Transportation Research Part C 114 (2020) 189–204

Contents lists available at ScienceDirect

Transportation Research Part C

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

Forecasting road traffic speeds by considering area-wide spatio-temporal dependencies based on a graph convolutional neural network (GCN)

Byeonghyeop Yu[a], Yongjin Lee[b], Keemin Sohn[a,*]

[a] Department of Urban Engineering, Chung–Ang University, Seoul, Korea, 84 Heukseok-ro, Dongjak-gu, Seoul 156-756, Republic of Korea
[b] KSB Convergence Research Department, ETRI, Daejeon, Republic of Korea

ARTICLE INFO

Keywords:
Traffic state forecasting
Graph convolutional neural network (GCN)
Spatio-temporal dependencies
Traffic management
Generative adversarial framework

ABSTRACT

The traffic state in an urban transportation network is determined via spatio-temporal traffic propagation. In early traffic forecasting studies, time-series models were adopted to accommodate autocorrelations between traffic states. The incorporation of spatial correlations into the forecasting of traffic states, however, involved a computational burden. Deep learning technologies were recently introduced to traffic forecasting in order to accommodate the spatio-temporal dependencies among traffic states. In the present study, we devised a novel graph-based neural network that expanded the existing graph convolutional neural network (GCN). The proposed model allowed us to differentiate the intensity of connecting to neighbor roads, unlike existing GCNs that give equal weight to each neighbor road. A plausible model architecture that mimicked real traffic propagation was established based on the graph convolution. The domain knowledge was efficiently incorporated into a neural network architecture. The present study also employed a generative adversarial framework to ensure that a forecasted traffic state could be as realistic as possible considering the joint probabilistic density of real traffic states. The forecasting performance of the proposed model surpassed that of the original GCN model, and the estimated adjacency matrices revealed the hidden nature of real traffic propagation.

1. Introduction

The present traffic state of a transportation network is determined by spatial propagations of past traffic states, even though the exact mechanism of the spatio-temporal dependencies is unknown. Time-series models in traffic-state forecasting recognize the temporal dependency of a single directed road link, and researchers have adopted various models in an effort to accurately reflect this dependency: an ARIMA, a space-time ARIMA (STARIMA), a seasonal ARIMA (SARIMA), a vector ARIMA (VARIMA), and a Bayesian VARIMA (Williams and Hoel, 2003; Stathopoulos and Karlaftis, 2003; Kamarianakis and Prastacos, 2005; Chandra and Al-Deek, 2009; and Mai et al., 2012). None of these models, however, deal with area-wide spatial dependencies due to the computational burden. When deep-learning technologies were introduced into the field of traffic forecasting, it expanded the scale of spatio-temporal dependencies that could be considered. In a recent breakthrough, the traffic state of a large transportation network was simultaneously predicted in an incorporated manner.

In the initial stages, researchers believed that a deep learning model could implicitly determine the spatio-temporal relationships

* Corresponding author.
E-mail addresses: yuruka@daum.net (B. Yu), solarone@etri.re.kr (Y. Lee), kmsohn@cau.ac.kr (K. Sohn).

https://doi.org/10.1016/j.trc.2020.02.013
Received 6 September 2019; Received in revised form 13 January 2020; Accepted 14 February 2020
Available online 21 February 2020

that exist among traffic states (Tan et al., 2016; Huang et al., 2014; Lv et al., 2015; and Ma et al., 2015) once a sufficient amount of data could be fed into the model for training. However, this scheme required much time for training and required many weight parameters. Furthermore, because a deep neural network is a black box, there was no way to confirm how spatio-temporal correlations contributed to predicting future traffic states. Nonetheless, such a deep learning model outperformed other engineering-based or econometric models.

Following this success, many researchers began to develop more efficient ways to incorporate the given information on a transportation network into establishing a deep neural network (Cao et al., 2018; Ke et al., 2017; Yu et al., 2017a, 2017b, Jo et al., 2018; Genders and Razavi, 2016; Liang et al., 2019). Researchers converted the traffic state of a transportation network into a grid structure and fed it into a deep learning model as input. Jo et al. (2018) succeeded in forecasting traffic speeds based on image-to-image learning utilizing physical maps augmented with traffic speeds as both input and output. This revolutionary method, however, required the preprocessing of data, which was a burden. The state of the traffic on a transportation network should be changed into a map image. Other researchers found an easier way to change the traffic state of a transportation network into a 2-D matrix structure that could be processed by convolution filters (Genders and Razavi, 2016; Liang et al., 2019). They abbreviated the topology of a transportation network to a simple grid structure, wherein some selected cells of the matrix covered road links and accounted for traffic state values such as speeds, flows, and densities. This simplification, however, must distort the true network shape, which results in a biased spatial structure. More basically, such a 2-dimensional convolution is inefficient because there are many empty cells both in the reinforced map image and in the grid-structured matrix.

The most plausible way to incorporate the topology of a transportation network into a deep neural network was to employ a graph convolutional neural network (GCN). A GCN uses an adjacency matrix to recognize the connectivity of a transportation network, and a graph convolution with the matrix is much more effective than a grid-type convolution. Some researchers have adopted a GCN for traffic prediction (Zhao et al., 2018; Yu et al., 2017a; Cui et al. 2018), and have devised an integrated architecture wherein a GCN can oversee spatial dependencies and a recurrent neural network (RNN) is in charge of temporal dependencies. This scheme was advantageous when compared with the previous deep learning method, because it required a smaller number of weight parameters, and the convergence was fast. Despite its good performance, the integrated architecture has an intrinsic drawback. The traditional GCN assumes that an adjacency matrix used to indicate first-order connections to neighbors is constant. That is, influences of a road link on its neighbor links are the same. This could hamper a model's capability of accounting for traffic propagations in a road network. It is more natural to assume that the change of the traffic state of a link would affect the future traffic state of neighbor links in a different manner. Furthermore, a RNN has to be separately attached to a GCN in order to accommodate the temporal dependencies, which complicates model architectures. The architecture proposed in the present study was composed of only graph convolution blocks, which were simpler and more compatible with the real phenomena of traffic propagation. A few fully connected layers (FCs) were optionally attached to the GCN architecture to enhance the performance.

