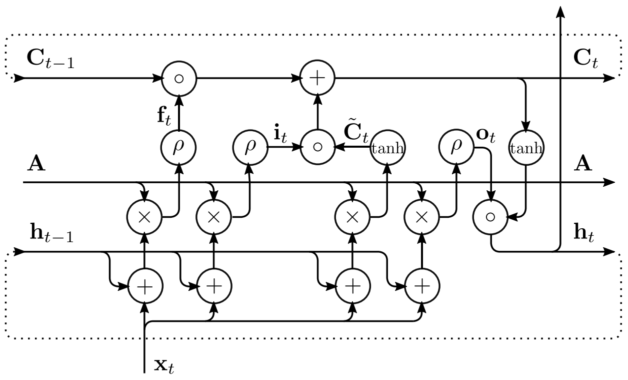


Fig. 1. Data flow of the proposed GAL cell state. Dotted line denotes the data flow between input data of consecutive time instances. Attentions, weights, and biases are remove to reduce visual distraction.

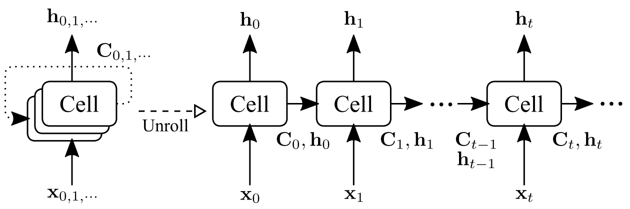

$\mathbf{h}_{0,1,\cdots}$
$\mathbf{C}_{0,1,\cdots}$
Cell
Unroll
$\mathbf{x}_{0,1,\cdots}$
$\mathbf{h}_0$
$\mathbf{h}_1$
$\mathbf{h}_t$
$\mathbf{C}_0, \mathbf{h}_0$
$\mathbf{C}_1, \mathbf{h}_1$
$\mathbf{C}_{t-1}$
$\mathbf{h}_{t-1}$
$\mathbf{C}_t, \mathbf{h}_t$
$\mathbf{x}_0$
$\mathbf{x}_1$
$\mathbf{x}_t$

Fig. 2. Data flow of the proposed GAL and its unrolled form.

where $\mathbf{f}_t$ and $\mathbf{i}_t$ control the information fusion from the previous state and new input data, respectively:

$$\mathbf{f}_t = \text{ReLU}(\mathbf{A}\alpha_f[\Omega_f\mathbf{x}_t + \Upsilon_f\mathbf{h}_{t-1}] + b_f), \tag{10b}$$
$$\mathbf{i}_t = \text{ReLU}(\mathbf{A}\alpha_i[\Omega_i\mathbf{x}_t + \Upsilon_i\mathbf{h}_{t-1}] + b_i), \tag{10c}$$
$$\tilde{\mathbf{C}}_t = \tanh(\mathbf{A}\alpha_c[\Omega_c\mathbf{x}_t + \Upsilon_c\mathbf{h}_{t-1}] + b_c), \tag{10d}$$

where $\mathbf{x}_t$ is the input graph data at time instance t, $\mathbf{h}_{t-1}$ is the output graph feature at time instance $t-1$, α matrices are the learnable attention term α_{ij} defined in (9), Ω and Υ matrices are new learnable weight parameters, and b vectors are learnable bias parameters. With the cell state being updated by new input data, the output can be correspondingly produced as follows:

$$\mathbf{h}_t = \mathbf{o}_t \circ \tanh(\mathbf{C}_t), \tag{10e}$$
$$\mathbf{o}_t = \text{ReLU}(\mathbf{A}\alpha_o[\Omega_o\mathbf{x}_t + \Upsilon_o\mathbf{h}_{t-1}] + b_o), \tag{10f}$$

where $\mathbf{o}_t$ controls the information output from the new state. Comparing with the LSTM propagation rules [42], (10) uses the attention mechanism given in (8) to substitute the original linear transformation counterparts in the form of **Wx**. This is how GAT and LSTM are integrated to construct the proposed GAL.

In Fig. 1 we present the data flow of GAL cell, which is a summary of Eq. (10). When some data are input into a GAL cell, data flows along the arrows in the figure until the output is produced and cell state is updated. Then the new information is looped back to the beginning of the flow, and the process proceeds to the next time instance. GAL is formulated by encapsulating a GAL cell as shown in the figure and transferring the inter-time-instance data. Fig. 2 illustrates a typical GAL and its unrolled form on the discrete time horizon. In this figure, the left-hand-side stacked block of cells refers to the structure presented in Fig. 1. When unrolling GAL, the same set of parameters (α, Ω, Υ, Θ, etc.) is employed to process the sequence of input data.

3.3. GA^2 network architecture

In Section 3.2, we propose GAL to model the temporal dependency in graph-structured data. While it is possible and straightforward to directly employ time-series citywide traffic speed data as the input $\mathbf{x}_t$ and enforce GAL to output speed predictions of the next time instances $\mathbf{h}_t$, such formulation cannot fully enjoy the benefits of deep neural network structures. By including adequately more layers to a same neural network and employing advanced network design principle, the model capacity can be further boosted, resulting in generally better latent information extraction capability.

We start with the task of generating future graph-data predictions based on historical information. In this multi-step data prediction, we leverage the sequence-to-sequence (seq2seq) architecture firstly proposed in [44], and stack multiple GALs to increase network depth as shown in Fig. 3. Specifically, the predictor is composed of two neural networks, namely, an encoder and a decoder. Both networks are recurrent neural networks with GAL as the recurrent unit to fully consider time-series data dependencies. To construct the encoder, five ($1+4$ in the figure) residual-link-enabled GAL layers with 32 filters each (Res-GAL, to be introduced next) are stacked first. Subsequently, a final standard GAL is employed to aggregate the extracted latent information and output the encoder state to initialize the decoder inputs. Batch normalization [45] is adopted between every two layers. The decoder shares a similar architecture with the encoder, except that the first Res-GAL layer accepts $32+F$ features instead of F in the encoder, where F is the number of node features in the input graph. In addition, the final GAL layer in the decoder has F filters, which is different from the encoder's 32.

In this encoder–decoder design, residual links — first proposed in deep residual networks [43] — are intensively employed to enable the networks to skip certain layers if deemed necessary by training. For GAL, a residual link can be included into (10) by replacing (10e) with the following equation:

$$\mathbf{h}_t = \mathbf{o}_t \circ \tanh(\mathbf{C}_t) + \mathbf{x}_t. \tag{11}$$

The principle of introducing such links is that the new structure is typically easier to be optimized using the current neural network training algorithms than the original one, since the solution space is smoothened by the identity mappings achieved by residual functions [43]. Therefore, the convergence will be possibly improved when training neural networks with deeper architectures, and overfitting issue can be alleviated. This hypothesis is supported by the case studies to be demonstrated in Section 5.3.

The data flow of the decoder is also different from that of the encoder, which takes the available time-series in the past as input. The output encoder state is later utilized to construct the input data to the decoder, which generates predictions given data in the previous time instance. When training the predictor, an available ground truth time-series is divided into two parts. The first sub-series is input into the encoder to develop the final state. Then the data in each time instance within the second ground-truth sub-series is concatenated with the final encoder state along the filter axis, which is used as the decoder input. At inference time when the network is well-trained and ready for making predictions, the available ground truth is input into the encoder. Then, instead of the ground truth of future time which is unavailable, the previous predictions generated by the network itself is combined with the encoder state. Finally, the decoder outputs the predicted time-series.

In a typical seq2seq model, the network is trained by maximizing the likelihood of the predicted data with ground truth, see [14,44] for some examples. Nonetheless, as will be demonstrated in the case studies, this training objective generally leads to unstable convergence due to its sensitivity to data noise in both predicted data and ground truth. To overcome this issue, we follow the principle of generative adversarial learning [46] and include a discriminator for better network tuning and name the whole network GA^2. The layered architecture of GA^2 is presented in Fig. 4, in which the encoder and decoder structures are the rolled form of the respective ones in Fig. 2 with y-axis transposed to x-axis in Fig. 4.