The present study was focused on devising extended GCN models that mimic true propagation patterns of traffic by incorporating a given domain knowledge into a deep neural network. The contribution of the present study is three-fold. First, a simpler graph model was developed that can intuitively account for traffic propagation in a spatio-temporal space. An adjacency matrix was parametrized, so that the matrix could account for different intensities of traffic propagation from a specific road link into neighbor links. Multiple adjacency matrices were mobilized to address different traffic propagation patterns for each time lag. The weight matrices that multiply after each graph convolution were not included in the proposed model, and thus only adjacency matrices included parameters. Furthermore, the adjacency matrix was weighted by the distance and capacity of road links, so that the intensity of traffic propagation from a road link could be proportional to the number of lanes of connected links and inversely proportional to the distance of the connected links. This scheme drastically reduced the number of parameters to be learned. Each adjacency matrix had only one parameter.

Second, unlike previous GCN-based traffic forecasting models, the present approach depends only on graph convolutions to accommodate spatio-temporal correlations. The model did not include a RNN sub architecture that could separately reflect temporal dependencies. Instead, a different number of graph convolutions were applied for each time lag, and an older traffic state could allow for more graph convolutions. Several fully connected layers were optionally attached after GCN blocks to consolidate the processing of spatio-temporal correlations between traffic features derived from different time lags. This led to a simpler model structure that was compatible with real traffic phenomena without separately including RNN and GCN blocks.

Last, a generative adversarial framework was employed to ensure that a predicted traffic state would be as realistic as possible. A traffic state in a transportation network could naturally be regarded as an instance sampled from the joint probabilistic density of traffic states. Thus, even though a forecasting model attempts to minimize the differences between predicted and observed traffic states, it is unlikely that a predicted state comes from real traffic states if the joint probabilistic density of traffic states had multiple modes. This is a common problem in a regression-based estimation. To solve the problem within a generative adversarial framework, a discriminator attempts to secure the reality of the output predicted by a generator that minimizes the mean squared error (MSE). Several researchers have already adopted a generative adversarial framework in traffic forecasting (Lei et al., 2019; Lin et al., 2019). They compared the performance of the adversarial framework with those from other deep learning models. The present study tested the frameworks of all suggested models and compared their performances with those from models without the framework.

The next section introduces the prototype of the original GCN and discusses its application history. The third section describes how to extend the original GCN to mimic the real phenomenon of traffic propagation. How to select the testbed and collect data is included in the fourth section. The fifth section compares the prediction results from the proposed model with those from other state-of-the-art methodologies. Conclusions are drawn and further possible expansion is suggested in the last section.

2. Overview of GCNs

Convolutional neural networks (CNNs) have succeeded in solving many engineering problems that can provide input from a grid-like structure. CNNs reduce the number of parameters by reusing a convolutional filter for each local region of grid input. Data that are not in the form of a grid structure, however, cannot be used as input for a CNN model. A transportation network is a typical example of data that cannot be represented by a grid structure, since it takes the form of a graph that consists of nodes and links. In the early stages, a graph neural network (GNN) was developed to process graph input wherein a recursive neural network repeatedly processed node states until they reached a fixed equilibrium state (Gori et al., 2005; Scarselli, 2009). Kipf and Welling (2016) later improved this scheme by introducing the concept of graph convolution.

The GCN devised by Kipf and Welling (2016) originated from the spectral-graph CNN introduced by Bruna et al. (2013) when devising a convolution operation in the Fourier domain that required computation of the Eigen-decomposition for the graph Laplacian, which entails a computational burden when dealing with a large-scale graph. In order to avoid computing the Eigen-decomposition, Defferrard et al. (2016) introduced spatially localized filters and approximated them based on the Chebyshev expansion to the graph Laplacian. Kipf and Welling (2016) finally simplified the GCN by confining the filters' operation to only the first-order neighbors. A brief description of the working principle of the GCN follows.

Given a graph G = (V, E), a GCN requires two matrices as input. A feature matrix ($X \in \mathbb{R}^{N \times F^0}$) and an adjacency matrix ($A \in \mathbb{R}^{N \times N}$) is fed into a GCN. V is a set of vertices (=nodes), E is a set of edges, N is the number of vertices, and F^0 is the number of given features for a node. A hidden layer of the GCN is represented as $H^l = f(H^{l-1}, A)$ where the initial hidden layer (H^0) is set to be X and f is a propagation rule. How to choose the propagation rule is the key in the concept of GCN. Eq. (1) is the simplest form of a layer-wise propagation rule.

$$H^l = \sigma(AH^{l-1}W^{l-1}) \tag{1}$$

In Eq. (1), σ represents an activation function such as a sigmoid and a rectified linear unit (ReLU), and $W^{l-1} \in \mathbb{R}^{F^{l-1} \times F^l}$ is a weight parameter matrix.