In GA^2, both the encoder–decoder sub-network and the discriminator sub-network form a two-player minimax game for a better parameter learning efficiency. In this game, the encoder–decoder develops new predicted time-series on $\mathcal{T}^+$ based on the information during $\mathcal{T}^-$, as illustrated previously. The discriminator tries to evaluate the "authenticity"[2] of the new time-series, and distinguish them from the available ground truth ones of $\mathcal{T}^-$. When the model is well-trained, the encoder–decoder is capable of making predictions that can compromise the discriminator. We construct the discriminator by adopting a GAL with 16 filters and stack four fully-connected layers on top of it, each with 256, 256, 32, and one neurons. The propagation rule of the layers is defined by

$$\mathbf{h} = \mathrm{ReLU}(\mathbf{W}\mathbf{x} + b), \tag{12}$$

where $\mathbf{x}$ and $\mathbf{h}$ are input and output data, $\mathbf{W}$ and b are learnable layer parameters.

This adversarial training purpose is achieved by a tailor-made training objective function. Let $\mathcal{C}(\cdot, \Theta^{\mathrm{C}})$, $\mathcal{D}(\cdot, \Theta^{\mathrm{D}})$ be the aggregated propagation rules of the encoder–decoder and the discriminator, respectively, where the Θ terms are their respective learnable parameters. Given a time-series $\mathbf{x}$, GA^2 adopts the following objective function for the minimax game:

$$\min_{\Theta^{\mathrm{C}}} \max_{\Theta^{\mathrm{D}}} \mathbb{E}_{\mathrm{P}_{\mathbf{x}}}[\log \mathcal{D}(\mathbf{x}, \Theta^{\mathrm{D}})] + \mathbb{E}_{\mathrm{P}_{\mathbf{x}}}[\log(1 - \mathcal{D}(\mathbf{x} \| \mathcal{C}(\mathbf{x}, \Theta^{\mathrm{C}}), \Theta^{\mathrm{D}}))], \tag{13}$$

where $\mathrm{P}_{\mathbf{x}}$ is the intrinsic data distribution of $\mathbf{x}$. When optimizing the learnable parameters Θ^{C} and Θ^{D}, the optimizer first tunes the encoder–decoder ($\min_{\Theta^{\mathrm{C}}}$) to generate predictions ($\mathcal{C}(\mathbf{x}, \Theta^{\mathrm{C}})$) and compromise the discriminator ($\mathcal{D}(\mathbf{x} \| \mathcal{C}(\mathbf{x}, \Theta^{\mathrm{C}}), \Theta^{\mathrm{D}}) \rightarrow 1$). Subsequently, the discriminator is adjusted ($\max_{\Theta^{\mathrm{D}}}$) so that the generated predictions and ground truth time-series can be distinguished, i.e., $\mathcal{D}(\mathbf{x}, \Theta^{\mathrm{D}}) \rightarrow 1$ and $\mathcal{D}(\mathbf{x} \| \mathcal{C}(\mathbf{x}, \Theta^{\mathrm{C}}), \Theta^{\mathrm{D}}) \rightarrow 0$, respectively. This process imitates the two participating players in a minimax game.

4. GA^2-based citywide traffic speed prediction

In the previous section, we introduced a new GA^2 model for non-Euclidean-structured data prediction task. In the meantime, citywide traffic speed prediction, due to its own characteristics, cannot directly employ GA^2. In this section we discuss the implementation of GA^2 in road network speed prediction and the respective training method.

One may note that in GA^2, data features concentrate on the graph nodes. The edges mostly provide only unweighted connectivity information of nodes, which does not accord with the properties of speed prediction. This issue can be easily resolved by transforming $\mathrm{G}(\mathcal{N}, \mathcal{E})$ into a new undirected road-connectivity graph $\mathrm{H}(\mathcal{E}, \mathcal{C})$, where[3]

$$\mathcal{C} = \{(e_1, e_2) \in \mathcal{E} \times \mathcal{E} | \{V^0_{e_1}, V^*_{e_1}\} \cup \{V^0_{e_2}, V^*_{e_2}\} \neq \emptyset\}. \tag{14}$$

[2] We use 1 to denote ground truth and 0 to denote generated data, both asserted by the discriminator. This definition will be reviewed in Section 4.

[3] Note that bi-directional roads are considered as two nodes in the new graph H.

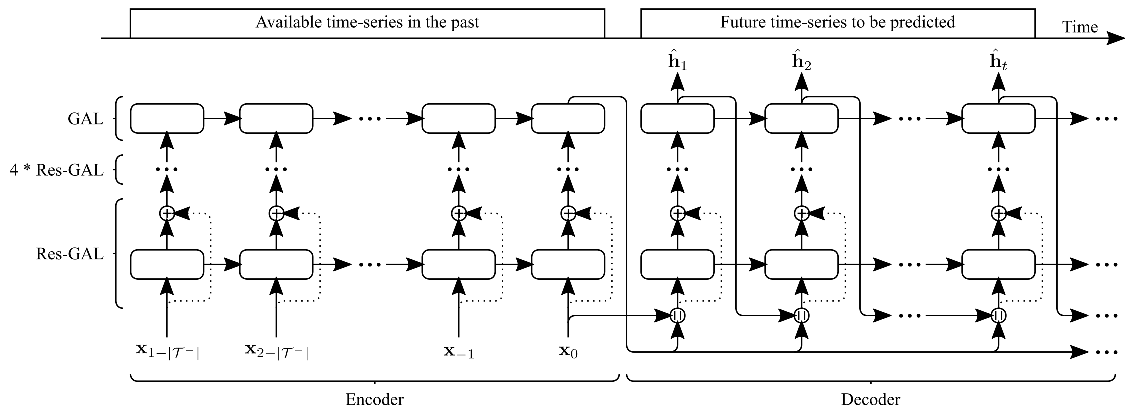


Fig. 3. Data flow of the proposed SEQ2SEQ-based graph-data predictor. Circled plus symbol denotes identity mapping [43], and circled double vertical line symbol denotes concatenation operation along the time axis.

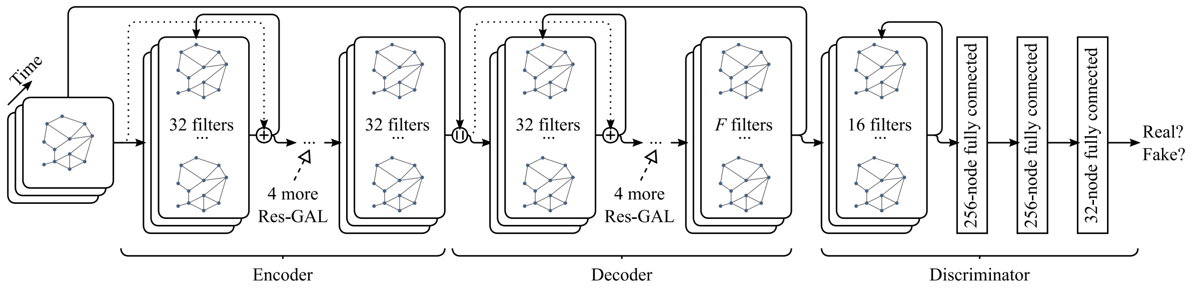


Fig. 4. Layered architecture of the proposed GA^2.

This transformation is inspired by the nature of traffic flow in road networks, where the speed of an arbitrary road is closely correlated with its connecting roads. In addition, the attention mechanism on the connectivity of $\mathrm{H}(\mathcal{E}, \mathcal{C})$ is capable of further identifying the more influential roads among all neighborhoods, which also accords with the intuition [47].

To construct the input data features for urban road networks, we follow a previous work [17] and employ six road features to construct $\mathbf{x}_t$, namely, (1) averaged road traffic speed, (2) traffic speed limit, (3) road length, (4) road width, (5) number of lanes, and (6) number of point-of-interests (PoI) nearby. The latter five properties of road e can be summarized with the previously defined symbol $\mathcal{S}_e$. Consequently, $\mathbf{x}_t = \{\mathcal{V}_t\}\|\{\mathcal{S}_e|\forall e \in \mathcal{E}\}$.