However, there are two typical problems regarding Eq. (1). First, multiplying an adjacency matrix with a hidden-feature matrix cannot transfer a node's own features to the next layer. Only features of its neighbor nodes can be passed to the next layer. To circumvent this drawback, an identity matrix is added to the given adjacency matrix ($\tilde{A} = I + A$). Second, feature values scale up as the layer-by-layer propagation repeats. The average normalization of the adjacency matrix ($\tilde{A}$) can be an option to circumvent this problem. A diagonal node-degree matrix (D) is prepared such that its diagonal element sums up the row values of the adjacency matrix ($D_{ii} = \sum_j \tilde{A}_{ij}$). The adjacency matrix is then normalized by multiplying the inverse of the diagonal matrix and the given adjacency matrix ($D^{-1}\tilde{A}$), and thus all rows of the resultant matrix sum to one. On the other hand, instead of this average normalization, Kipf and Welling (2016) adopted a spectral normalization to secure more dynamics for the propagation rule ($D^{-\frac{1}{2}}\tilde{A}D^{-\frac{1}{2}}$). The spectral normalization scheme divided an element of the adjacency matrix by the row sum and the column sum $\left(\frac{\tilde{A}_{ij}}{\sqrt{D_{ii}}\sqrt{D_{jj}}}\right)$, whereas the average normalization scheme divided an element only by the row sum in ($\frac{\tilde{A}_{ij}}{D_{ii}}$). The final propagation rule established by Kipf and Welling (2016) takes the form of Eq. (2).

$$H^l = \sigma\left(D^{-\frac{1}{2}}\tilde{A}D^{-\frac{1}{2}}H^{l-1}W^{l-1}\right) \tag{2}$$

It should be noted that this propagation rule has a handicap when used to forecast the traffic state in a transportation network. The rule assumes a fixed adjacency matrix. However, when a road link's traffic sate propagates into downstream and upstream road links, the influence of the road link on each neighbor road link differs. Graph-attention networks were developed to assign different weights to neighboring nodes and to learn the weights together with other model parameters (Veličković et al., 2017; Zhang et al., 2018a, 2018b; and, Abu-El-Haija et al., 2017). Our attempt to differentiate the intensities of traffic propagation to neighboring road links is compatible with the attention context devised by the previous researchers.

3. Extended GCN

When an end-to-end deep learning model is employed in a specific domain, domain experts tend to suggest that the model cannot explain its working mechanism. No matter how good of a performance the deep learning model yields, the model is unacceptable unless the good performance can be fully explained. The objective of the present study was to develop an extended GCN model, the architecture of which could be considered consistent with a real traffic propagation mechanism. That is, a graph convolution was interpreted as a spatio-temporal traffic propagation. As mentioned earlier, an adjacency matrix was parametrized so that a larger weight could be assigned to more influential downstream or upstream road links. Moreover, multiple adjacency matrices were parametrized, each of which corresponded to a traffic propagation starting from a specific time interval in the past. The extended GCN allowed a traffic state to propagate in a spatio-temporal space, so that a road link's current traffic state could reflect past traffic states of neighboring road links. This made it possible for the extended GCN to accommodate spatio-temporal dependencies without the incorporation of a RNN.

A transportation network should be transformed to a graph to be dealt with in a GCN. Because the average vehicle speed on a road link was the target to be predicted, each road link of a transportation network was regarded as a node of a graph. Connections

between a road link and its first-order neighbors are defined as edges. Prior to setting up the model, a fixed adjacency matrix that indicates direct connections between nodes (=road links) in a graph was set up and used as a mask to build variable adjacency matrices.

Each variable adjacency matrix was parametrized using softmax functions [see Eq. (3)]. A road link had a constant subset of directly connected road links, and this information had to be recognized in advance. Each adjacency matrix assigned free parameters only to non-zero elements for these direct connections. According to this scheme, the intensity of traffic propagation was learned based on observed data. After training the extended GCN, the estimated adjacency matrix can show which neighbor links are most important in a specific time lag.

$$A_{ij}^{l} = \text{softmax}(\alpha_{ij}^{l}) = \frac{e^{\alpha_{ij}^{l}}}{\sum_{k\in\mathfrak{N}_i} e^{\alpha_{ik}^{l}}} \tag{3}$$

In Eq. (3), A_{ij}^{l} is an element of an adjacency matrix for the l^{th} hidden layer (A^{l}), α_{ij}^{l} is a weight parameter to be learned, and $\mathfrak{N}_i$ is a set of road links that are directly connected to the road link i.

Another option for setting up an adjacency matrix was devised to utilize the given road information and to facilitate the computation. The distance and the number of lanes of road links were embedded with the adjacency matrices [see Eq. (4)]. This scheme drastically reduced the number of parameters to be learned. An adjacency matrix had only one parameter. The softmax function is known as the multinomial logit model in the transportation study field and has been widely used for the discrete choice analysis (Beak and Sohn, 2016; Chang and Sohn, 2015). According to the formula, even though a single free parameter, β^{l}, plays a role in determining the weight intensity (A_{ij}^{l}), a road link with a shorter distance (d_j) and more lanes (n_j) is likely to have a higher probability of traffic propagation, which is a basic assumption of route-choice analysis in the transportation study field (Yang et al., 2016; Zhang et al., 2018a, 2018b; and, Dell'Orco and Marinelli, 2017). This mechanism could also be explained using the concept of graph-attention networks (Veličković et al., 2017). However, the aforementioned interpretation is much easier for transportation engineers to understand.

$$A_{ij}^{l} = \text{softmax}(\alpha_{ij}^{l}) = \frac{e^{\beta^{l}(\frac{n_j}{d_j})}}{\sum_{k\in\mathfrak{N}_i} e^{\beta^{l}(\frac{n_k}{d_k})}} \tag{4}$$

In Eq. (4), β^{l} is a weight parameter to be learned, d_j is the distance of a road link j, and n_j is the number of lanes in the road link j.

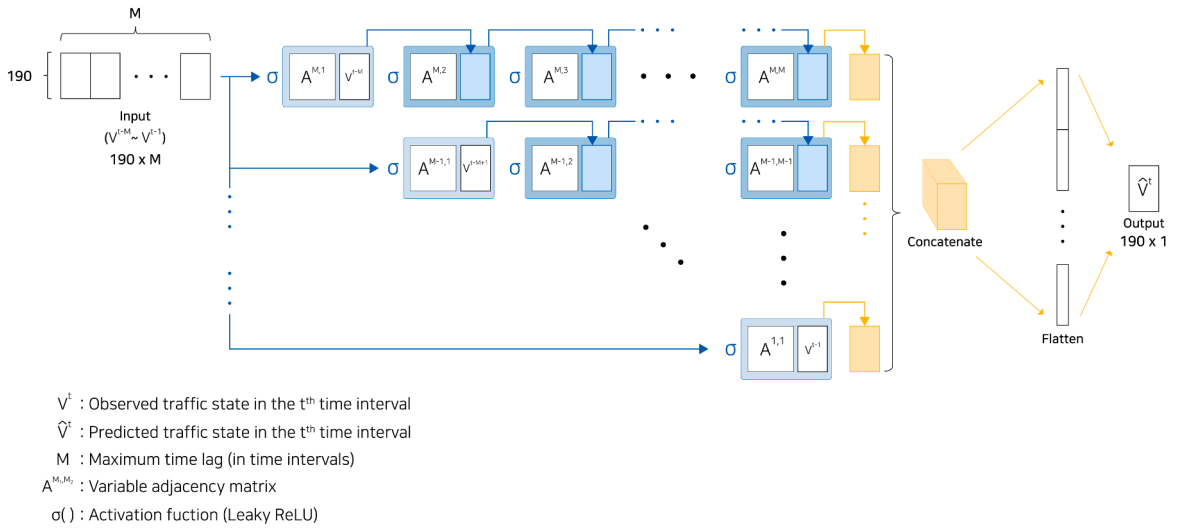
$$H^{l} = \sigma(A^{l-1}H^{l-1}) \tag{5}$$

A layer-wise convolution of the extended GCN is represented by Eq. (5). The difference between Eqs. (1) and (5) is that adjacency matrices (A^{l})of the extended GCN are parametrized based on softmax functions without weight matrices, whereas an adjacency matrix (A) of an original GCN is a constant matrix and weight matrices (W^{l-1}) have parameters to be estimated. In fact, if a single feature (=traffic speed) is used as in the present study, weight matrices (W^{l}) are redundant in the extended GCN. As for how to parametrize adjacency matrices, two different extended models were devised. The first extended GCN parametrized all non-zero elements of an adjacency matrix [see Eq. (3)], whereas the second extended GCN had only a single parameter for each adjacency matrix [see Eq. (4)]. The second extended GCN instead incorporated road properties such as capacity and length into the model architecture.

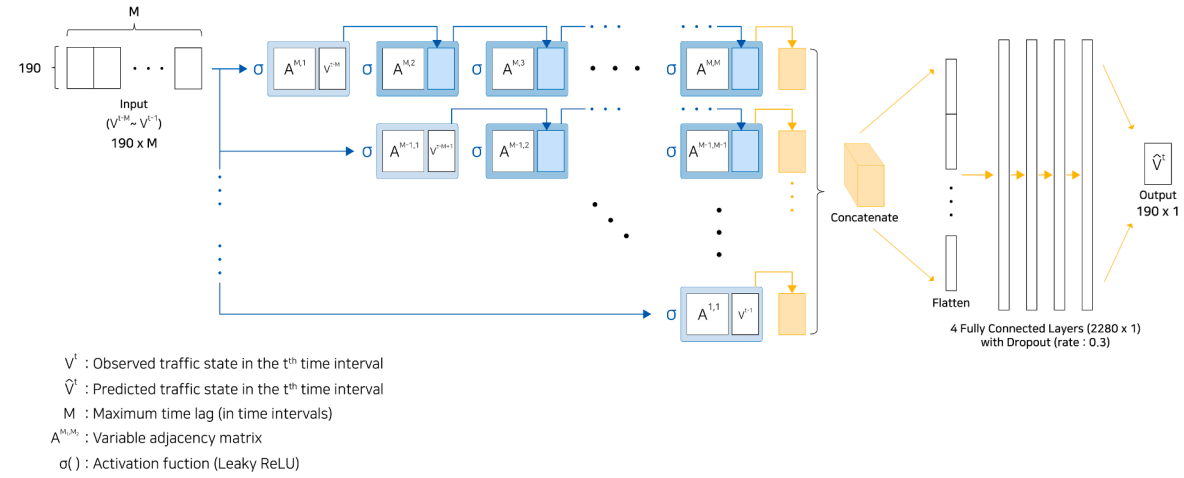
The architecture of the extended GCN was set up to mimic the real propagation of a traffic state. The current traffic state of a road is influenced by the past traffic state of upstream or downstream roads. As the time lag grows longer, remote road links are more likely to impact the current traffic state of a road of concern. The influential range widens as the time lag increases. In this context, the higher-order neighbor links require the graph convolution process more than the lower-order neighbor links that are closely connected to a road link of concern. According to this simple concept, the architecture of an extended GCN was established, as shown in Fig. 1. The top stream of graph convolutions accounts for the mechanism of how the oldest traffic state influences the current traffic state. The oldest traffic state went through the same number of graph convolutions as the number of intervals for the largest time lag. On the other hand, the traffic state in the previous time interval undertook only a single graph convolution, as shown in the bottom of the model, because during a single time interval the traffic states of remote road links cannot affect the current traffic state.

More concretely, the top stream of the proposed model architecture extracted the influence of the traffic state in the oldest time interval, and the bottom stream abstracted the influence of the traffic state of the previous time interval. Output tensors from all streams (=past time intervals) were concatenated together and fed into the next layer. There were two options to accommodate the outputs. Basically, the concatenated tensor was directly connected to the output layer. This means the current traffic state is a linear combination of the features extracted from the past time intervals, each of which derived from repeating a different number of graph convolutions. As an alternative to reinforce the capability of the model to recognize spatio-temporal dependencies, several fully connected (FC) layers were attached to the concatenated tensor, although this entailed increasing the number of parameters to be learned.