With the constructed input time-series, GA^2 can be trained using typical neural network optimizers, e.g., RMSProp. In traditional generative adversarial models like [46], the discriminator is activated by a sigmoid function at the end to limit the output to $(0, 1)$. Nonetheless, such activation typically leads to model collapse due to the intrinsic gradient vanishing problem with respect to Θ^{G} [48]. In addition, the convergence of sigmoid-activated adversarial model cannot be easily observed. In this work, we adopt the design principle of Wasserstein generative adversarial network (W-GAN) [48] to reformulate the adversarial training objective (13) so as to overcome these issues. In particular, the original design of GAN tries to optimize both Kullback–Leibler divergence and Jensen–Shannon divergence at the same time [48], which may not converge with imperfect discriminator. As an alternative, the Wasserstein (or Earth-Mover) distance can be employed to re-construct the training objective function as follows [48]:

$$W(\mathrm{P}_{\mathbf{x}}, \mathrm{P}_{\mathbf{h}}) = \inf_{\gamma\sim\prod(\mathrm{P}_{\mathbf{x}},\mathrm{P}_{\mathbf{h}})} \mathbb{E}_{(\mathbf{x},\mathbf{h}\sim\gamma)}\|\mathbf{x}-\mathbf{h}\|, \tag{15}$$

where $\mathrm{P}_{\mathbf{h}}$ is the data distribution of the predicted data by the encoder–decoder. By Kantorovich–Rubinstein duality theorem [49], the intractable infimum term can be transformed into

$$W(\mathrm{P}_{\mathbf{x}}, \mathrm{P}_{\mathbf{h}}) = \frac{1}{K} \sup_{\|\mathcal{F}\|_L<K} \mathbb{E}_{\mathbf{x}\sim\mathrm{P}_{\mathbf{x}}}\mathcal{F}(\mathbf{x}) - \mathbb{E}_{\mathbf{h}\sim\mathrm{P}_{\mathbf{h}}}\mathcal{F}(\mathbf{h}), \tag{16}$$

where $\mathcal{F}(\cdot)$ is a Lipschitz continuous function with constant K, i.e., $|\mathcal{F}(x_1) - \mathcal{F}(x_2)| \leq K|x_1 - x_2|$. If we relax the limit of discriminator output from $(0, 1)$ and let $\mathcal{D}(\cdot, \Theta^{\mathrm{D}})$ satisfies 1-Lipschitz continuity, we have

$$W(\mathrm{P}_{\mathbf{x}}, \mathrm{P}_{\mathbf{h}}) = \max_{\|\mathcal{D}\|_L<1} \mathbb{E}_{\mathbf{x}\sim\mathrm{P}_{\mathbf{x}}}\mathcal{D}(\mathbf{x}, \Theta^{\mathrm{D}}) - \mathbb{E}_{\mathbf{h}\sim\mathrm{P}_{\mathbf{h}}}\mathcal{D}(\mathbf{h}, \Theta^{\mathrm{D}}). \tag{17}$$

Substituting $\mathbf{h}$ with $\mathcal{C}$, we have the Wasserstein distance objective function for GA^2:

$$\min_{\Theta^{\mathrm{G}}} \max_{\|\mathcal{D}\|_L<1} \mathbb{E}_{\mathbf{x}\sim\mathrm{P}_{\mathbf{x}}}\mathcal{D}(\mathbf{x}, \Theta^{\mathrm{D}}) - \mathbb{E}_{\mathbf{x}\sim\mathrm{P}_{\mathbf{x}}}\mathcal{D}(\mathcal{C}(\mathbf{x}, \Theta^{\mathrm{G}}), \Theta^{\mathrm{D}}). \tag{18}$$

In order to utilize (18) as the objective function, $\mathcal{D}$ must be 1-Lipschitz continuous. According to spectral normalized GAN (SN-GAN) [50], a neural network propagation rule satisfies this continuity if all learnable parameters are normalized by their respective spectral norm $\sqrt{\lambda_1}$, i.e., the square root of largest eigenvalue of $\mathbf{W}^{\mathsf{T}}\mathbf{W}$ for parameter matrix $\mathbf{W}$. By power iteration, $\sqrt{\lambda_1}$ can be approximated by $\tilde{u}^{\mathsf{T}}\mathbf{W}\tilde{v}$, where $\tilde{u}$ and $\tilde{v}$ can be obtained iteratively by [50]

$$\tilde{v} = \frac{\mathbf{W}^{\mathsf{T}}\tilde{u}}{\|\mathbf{W}^{\mathsf{T}}\tilde{u}\|_2}, \quad \tilde{u} = \frac{\mathbf{W}^{\mathsf{T}}\tilde{v}}{\|\mathbf{W}^{\mathsf{T}}\tilde{v}\|_2}. \tag{19}$$

With a randomly initialized $\tilde{u}_0$, $\sqrt{\lambda_1}$ can be updated iteratively. We combine this iteration with the update of all parameters in both $\mathcal{C}$ and $\mathcal{D}$ during the training process to reduce computation complexity. This means that whenever any learnable parameter in GA^2 is updated, (19) is calculated once and the new $\sqrt{\lambda_1}$ value is used to normalize the respective learnable parameter.

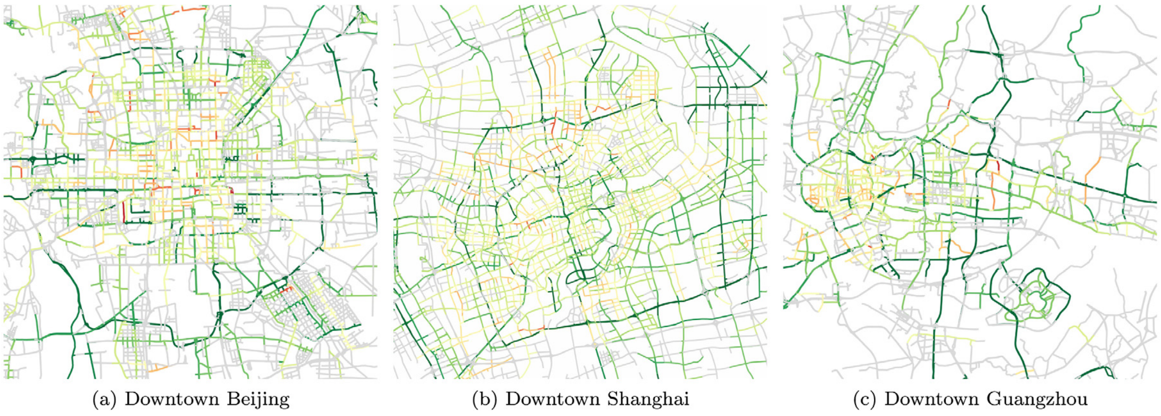


Fig. 5. Urban road network for downtown Beijing, Shanghai, and Guangzhou, with the average traffic speed at 8:00 January 2nd, 2019 plotted. Green stands for high traffic speed, yellow stands for low traffic speed, and red stands for congestion ($\leq 5\,\mathrm{km\,h^{-1}}$). Roads in gray are those without data. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Given a collection of historical road network speed time-series, the learnable parameter tuning is conducted offline. Each time-series of length $T = |\mathcal{T}^-|$, i.e., $\mathbf{x} = \{\mathcal{V}_{1-T}, \dots, \mathcal{V}_{-1}, \mathcal{V}_0\}$ is employed to create $T-1$ training samples. For the ith sample ($i < T$) among them, sub-series $\{\mathcal{V}_{1-T}, \dots, \mathcal{V}_{i-T}\}$ is fed into the encoder–decoder network to generate predictions $\{\hat{\mathcal{V}}_{i-T+1}, \dots, \hat{\mathcal{V}}_0\}$. Subsequently, the ground truth and predicted time-series are both input into the discriminator to calculate

$$\mathcal{D}(\{\mathcal{V}_{1-T}, \dots, \mathcal{V}_0\}, \Theta^{\mathrm{D}}) \\ -\mathcal{D}(\{\mathcal{V}_{1-T}, \dots, \mathcal{V}_{i-T}, \hat{\mathcal{V}}_{i-T+1}, \dots, \hat{\mathcal{V}}_0\}, \Theta^{\mathrm{D}}) \tag{20a}$$

according to (18). Similarly, the following expression is evaluated for later updating the encoder–decoder parameters:

$$\sum_{t\in\mathcal{T}^-} \|\mathcal{V}_t - \hat{\mathcal{V}}_t\|_2^2 \\ -\mathcal{D}(\{\mathcal{V}_{1-T}, \dots, \mathcal{V}_{i-T}, \hat{\mathcal{V}}_{i-T+1}, \dots, \hat{\mathcal{V}}_0\}, \Theta^{\mathrm{D}}), \tag{20b}$$

in which the first term is the L2-norm of the prediction error. The second term of (20b) is used to penalize the objective function considering discriminator output: if the prediction is regarded as authentic ground truth by the discriminator, the prediction is more realistic and this penalty term is minimized. This design follows the principle of adversarial training [46]. For every 16 samples (mini-batch size), (20) is evaluated and averaged over the population, and we employ the RMSProp optimizer (maximize (20a), minimize (20b)) and updated $\sqrt{\lambda_1}$ to update all learnable network parameters. This calculation iterates with new samples until all time-series in the training data set is utilized, which is recorded as an epoch. The whole process terminates when the objective value of (20a) at the end of each epoch converges, and the parameters are frozen for online speed prediction.