The proposed architecture removed the weight matrix (W^{l}), because each road link had a single feature (=traffic speed) as input. The weight matrices were included in the original GCN only to recognize hidden correlations between multiple features. The proposed model allowed multiple adjacency matrices ($A^{t,t'}$) to reflect different connection intensities for different time lags. That is, the influence that older traffic states exerted on the current traffic state was separated from the influence of more recent traffic states.



(a) Extended GCN



(b) Extended GCN + FCs

Fig. 1. The proposed model architecture.

Although the proposed architecture of the extended GCN was straightforward in explaining the actual traffic propagation, the number of time lags that are identical to the number of graph convolutions might not be consistent with actual traffic phenomena. At the present time, unfortunately there is no way to consider the exact time taken for the link-to-link traffic propagation within a GCN framework. It is very difficult to parametrize the propagation time in a GCN model. As a practical matter, the propagation of a traffic state to first-order neighbors was assumed to finish within a prescribed time interval.

4. Testbed and data collection

The Seoul metropolitan government provides the public with traffic speed data across every road segment within its jurisdiction area. The Gangnam is one of the busiest regions in the Seoul metropolitan area and thus was chosen as the testbed for the present study. Fig. 2 shows the testbed that includes 190 road links, each of which was a spatial unit wherein speed data were collected from 20,000 probes. Each probe taxi was embedded with an on-board unit that transmitted its location and speed on a regular basis (= 0.2 sec). Five minutes of the collected speed data for every road link were then averaged and released to an individual after a simple authentication, as requested.

We obtained speed data for 14 months (2016/8/1–2017/9/30). Missing data were imputed by averaging previous and next speeds. The present study adopted two intervals: 5 and 15 min. For the latter, three speeds from the original 5-minute intervals were aggregated. 80% of the data were used for training, and the remainder (20%) were reserved for the final test. Among the training data, 10% was randomly chosen for cross-validation.

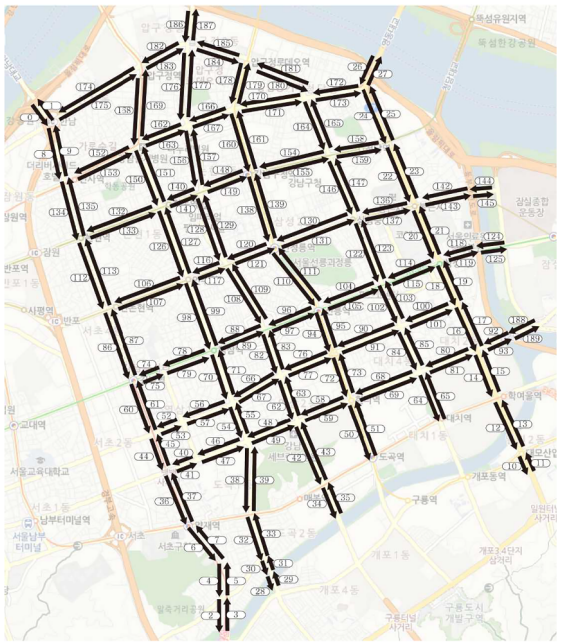


Fig. 2. Road map of the testbed.

A constant adjacency matrix representing the road connection was prepared in advance. For each road link, all first-order neighbors were identified, and cells that belonged to the row of the link took the unity value only when the cell's column matched the neighboring links. The matrix was then used as a mask to comprise parametrized variable adjacency matrices. The neighboring links encompassed both downstream and upstream road directions. The rationale to include both directions stemmed from the assumption that both traffic states affect the future traffic state of a link of concern. As mentioned earlier, a road link's adjacent status covered the autocorrelation by adding an identity matrix. How to comprise the constant adjacency matrix is depicted in Fig. 3.

5. Test results and comparisons

Many studies have dealt with forecasting short-term traffic states such as traffic flows, speeds, vehicle occupancies, travel times, and their combinations. The proposed model cannot be compared with all the methodologies developed thus far. We focused only on learning-based methodologies that have attempted to accommodate area-wide spatio-temporal dependencies. First, an original GCN model incorporated with gated recurrent units (GRU) was chosen as the baseline (Zhao et al., 2018a, 2018b), since it adopted a single constant adjacency matrix, which was unlike the proposed model with variable adjacency matrices. Our primary concern was to confirm whether variable connection intensities could enhance the forecasting accuracy. An image-to-image learning model was chosen as the second reference because it accommodated area-wide spatio-temporal correlations based on a grid-type convolution unlike the proposed model with a graph convolution (Jo et al., 2018). Both reference models outperformed other learning-based models without directly integrating a transportation network topology (Zhao et al., 2018a, 2018b; Jo et al., 2018). Thus, for brevity the comparisons with other vanilla deep learning models were not included in the present study.

Another interest of the present study was to confirm whether a generative adversarial framework could enhance the forecasting

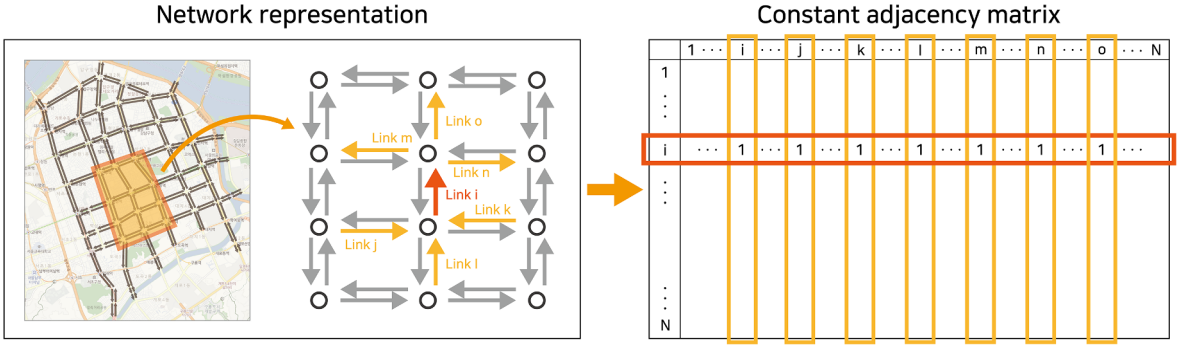


Fig. 3. The connectivity of a transportation network.

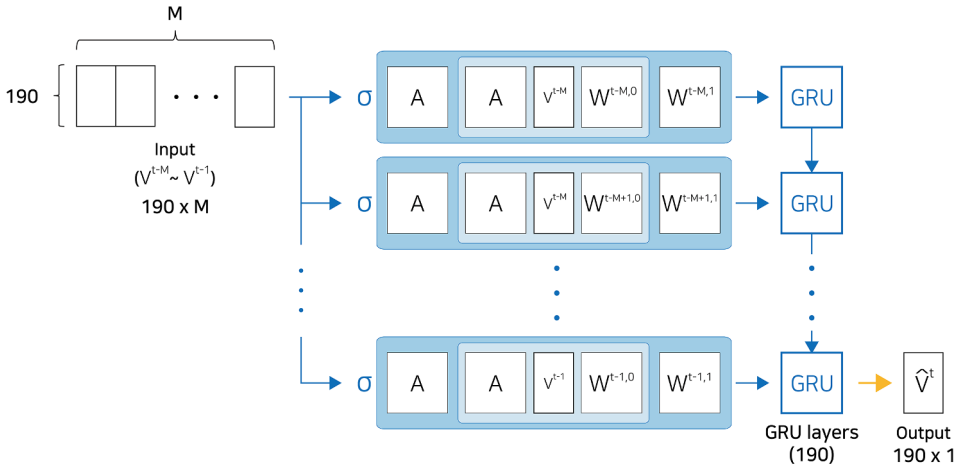


Fig. 4. Architecture of a reference model that incorporates an original GCN with a GRU.

models. Several previous researchers confirmed that a model with this framework outperformed other vanilla deep learning models (Lei et al., 2019; Lin et al., 2019). We verified whether a generative adversarial framework could be beneficial to the extended GCNs and all other reference models. The structure of reference models and generative adversarial frameworks are briefly introduced here, as follows.

5.1. Original GCN model

Several researchers have attempted to use GCN-based models to predict traffic states (Zhao et al., 2018; Yu et al., 2017a; Cui et al., 2018). They all adopted similar model architectures wherein an original GCN, which had been developed by Kipf and Welling (2016), was integrated with a RNN. Zhao et al. (2018) set up a model architecture that incorporated an original GCN with GRU blocks. The model architecture was chosen as the baseline model, as shown in Fig. 4. The GCN block for a time lag in the architecture included two consecutive graph convolutions and applied a RELU activation to the convoluted outcome.

5.2. Image-to-image model

Jo et al. (2018) confirmed that an image-to-image learning model based on a CNN outperformed other learning models with vector input such as a deep neural network (DNN) and a long short-term memory model (LSTM). A map augmented with traffic speeds was used as both input and output for the model. A CNN proved successful in abstracting area-wide spatio-temporal correlations between traffic states. Considerable effort was required, however, to create augmented map images. The color intensity differentiated the speed levels for each road link in the map, and the link width was drawn proportional to the number of lanes to include the capacity effect. Filters of grid-type convolution were then slid through the map images to extract spatio-temporal correlations between traffic states. Such a 2-D convolution, moreover, was successful only when sufficiently fine-grained images were used (512 × 384), which considerably increased the computation time. An image-to-image model was chosen as a reference for comparison [see Fig. 5].

5.3. Generative adversarial framework

Either a mean square error (MSE) estimation or a mean absolute error (MAE) estimation has been adopted in most traffic forecasting models. Both have drawbacks, however, wherein the outcome leads to the mean value of the probabilistic density of a target variable. Such an outcome would not be realistic when a target variable has a skewed or a multi-modal probability density. In computer-vision studies, an adversarial framework has been adopted to resolve this problem (Gafni et al., 2019; Zhu et al., 2017; and Isola et al., 2016). This framework employs a discriminator and adapts it to distinguish whether the outcome is real or synthesized. A solution from this scheme at least guarantees one of the most frequently observed outcomes, which occurs when a target variable is distributed with a multi-modal density function.

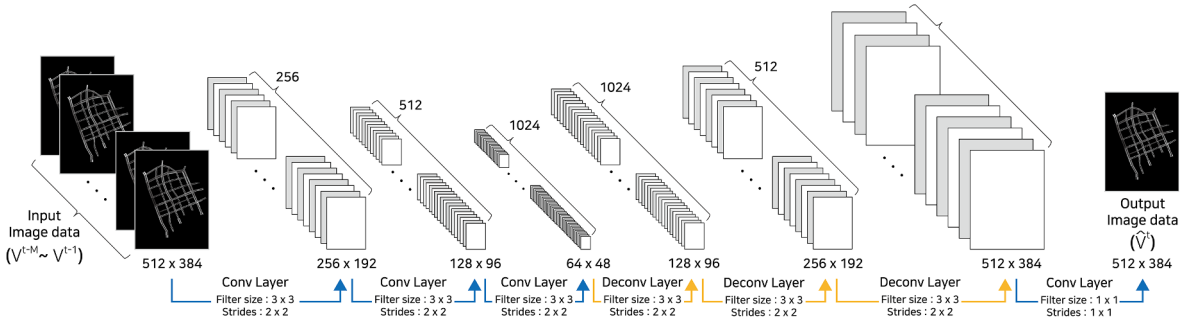


Fig. 5. Architecture of a reference model based on image-to-image learning.