With a well-trained GA^2, the online speed prediction only employs the encoder–decoder network as shown in Fig. 3. Similar to the training process, the available road network speed time-series is sequentially input into the network to develop the prediction of the next immediate time instance. Recall that the decoder actually outputs all six road features instead of only the speed. We replace the other five predicted road features with their ground truth in $\mathcal{S}_e$, and the new data is subsequently fed into the network. This iteration can produce data on the infinite prediction horizon, and system operator may decide the termination considering requirements in practice.

5. Case studies

In this work, we propose GA^2 to predict citywide traffic speed based on historical data. To fully evaluate the performance of the proposed model, we carry out three comprehensive case studies with real world urban road networks and their historical data. In particular, we first assess the speed prediction accuracy and system computation time, and compare the performance with a wide range of existing generative models and tailor-made techniques. Then, we perform a model structure search to examine the influence of architectural changes on the system performance, and identify the best performing structure. Finally, we investigate the performance on historical data points with sampling noise and different sampling frequencies. All case studies are performed on a DGX-2 instance equipped with Nvidia Tesla V100 GPUs for parallel computing acceleration, and GA^2 is implemented with PyTorch [51]

5.1. Data sets and settings

We adopt the real-world traffic data at three major cities in China for investigation, namely, Beijing (BJ), Shanghai (SH), and Guangzhou (GZ). Two types of data – urban road network topology and traffic speed data – are collected and fused to construct the citywide traffic speed data set. In particular, road topologies of the three cities are developed from OpenStreetMap, whose downtown networks are presented in Fig. 5. The traffic speed data are obtained from NavInfo Traffic Data platform,[4] and the data from 8:00 AM January 1st, 2019 to 7:55 AM February 1st, 2019 (referred to as data time horizon in the sequel) are employed in this work. The data comprise averaged traffic speed of approximately 1400, 1500, and 1000 roads in the respective three cities, and the sampling interval is 5 min. Consequently, there are 8928 time instances in the data set, each of which corresponds to urban road speed maps of the three cities. These data are the ground truths in the case studies. When fed into GA^2, speed values are normalized using Z-score normalization. As not all roads in the three cities are observed in the data set, the respectively missing speed values are encoded by zeros after input data normalization, which is a common practice in missing data recovery with deep learning techniques [17,52].

Based on the ground truths, we further construct the training and testing samples for GA^2. As the model accepts variable-length time-series of graph-structured data as inputs, these samples can

[4] http://nitrafficindex.com/.

be developed with a rolling manner. Starting from the first time instance in the time horizon, i.e., 8:00 AM January 1st, 48 time-series samples are created with the same starting and different stopping times. For the ith sample, $2i$ time instances are included. The first i time instances are considered as $\mathcal{T}^-$, and the remainder is $\mathcal{T}^+$ in this sample. This process repeats for all possible starting time instance until 7:55 AM February 1st, and samples with time instances not included in the data set are discarded. As a result, 426 240 samples are generated for each city.

For cross validation, the samples are grouped into two non-overlapping sets, i.e., a training set and a testing set. All samples with the starting time before 8:00 AM January 15th, 2019 and after 7:55 AM January 23th, 2019 are utilized to train GA^2, totally 317 952 samples. The rest 108 288 are employed to test the system performance after the model is well trained. When testing the system, no matter how large $\mathcal{T}^-$ is for each sample, speed predictions of $|\mathcal{T}^+| = 48$ future time instances are developed in order to evaluate the impact of input time-series length on the performance.

5.2. Prediction accuracy

The accuracy of predicted citywide traffic speed data compared with the ground truths is among the principal performance metrics in investigating the efficacy of traffic speed prediction techniques. In this test, we train three independent GA^2 models using the training data of the three cities, and compare the accuracy with other speed prediction techniques in terms of MAE, MAPE, and RMSE.

We first compare GA^2 with widely-adopted time-series regression techniques and state-of-the-art traffic speed prediction models as follows:

- History Average (HA): HA models traffic speed dynamics as a seasonal process. Weighted average of data in previous seasons are utilized as the prediction. In this test, we adopt a season with four weeks, and the time period is one week. For instance, to predict the speed of road A at 8:00 AM on an arbitrary Monday, the speed values at the same time of four previous Mondays are averaged as the prediction.
- ARIMA with Kalman filter (AK): AK considers a stationary regression-type time-series and makes use of the dependent relationship between a data point and some number of lagged ones. In this test, we consider three lagged data points in the time-series, zero degree of differencing, and the moving average window size is one.
- Linear support vector regression (SVR): SVR maps time-series data into higher dimensional kernel-induced feature spaces for subsequent linear regression. In this test, we employ a penalty term $C = 0.1$ for SVR, and the number of historical data points is 5. Gaussian kernel function is adopted.
- Bayesian-connected LSTM (B-LSTM) [53]: B-LSTM is an encoder–decoder framework similar to Fig. 3 constructed with only LSTM cells. Both encoder and decoder have two LSTM layers, which are fully-connected in all time-steps. The model is trained with mini-batch size 128 and loss function MAE. Other settings of B-LSTM follows the literature.
- Diffusion convolutional recurrent neural network (DCRNN) [14]: DCRNN extracts the spatial data-correlation with bi-directional random-walk on the road network, and the temporal correlation with scheduling sampling and encoder–decoder structure for time-series prediction.
- Temporal Graph Convolutional Network (T-GCN) [24]: T-GCN employs gated recurrent units and GCN to capture both the temporal and spatial data characteristics for traffic speed prediction.
- Attention Temporal Graph Convolutional Network (A3T-GCN) [54]: A3T-GCN is among the latest state-of-the-art approach for traffic prediction, which introduces the attention mechanism into GCN to adjust the importance of timestamps and incorporate global temporal information to improve prediction accuracy.

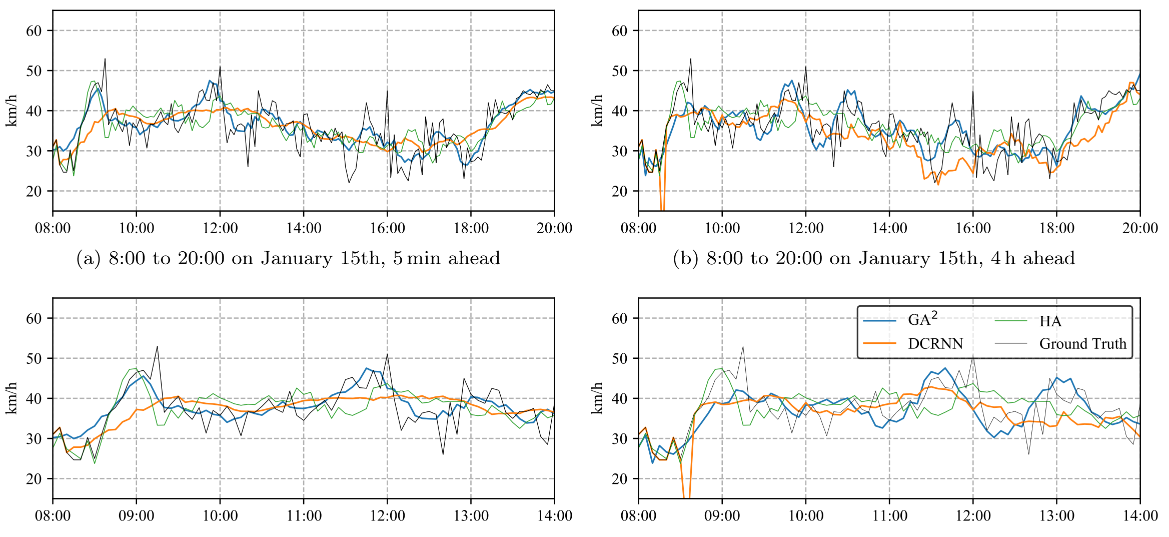
For the baseline approaches, we adopt publicly available source code for DCRNN, T-GCN, and A3T-GCN with minuscule noncritical adaptive changes. Other baselines are implemented with Python. The baselines are selected as they are either the best-performing naïve and parametric ones, or are the state-of-the-art deep learning-based non-parametric models. The same data is employed for all models in accordance with Section 5.1. As HA makes use of the speed data before data time horizon, additional data from December 2018 is included for this specific technique. In addition, all compared models except for DCRNN, T-GCN, and A3T-GCN can only deal with one road each time. To avoid excessive computation, we randomly select 50 roads from each of the three cities and test the prediction performance on them. Consequently, 50 independent AK, SVR and B-LSTM models are trained for each city and the performance is evaluated based on their averaged metrics. All other case study configurations are kept identical.

Table 1 presents the comparison of different speed prediction approaches for 5 min, 15 min, 1 h, and 4 h ahead forecasting on all three investigating cities using the three accuracy metrics in (2) with the 108 288 testing samples. The best performing results are in bold font. The performance comparison clearly indicates that GA^2 outperforms other techniques in the majority of test cases. The only exception is 4 h ahead forecasting in Shanghai and Guangzhou where HA slightly overmatch others. This is due to the characteristics of HA which directly produces raw data points based on historical data samples instead of a time-series, therefore relaxing the technique from accumulating prediction errors during forecasting. Nonetheless, this relaxation is a double-edged sword: the short-term prediction accuracy (≤ 1 h) is sacrificed as HA cannot utilize immediate data in the recent past. In this context, GA^2 greatly benefits from its spatial–temporal feature extraction capability to develop more precise predictions than others. Furthermore, the comparison between GA^2 and B-LSTM also indicates the importance of network topologies in traffic speed prediction problem. The major difference between these two approaches is that GA^2 uses graph neural networks, in particular graph attention mechanism, to replace the Bayesian network counterpart in B-LSTM. The performance improvement can be credited to the inclusion of topological information into the computation process, i.e., (10). While DCRNN, T-GCN, and A3T-GCN approaches GA^2 in the comparison, the unique generative model design and residual links improve the model capacity of the deep learning architecture GA^2 employs. From these comparisons, we derive rules of thumb to guide future speed prediction model design:

- Statistical learning-based approaches work better for near-future prediction, while long-term prediction still relies on time-invariant forecasts – namely, HA and variants – until prediction error accumulation issue were resolved for learning approaches. This is a promising future research direction.
- When designing short-term prediction models, generative models work better than other current model design paradigm of deep learning. This will be further justified in Section 5.3. The inclusion of transportation network topology information is another significant prediction accuracy booster.

Table 1
Performance comparison for citywide traffic speed prediction.

		MAE				MAPE				RMSE			
		5 min	15 min	1 h	4 h	5 min	15 min	1 h	4 h	5 min	15 min	1 h	4 h
BJ	GA^2	**2.040**	**2.384**	**3.175**	**4.465**	**6.0%**	**6.9%**	**9.2%**	**12.7%**	**2.866**	**3.629**	**4.434**	6.661
	HA	4.730	4.730	4.730	4.730	13.7%	13.7%	13.7%	13.7%	6.670	6.670	6.670	6.670
	AK	3.603	4.171	5.626	8.294	10.4%	12.3%	16.3%	24.3%	5.032	5.866	8.048	11.405
	SVR	3.302	3.937	5.294	7.711	9.6%	11.3%	15.3%	22.4%	4.596	5.673	7.676	10.734
	B-LSTM	2.757	3.126	3.886	4.986	8.0%	9.1%	11.4%	15.0%	3.891	4.270	5.461	6.730
	DCRNN	2.410	2.769	3.760	4.590	7.0%	8.1%	10.7%	13.3%	3.431	3.781	5.218	**6.465**
	T-GCN	2.552	2.713	3.824	4.840	7.4%	8.0%	10.9%	14.5%	3.497	3.636	5.277	6.618
	A3T-GCN	2.250	2.589	3.669	4.547	7.0%	7.6%	10.6%	14.0%	3.060	3.539	5.121	6.506
SH	GA^2	**2.159**	**2.586**	**3.450**	4.710	**6.4%**	**7.5%**	**10.2%**	**13.4%**	**2.954**	**3.566**	**4.761**	6.703
	HA	4.548	4.548	4.548	**4.548**	13.4%	13.4%	13.4%	**13.4%**	6.065	6.065	6.065	**6.065**
	AK	3.354	4.424	5.737	8.410	9.9%	12.8%	16.9%	24.4%	4.585	6.160	7.752	12.399
	SVR	3.429	4.222	4.897	7.561	10.0%	12.2%	14.6%	21.9%	4.832	5.876	6.738	10.678
	B-LSTM	3.023	3.197	4.300	5.258	8.9%	9.4%	12.4%	15.4%	4.128	4.391	6.194	7.140
	DCRNN	2.632	2.856	3.830	5.251	7.7%	8.3%	11.1%	15.0%	3.598	4.012	5.447	7.419
	T-GCN	2.584	2.798	3.740	4.917	7.5%	8.2%	10.8%	13.9%	3.382	3.884	5.132	6.614
	A3T-GCN	2.397	2.642	3.706	4.746	7.3%	8.0%	10.5%	13.7%	3.194	3.592	5.065	6.454
GZ	GA^2	**2.144**	**2.426**	**3.169**	4.665	**6.3%**	**7.2%**	**9.3%**	13.6%	**2.983**	**3.271**	**4.478**	6.447
	HA	4.314	4.314	4.314	**4.314**	12.8%	12.8%	12.8%	**12.8%**	5.883	5.883	5.883	**5.883**
	AK	3.407	4.283	5.748	8.083	10.0%	12.3%	16.4%	23.3%	4.750	6.368	8.298	11.168
	SVR	3.108	3.965	5.180	7.428	9.0%	11.6%	15.3%	21.6%	4.413	5.462	7.064	10.112
	B-LSTM	2.965	2.977	3.932	5.120	8.6%	8.8%	11.6%	14.9%	4.154	4.032	5.395	7.364
	DCRNN	2.464	2.745	3.802	4.765	7.2%	8.1%	11.2%	14.0%	3.351	3.817	5.271	6.605
	T-GCN	2.333	2.681	3.547	4.715	6.9%	8.0%	10.4%	13.8%	3.133	3.654	4.899	6.427
	A3T-GCN	2.254	2.704	3.324	4.635	6.9%	7.7%	10.5%	13.7%	3.115	3.548	5.090	6.305



(a) 8:00 to 20:00 on January 15th, 5 min ahead
(b) 8:00 to 20:00 on January 15th, 4 h ahead
(c) 8:00 to 14:00 on January 15th, 5 min ahead
(d) 8:00 to 14:00 on January 15th, 4 h ahead

Fig. 6. 5 min and 4 h ahead predicted speed time-series of West Chang'an Avenue in Beijing by GA^2, DCRNN, and HA.

- Similar to related research, including residual links drives the neural network to train smoothly without pre-maturely getting trapped in local optimum of parameter searching space [22,43]. This is especially helpful for traffic prediction with the ever-increasing size of deep learning-based models in term of trainable network parameters.

To better illustrate the characteristics of this model, we visualize the traffic speed prediction result of West Chang'an Avenue in Beijing with GA^2 in Fig. 6, and compare the predictions with DCRNN, HA, and the ground truth data. Both 5 min and 4 h ahead predictions are presented. Generally speaking, both GA^2 and DCRNN demonstrate better forecasting accuracy when the ahead time is small. At the same time, HA is independent of the time, thus shows the same result in either scenarios. A careful investigation in the predicted traffic speed data by GA^2 and DCRNN reveals more insights. Compared with GA^2, DCRNN fails to capture the local speed fluctuations within the prediction time horizon. This indicates that GA^2 is more likely to accurately predict sudden changes, and DCRNN provides a smoothened speed profile like moving average. This is because that GA^2 explicitly makes use of the network adjacency information in the learning process, which contributes to more accurate predictions. The illustration demonstrates the importance of accurate road network topology on traffic speed prediction problem, as DCRNN embeds adjacency information through a random-walk process, which cannot explicitly presents the topological information. Furthermore, it can be observed from the ground truth plot in Fig. 6 that the traffic speed is highly unstable and stochastic within short periods ($<$ 1 h), which is possibly contributed by local meteorology, dynamic traffic events, etc. Without such data, it is unlikely that prediction approaches can perfectly fit all speed fluctuations. Properly integrating these information into deep learning models to better calibrate the prediction is a possible future research topic.