Several researchers have employed a generative adversarial framework in forecasting traffic states (Lei et al., 2019; Lin et al., 2019). These researchers verified that predictions with the framework outperform other machine learning technologies. We applied a generative adversarial framework commonly to the proposed GCN model and the two reference models. A Wasserstein generative adversarial framework (WGAN) was commonly applied to all models to secure consistent convergence (Arjovsky et al., 2017). The prediction model was regarded as a generator, and an additional discriminator was set up and integrated with the generator (see Fig. 6). The architecture of the discriminator for the image-to-image model took a CNN, and those for the extended GCN and the original GCN were composed of simple dense layers according to the specifications of the two previous studies (see Fig. 7). All the discriminators took the form of a PatchGAN (Isola et al., 2016; Ledig et al., 2016; and Li and Wand, 2016), wherein a tensor output was used for the discriminator rather than a single entropy output, so that each element of the tensor output could judge whether a part of the input tensor was observed or synthesized. The ground-truth label had the same dimensions and was filled with binary values according to whether an input tensor was observed or synthesized.

In principle, a generator should have random noise as an input, so that the output will match any probabilistic distribution (Xie et al., 2015). The present framework did not adopt explicit random input, however, unlike the original GAN structure. An indirect measure was taken instead in order to secure randomness inside the generator models. We provided generators with noise in the form of dropout and instantaneous normalization (Owens et al., 2015; Isola et al., 2016). The dropout operation randomly selects weight parameters to be updated for each iteration (Srivastava et al., 2014). The instance normalization adds random noise to the standard deviation computed across spatial dimensions independently of each feature map within a batch input (Ulyanov et al., 2016).

The algorithm for training the given traffic forecasting models within an adversarial framework will be briefly introduced. An adversarial framework is composed of two models: a generator and a discriminator. In each iteration of the algorithm for training a GAN, there are two standpoints. The first involves the perspective for a generator model, which is updated based on a batch of data such that the discrepancy between the estimated and observed traffic speeds can be minimized. Simultaneously, a discriminator is trained on the estimated results by regarding them as real. Second, from an adversarial point of view, the discriminator is updated so that the estimated outcome can be classified as fake while the observed speeds are regarded as true. Alternately repeating these two steps guarantees a convergence of the GAN model. For more details, readers should refer to our previous work (Lee et al., 2019).

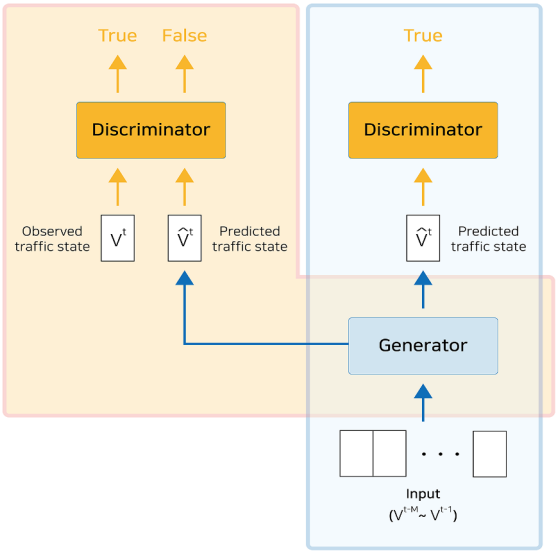
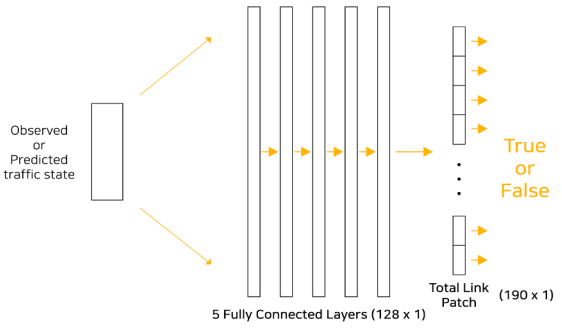
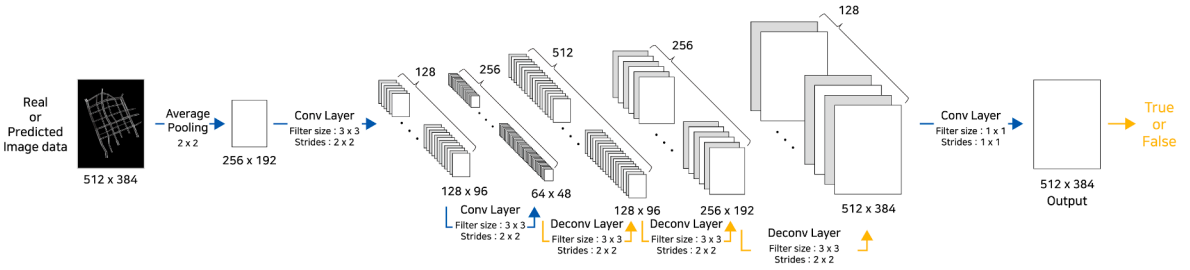


Fig. 6. Applied generative adversarial framework.



(a) Discriminator for the extended GCN and the original GCN



(b) Discriminator for the image-to-image model

Fig. 7. Applied discriminator structures.

5.4. Comparing test results

The mean absolute error (MAE) and the percent root mean square error (PRMSE) were selected as basic performance measures. Fortunately, there were no contradictory results between assessments by the two performance indices. Regarding the comparison of computing time, two indices were adopted: the time taken to reach convergence and the time for a single evaluation of a model. For the former, the training times required to repeat 10,000 epochs were measured for all models. While training, the best results obtained from a 10% cross-validation dataset were saved. For models of 15-minute intervals, four kinds of time lags for an input were tested: 30 min (=2 intervals), 45 min (=3 intervals), an hour (=4 intervals), and two hours (=8 intervals). For models with 5-minute intervals, only an hour of time lag (=12 × 5 min) was tested due to the shortage of GPU memory for input.