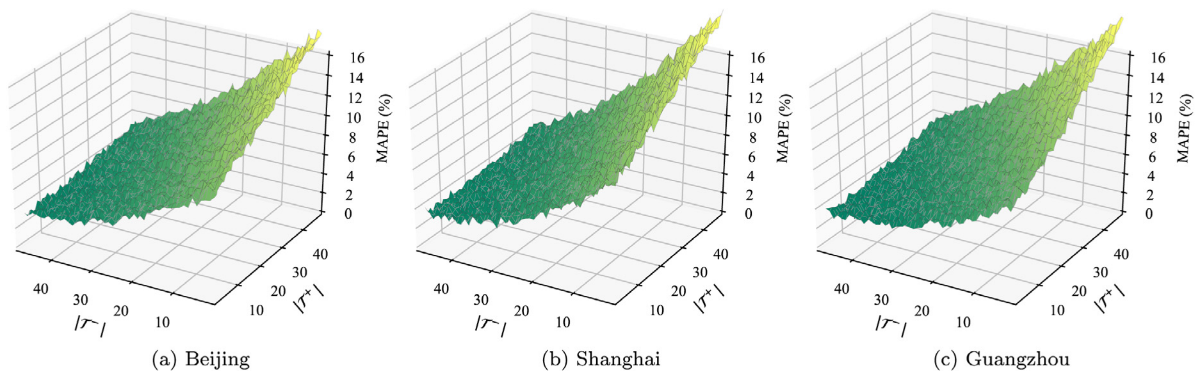


Fig. 7. Impact of $|\mathcal{T}^-|$ and $|\mathcal{T}^+|$ on prediction MAPE.

Table 2
Training and prediction time of GA^2.

	Time consumed		
	Training	Prediction (5 min)	Prediction (4 h)
BJ	20.12 h	0.0152 s	0.0700 s
SH	31.49 h	0.0192 s	0.0955 s
GZ	12.14 h	0.0095 s	0.0414 s

Table 3
Accuracy MAPE comparison of GA^2 variants.

	5 min	15 min	1 h	4 h
GA^2	**6.0%**	**6.9%**	**9.2%**	**12.7%**
GA^2-GC	6.7%	7.5%	9.6%	13.1%
GA^2-NR	7.2%	8.2%	10.4%	14.3%
GA^2-ED	7.4%	8.1%	10.3%	13.9%
GA^2-ND	6.5%	7.3%	9.9%	13.4%
GAL	7.7%	8.4%	10.5%	14.7%

Fig. 7 summarizes the impact of $|\mathcal{T}^-|$ and $|\mathcal{T}^+|$ on MAPE of the forecast. The former represents the real-time data available in the immediate past, and the latter is the length of prediction horizon. A general rule of thumb can be obtained from the figure, i.e., larger $|\mathcal{T}^-|$ results in better MAPE in the near future, and the performance degrades when the prediction horizon length increases. The reason is intuitive: larger $|\mathcal{T}^+|$ renders more historical information, which can better resemble the evolving dynamics of road traffic. At the same time, it is inevitable that errors are made during the data prediction process, which accumulates with a larger $|\mathcal{T}^-|$, resulting in worst prediction performance. This also accords with the statistics presented in Table 1 and general intuition.

Last but not least, the training and prediction time required by GA^2 on a single Tesla V100 are presented in Table 2. While training the parameters in GA^2 requires a significant amount of time, the process is typically conducted in an offline manner. The fine-tuned parameters are then utilized in the swift online prediction. Considering that the predictions are made with a granularity of five minutes, the less than 0.1 s forecasting time can be safely regarded as real-time. As a reference, DCRNN requires approximately eleven hours to get trained on BJ data, and develops a prediction with less than 0.1 s. Although GA^2 requires almost twice the training time, the offline nature of training makes it still acceptable considering the fast online prediction.

When comparing with the other prediction techniques, GA^2 demonstrates unique merits. In contrast to HA, AK, SVR, and B-LSTM which handles one road for each model, GA^2 develops data for all roads in one pass. This drastically reduces the prediction time, as others require more than 1000 independent regressors to achieve the same objective. While DCRNN, T-GCN, and A3T-GCN can also handle all roads with one model, the intensive

Table 4
Hyper-parameter configurations of GA^2 variants.

	Res-GAL (SEQ2SEQ)		Discriminator	
	Layers	Filters	GAL filters	Fully-connected
GA^2	5	32	16	256 + 256 + 32
GA^2-A	3	32	16	256 + 256 + 32
GA^2-B	7	32	16	256 + 256 + 32
GA^2-C	5	16	16	256 + 256 + 32
GA^2-D	5	64	16	256 + 256 + 32
GA^2-E	5	32	8	256 + 256 + 32
GA^2-F	5	32	32	256 + 256 + 32
GA^2-G	5	32	16	256 + 64 + 32
GA^2-H	5	32	16	256 + 256 + 128

Table 5
Accuracy MAPE and training time of GA^2 variants.

	Time ahead				Training time
	5 min	15 min	1 h	4 h	
GA^2	**6.0%**	**6.9%**	**9.2%**	**12.7%**	20.12 h
GA^2-A	6.2%	7.2%	10.3%	14.5%	18.63 h
GA^2-B	6.1%	**6.9%**	9.3%	12.9%	37.12 h
GA^2-C	**6.0%**	7.2%	9.7%	14.1%	**14.60 h**
GA^2-D	**6.0%**	**6.9%**	9.3%	13.1%	57.21 h
GA^2-E	6.1%	7.0%	9.8%	13.7%	22.19 h
GA^2-F	**6.0%**	**6.9%**	**9.2%**	**12.7%**	35.32 h
GA^2-G	6.1%	7.0%	9.4%	12.9%	20.23 h
GA^2-H	**6.0%**	**6.9%**	9.3%	**12.7%**	22.51 h

computational requirement leads to more than 50 h of training time, doubling that of GA^2. To conclude, GA^2 can develop accurate citywide traffic speed predictions in real-time, and outperforms compared techniques.

5.3. GA^2 architecture

In this work, the encoder–decoder architecture is adopted, in which the graph attention mechanism and residual links are among the fundamental components. In addition, GA^2 utilizes a generative adversarial model for adversarial training, and the discriminator is expected to contribute to performance improvement. In this sub-section, we first investigate the influence of these components to the overall prediction accuracy. Four new GA^2 variants are constructed as follows:

- GA^2-GC: The typical GCN is adopted to replace the graph attention mechanism in Fig. 4. Other components remain unchanged, as is the hyper-parameters, e.g., number of layers and neurons, etc.
- GA^2-NR: All residual links are removed from Fig. 4.
- GA^2-ED: The decoder is removed from Fig. 4. The last layer in the encoder has F filters instead of 32.

- GA^2-ND: The discriminator is removed from Fig. 4. The new network is trained by minimizing the L1 loss.
- GAL: Only one layer of GAL is employed to perform the prediction.

Table 3 presents the MAPE of prediction accuracy with the three GA^2 variants on the Beijing data set. The result implies a straightforward conclusion — graph attention mechanism, residual links, SEQ2SEQ structure, and adversarial training all contribute positively to data forecasting accuracy in the tested scenario. Compared with GCN, GAL employs the attention mechanism to explicitly assign an independent weight to each of connected roads, which follows the intuition that multiple incoming and outgoing flows should have different impact on the traffic speed. In addition, the residual links help the network to skip layers deemed unnecessary for specific samples in the architecture. This greatly reduces the training difficulty for convergency, thus in turn leads to a better performance. The decoder is critical in interpreting the encoded message developed by the encoder, and removing this data processing block significantly reduces the deep learning model capacity, rendering inferior results. Finally, discarding the discriminator indicates that instead of learning the statistical distribution of ground truth geometry data, data points in samples are directly learnt, rendering a network sensitive to noise and outliers.

Besides the constituting components, the hyper-parameters of GA^2 are other critical factors that impact the system performance. We specifically assess the influence of layer and neuron numbers by considering hyper-parameter configurations as demonstrated in Table 4. All GA^2 variants are trained with the same data set of Beijing. The MAPE of prediction accuracy and training time is presented in Table 5. From the results, a series of conclusions can be made. First, while removing Res-GAL from SEQ2SEQ has a negative impact on the accuracy, including more layers requires significantly more training time without better performance. This can be observed by comparing GA^2 with GA^2-A and GA^2-B, and the result suggests an adequate number of layers in the encoder and decoder to make a trade-off between training speed and system performance. Furthermore, it seems that reducing the number of filters leads to under-performing architectures (GA^2-C and GA^2-E), and increasing the number cannot further improve the performance (GA^2-D). Finally, alternating the discriminator configurations does not have significant influence on the forecast accuracy. The training time, though, may drastically increase with the GAL filter size if it is more than 16. Comparing Tables 1 and 5, we have an observation that while hyper-parameter setting has influence on GA^2 performance, the deviation is not as significant as the improvement over baseline approaches. This indicate that hyper-parameter optimization is recommended but not essential in developing GA^2 to make it supersede existing approaches.

5.4. Data noise and sample frequency

In the previous case studies, the real-world data samples of urban networks are employed. While the data may contain sampling errors, the ground truth values are not available. Nonetheless, as noise is inevitable during the sampling process, we are interested in how the proposed GA^2 perform with only noisy data input. In this case study, we manually introduce different levels of noise based on the Beijing data set. The new sets are employed to train GA^2, whose corresponding predictions are compared with the original data set for performance evaluation. Specifically, we assume that the sampling noise is Additive white Gaussian noise (AWGN).[5] New sets with Gaussian standard errors of 1%, 2%, 3%, . . . , 10% on the original Beijing data are constructed, whose average deviations are therefore 0.5%, 1%, 1.5%, . . . , 5% of the ground truth, respectively. Ten GA^2 models are trained independently with the new noisy sets, and the accuracy MAPE results are depicted in Fig. 8. The results follow the intuition that increasing noise level generally leads to larger MAPE, as noise patterns cannot be learnt by GA^2 unlike noise probability distributions. In the meantime, an interesting conclusion can be achieved that compared with short-term predictions (< 1 h), long-term forecast accuracy are less influenced by sampling errors. The observation is credited with the fact that long-term forecast errors are notably larger than short-term ones, rendering the impact of noise easily “hidden” in the existing errors. This also explains another piece of information conveyed by the results that the MAPE increase rate attenuates with the standard error — relatively larger prediction errors hide the noise better.

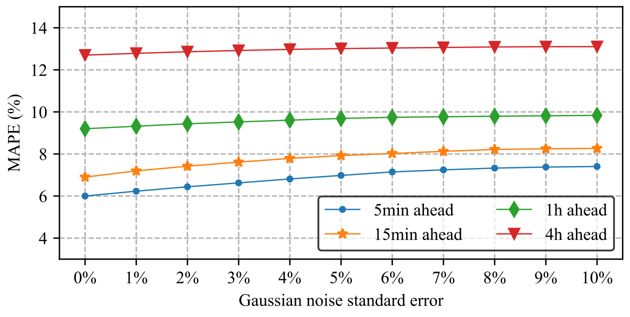


Fig. 8. MAPE of GA^2 models with various levels of sampling error.

Table 6
Accuracy MAPE comparison on different sampling intervals.

	Number of time instances ahead			
	1	3	12	48
BJ (5 min)	6.0%	6.9%	9.2%	12.7%
BJ-10 m	5.9%	6.8%	9.1%	12.6%
BJ-15 m	5.9%	6.8%	9.1%	**12.5**%
BJ-30 m	5.7%	6.7%	**9.0**%	12.7%
BJ-1 h	**5.4**%	**6.5**%	9.1%	12.6%

Last but not least, we investigate whether GA^2 depends on the 5 min sampling interval of the original data set. By removing data points of selected time instances, the original Beijing data set is down-sampled by averaging to new ones with 10 min, 15 min, 30 min, 1 h sampling interval, label by BJ-10m, BJ-15m, BJ-30m, BJ-1h, respectively. The same GA^2 architecture is employed to train the new data sets individually, and the prediction accuracy is presented in Table 6. In this case study, the number of time instances ahead from which GA^2 makes prediction is employed for a fair comparison. For instance, up to 48 time instances (8 h) data is used for making predictions in the future 48 time instances (8 h) in BJ-10 m data set, which is identical to all other data sets except for the absolute length of time. From the results it can be concluded that increasing the sampling interval can improve the prediction accuracy, though the magnitude is not significant. This is because that down-sampling the original data set renders notably smoother and regular speed data dynamics as shown by an example in Fig. 9. Consequently, we can conclude that GA^2 develops stable (approx. 0.5% MAPE difference) forecast performance when handling data sets with different sampling frequencies.

6. Conclusions

In this paper, we propose a new geometric deep learning approach for citywide traffic speed prediction in modern intelligent transportation systems. Different from previous work,

[5] Other noise distributions are tested offline where similar performance of GA^2 is observed.

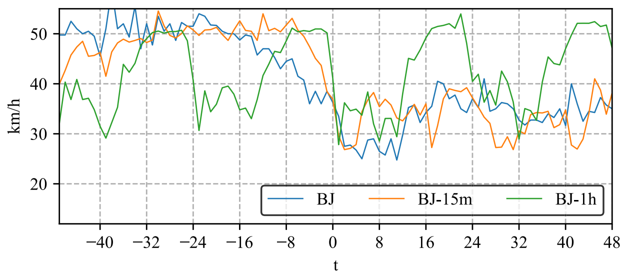


Fig. 9. Traffic speed time-series of West Chang'an Avenue in Beijing with different sampling intervals.

the proposed approach extensively utilizes the road network topology in the learning process to forecast data for every road within the transportation network in one go. The proposed approach employs a tailor-made deep neural network architecture to learn from both the geometric and temporal data characteristics of historical information. The architecture incorporates design principle of recent theoretical advancement in geometric deep learning and attention mechanism, and the adversarial training with spectral normalized design is adopted to further improve the system performance and robustness. To evaluate the efficacy of the proposed approach, a series of comprehensive case studies are conducted based on three data sets of real-world urban road networks in China. The results demonstrate outstanding data prediction accuracy compared with classical and recent statistical learning prediction techniques. Furthermore, we investigate the structural design of the proposed approach via experiments to reveal the best-performing architecture. Finally, we study the sensitivity to data sampling error and interval of the proposed approach.

Declaration of competing interest

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

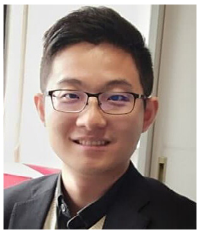
Acknowledgments

This research was supported by the General Program of Guangdong Basic and Applied Basic Research Foundation, China No. 2019A1515011032 and Guangdong Provincial Key Laboratory, China No. 2020B121201001.

References

[1] Y. Lv, Y. Duan, W. Kang, Z. Li, F.Y. Wang, Traffic flow prediction with big data: a deep learning approach, IEEE Trans. Intell. Trans. Syst. 16 (2) (2015) 865–873.
[2] F.Y. Wang, Parallel control and management for intelligent transportation systems: concepts, architectures, and applications, IEEE Trans. Intell. Trans. Syst. 11 (3) (2010) 630–638.
[3] J.J.Q. Yu, A.Y.S. Lam, Autonomous vehicle logistic system: joint routing and charging strategy, IEEE Trans. Intell. Trans. Syst. 19 (7) (2018) 2175–2187.
[4] J. Zhang, F.Y. Wang, K. Wang, W.H. Lin, X. Xu, C. Chen, Data-driven intelligent transportation systems: a survey, IEEE Trans. Intell. Trans. Syst. 12 (4) (2011) 1624–1639.
[5] A.M. Nagy, V. Simon, Survey on traffic prediction in smart cities, Pervasive Mob. Comput. 50 (2018) 148–163.
[6] J.J.Q. Yu, Two-stage request scheduling for autonomous vehicle logistic system, IEEE Trans. Intell. Trans. Syst. 20 (5) (2019) 1917–1929.
[7] X. Yin, G. Wu, J. Wei, Y. Shen, H. Qi, B. Yin, A comprehensive survey on traffic prediction, 2020, arXiv:2004.08555 [eess.SP].
[8] N.L. Nihan, K.O. Holmesland, Use of the Box and Jenkins time series technique in traffic forecasting, Transportation 9 (2) (1980) 125–143.
[9] M. Van Der Voort, M. Dougherty, S. Watson, Combining Kohonen maps with ARIMA time series models to forecast traffic flow, Transp. Res. C 4 (5) (1996) 307–318.
[10] J. Guo, W. Huang, B.M. Williams, Adaptive Kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification, Transp. Res. C 43 (2014) 50–64.
[11] S. Sun, C. Zhang, G. Yu, A Bayesian network approach to traffic flow forecasting, IEEE Trans. Intell. Trans. Syst. 7 (1) (2006) 124–132.
[12] C.M. Queen, C.J. Albers, Intervention and causality: forecasting traffic flows using a dynamic Bayesian network, J. Amer. Statist. Assoc. 104 (486) (2009) 669–681.
[13] H. Yu, Z. Wu, S. Wang, Y. Wang, X. Ma, Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks, 2017, arXiv:1705.02699 [cs.LG].
[14] Y. Li, R. Yu, C. Shahabi, Y. Liu, Diffusion convolutional recurrent neural network: Data-driven traffic forecasting, in: Proc. International Conference on Learning Representations, Vancouver, Canada, Apr. 2018.
[15] C. Xiong, Z. Zhu, X. He, X. Chen, S. Zhu, S. Mahapatra, G.-L. Chang, L. Zhang, Developing a 24-hour large-scale microscopic traffic simulation model for the before-and-after study of a new tolled freeway in the Washington, DC–Baltimore region, J. Transp. Eng. 141 (6) (2015) 05015001.
[16] Y.-C. Chiu, L. Zhou, H. Song, Development and calibration of the anisotropic mesoscopic simulation model for uninterrupted flow facilities, Transp. Res. B 44 (1) (2010) 152–174.
[17] J.J.Q. Yu, J. Gu, Real-time traffic speed estimation with graph convolutional generative autoencoder, IEEE Trans. Intell. Trans. Syst. 20 (10) (2019) 3940–3951.
[18] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016.
[19] R. Yu, Y. Li, C. Shahabi, U. Demiryurek, Y. Liu, Deep learning: a generic approach for extreme condition traffic forecasting, in: Proc. SIAM International Conference on Data Mining, SIAM, 2017, pp. 777–785.
[20] X. Ma, Z. Dai, Z. He, J. Ma, Y. Wang, Y. Wang, Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction, Sensors 17 (4) (2017) 818.
[21] M. Defferrard, X. Bresson, P. Vandergheynst, Convolutional neural networks on graphs with fast localized spectral filtering, in: Proc. Advances in Neural Information Processing Systems, Barcelona, Spain, Dec. 2016.
[22] J. Zhang, Y. Zheng, D. Qi, Deep spatio-temporal residual networks for citywide crowd flows prediction, in: Proc. AAAI Conference on Artificial Intelligence, 2017.
[23] J.J.Q. Yu, W. Yu, J. Gu, Online vehicle routing with neural combinatorial optimization and deep reinforcement learning, IEEE Trans. Intell. Transp. Syst. 20 (10) (2019) 3806–3817.
[24] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-GCN: a temporal graph convolutional network for traffic prediction, IEEE Trans. Intell. Transp. Syst. (2019) 1–11.
[25] S. Guo, Y. Lin, N. Feng, C. Song, H. Wan, Attention based spatial–temporal graph convolutional networks for traffic flow forecasting, in: Proc. AAAI Conference on Artificial Intelligence, Honolulu, HI, Jan. 2019, pp. 922–929.
[26] B.L. Smith, B.M. Williams, R.K. Oswald, Comparison of parametric and nonparametric models for traffic flow forecasting, Transp. Res. C 10 (4) (2002) 303–321.
[27] B. Ghosh, B. Basu, M. O'Mahony, BayesIan time-series model for short-term traffic flow forecasting, J. Trans. Eng. 133 (3) (2007) 180–189.
[28] Y. Kamarianakis, P. Prastacos, Forecasting traffic flow conditions in an urban network: comparison of multivariate and univariate approaches, Transp. Res. Rec. 1857 (1) (2003) 74–84.
[29] W. Min, L. Wynter, Real-time road traffic prediction with spatio-temporal correlations, Transp. Res. C 19 (4) (2011) 606–616.
[30] B.M. Williams, Multivariate vehicular traffic flow prediction: evaluation of ARIMAX modeling, Transp. Res. Rec. 1776 (1) (2001) 194–200.
[31] B. Yu, Y. Lee, K. Sohn, Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network (GCN), Transp. Res. C 114 (2020) 189–204.
[32] Z. Cui, R. Ke, Z. Pu, X. Ma, Y. Wang, Learning traffic as a graph: a gated graph wavelet recurrent neural network for network-scale traffic prediction, Transp. Res. C 115 (2020) 102620.
[33] X. Yang, Y. Yuan, Z. Liu, Short-term traffic speed prediction of urban road with multi-source data, IEEE Access 8 (2020) 87541–87551.
[34] K. Guo, Y. Hu, Z. Qian, H. Liu, K. Zhang, Y. Sun, J. Gao, B. Yin, Optimized graph convolution recurrent neural network for traffic prediction, IEEE Trans. Intell. Transp. Syst. (2020) 1–12.
[35] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: Proc. International Conference on Learning Representations, Vancouver, Canada, 2018.
[36] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: Proc. International Conference on Learning Representations, Toulon, France, Apr. 2017.
[37] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and locally connected networks on graphs, in: Proc. International Conference on Learning Representations, Banff, Canada, Apr. 2014.

[38] R.N. Bracewell, R.N. Bracewell, The Fourier Transform and its Applications, Vol. 31999, McGraw-Hill, New York, 1986.
[39] Z. Zhang, P. Cui, W. Zhu, Deep learning on graphs: A survey, 2018, arXiv:1812.04202 [cs.LG].
[40] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, in: Proc. International Conference on Learning Representations, San Diego, CA, Dec. 2015.
[41] J.F. Kolen, Fool's gold: Extracting finite state machines from recurrent network dynamics, in: Proc. Advances in Neural Information Processing Systems, 1993, pp. 501–508.
[42] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[43] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proc. IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2016, pp. 770–778.
[44] I. Sutskever, O. Vinyals, Q.V. Le, Sequence to sequence learning with neural networks, in: Proc. Advances in Neural Information Processing Systems, 2014.
[45] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, in: Proceedings of the 32nd International Conference on International Conference on Machine Learning, 2015, pp. 448–456.
[46] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Proc. Advances in Neural Information Processing Systems, 2014, pp. 2672–2680.
[47] H. Hu, G. Li, Z. Bao, Y. Cui, J. Feng, Crowdsourcing-based real-time urban traffic speed estimation: from trends to speeds, in: Proc. IEEE International Conference on Data Engineering, Helsinki, Finland, 2016, pp. 883–894.
[48] M. Arjovsky, S. Chintala, L. Bottou, Wasserstein generative adversarial networks, in: Proc. International Conference on Machine Learning, Sydney, Australia, Aug. 2017, pp. 214–223.
[49] D.A. Edwards, On the Kantorovich–Rubinstein theorem, Expo. Math. 29 (4) (2011) 387–398.
[50] T. Miyato, T. Kataoka, M. Koyama, Y. Yoshida, Spectral normalization for generative adversarial networks, in: Proc. International Conference on Learning Representations, Vancouver, Canada, 2018.
[51] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in PyTorch, in: Proc. Advances in Neural Information Processing Systems, Long Beach, CA, Dec. 2017.
[52] L. Li, Y. Li, Z. Li, Efficient missing data imputing for traffic flow by considering temporal and spatial dependence, Transp. Res. C 34 (2013) 108–120.
[53] L. Zhu, N. Laptev, Deep and confident prediction for time series at Uber, in: Proc. IEEE International Conference on Data Mining, New Orleans, LA, Nov. 2017.
[54] J. Zhu, Y. Song, L. Zhao, H. Li, A3T-GCN: Attention temporal graph convolutional network for traffic forecasting, 2020, arXiv:2006.11583 [cs.LG].



James J.Q. Yu received the B.Eng. and Ph.D. degree in electrical and electronic engineering from the University of Hong Kong, Pokfulam, Hong Kong, in 2011 and 2015, respectively. He was a post-doctoral fellow at the University of Hong Kong from 2015 to 2018. He is currently an assistant professor at the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China, and an honorary assistant professor at the Department of Electrical and Electronic Engineering, the University of Hong Kong. He is also the chief research consultant of GWGrid Inc., Zhuhai, and Fano Labs, Hong Kong. His research interests include smart city and urban computing, deep learning, intelligent transportation systems, and smart energy systems. He is an Editor of the IET Smart Cities journal.