Two extended GCNs were included in the present investigation. The extended GCN I had free parameters (=non-zero elements) for an adjacency matrix, whereas the extended GCN II had only a parameter for an adjacency matrix [see Eq. (4)] with the capacity (=number of lanes) and length of road links incorporated into an adjacency matrix. To consolidate the explanatory power, the two extended GCNs optionally four FCs after being flattened. For brevity, the original GCN model incorporated with GRUs will be referred to as the original GCN model. The performance of each model was compared based on the same test dataset that had been reserved in advance. The results are compared in Table 1, and for clarity the best performance measures were highlighted with a bold font for each model.

Regarding models without a generative adversarial framework, the extended GCNs recorded a better MAE and PRMSE than the original GCN model with a fixed adjacency matrix. This implies that a variable adjacency matrix is definitely advantageous in extracting spatio-temporal dependencies, and the model structure that mimicked the real traffic propagation worked well. In particular, it was surprising that the extended GCN II outperformed the original GCN for both performance measures (=MAE and PRMSE). This result was revolutionary since the model assigned only a single parameter to an adjacency matrix, and it proves that the pre-information on road properties (=capacity and length of road link) enhanced the accuracy of traffic speed prediction. The model's performance did not exceed that of the extended GCN I with all elements of an adjacency matrix freely parametrized.

By comparison with an image-to-image model, the extended GCNs exhibited slightly inferior performance. The reason the extended GCN I was inferior to the reference model with 2-D convolutions could have been due to the absence of information on the road properties such as capacity and road length. A fine-grained input for the image-to-image model could involve almost all spatial properties of the transportation network. The reason the extended GCN II was also inferior to the reference model was that it had insufficient free parameters despite the inclusion of the two road properties in the model architecture. However, it should be noted that the convergence of the GCN models was much faster and lighter than the reference model. Unlike a 2-D convolution model, the proposed GCN models required no preprocessing to generate very fine-grained images for input. If course images are used, however, an image-to-image model must not exceed the extended GCN models. Moreover, the computation time for training the reference model was much longer than that for the extended GCN models (see Table 1). When four FCs were attached to the end of graph

Table 1
Model performances.

	WGAN	Time interval	Time lag	Extended GCN I	Extended GCN I + FCs	Extended GCN II	Extended GCN II + FCs	Image-to-Image model	Original GCN + GRU
MAE	No	5 min	12	2.048	1.994	2.110	1.976	1.970	2.322
		15 min	8	1.507	1.417	1.524	1.407	1.498	1.760
			4	1.493	1.412	1.528	**1.402**	1.447	1.825
			3	**1.485**	**1.396**	**1.520**	1.409	**1.440**	**1.748**
			2	1.490	1.406	1.521	1.409	1.449	1.794
	Yes	5 min	12	2.040	1.963	2.114	1.986	1.960	2.359
		15 min	8	1.534	1.412	1.529	**1.412**	1.495	**1.808**
			4	1.530	1.412	1.525	1.419	1.476	1.833
			3	**1.513**	**1.401**	**1.516**	1.415	**1.453**	1.840
			2	1.528	1.402	1.520	1.420	1.464	1.972
%RMSE	No	5 min	12	12.204	12.388	12.351	11.919	12.222	13.161
		15 min	8	8.605	8.215	8.647	8.087	8.286	9.855
			4	8.498	8.086	8.626	8.059	8.202	10.192
			3	8.477	8.035	8.598	8.085	8.311	9.769
			2	8.488	8.094	8.618	8.100	8.250	10.072
	Yes	5 min	12	12.201	11.864	12.411	11.908	12.016	13.589
		15 min	8	8.702	8.114	8.670	8.128	8.351	10.177
			4	8.661	8.082	8.609	8.184	8.350	10.309
			3	8.577	8.066	8.580	8.116	8.394	10.299
			2	8.671	8.083	8.632	8.180	8.495	11.116
Time for convergence (minute)	No	5 min	12	1041	1210	1547	1534	1249	1040
		15 min	8	551	681	707	838	1201	800
			4	204	340	276	392	1137	548
			3	153	212	183	220	1116	555
			2	128	152	154	181	1006	492
	Yes	5 min	12	1541	1739	1901	1994	4220	1518
		15 min	8	668	1120	888	939	2171	1101
			4	263	359	383	464	2720	760
			3	233	342	238	391	1931	599
			2	173	204	170	325	1901	526
Model evaluation time (second)	No	5 min	12	0.022	0.038	0.036	0.030	0.091	0.013
		15 min	8	0.019	0.021	0.016	0.016	0.074	0.012
			4	0.006	0.008	0.005	0.006	0.066	0.010
			3	0.004	0.005	0.003	0.004	0.060	0.010
			2	0.002	0.003	0.002	0.003	0.059	0.009
	Yes	5 min	12	0.034	0.034	0.034	0.034	0.091	0.016
		15 min	8	0.016	0.017	0.016	0.016	0.076	0.012
			4	0.008	0.007	0.008	0.008	0.065	0.016
			3	0.003	0.004	0.003	0.004	0.064	0.010
			2	0.002	0.003	0.002	0.003	0.063	0.009
# of params (thousand)	No	5 min	12	503	21,341	433	21,236	21,265	118
		15 min	8	320	19,571	289	19,503	21,255	115
			4	153	17,815	144	17,770	21,246	112
			3	113	17,553	108	17,337	21,244	111
			2	75	16,943	72	16,904	21,242	110
	Yes	5 min	12	503 + 115	21,305 + 115	433 + 115	21,236 + 115	24,413	118 + 115
		15 min	8	320 + 115	19,534 + 115	289 + 115	19,503 + 115	24,404	115 + 115
			4	153 + 115	17,778 + 115	144 + 115	17,770 + 115	24,395	112 + 115
			3	113 + 115	17,342 + 115	108 + 115	17,337 + 115	24,392	111 + 115
			2	75 + 115	16,906 + 115	72 + 115	16,904 + 115	24,390	110 + 115

convolutions, the performance of the extended GCNs exceeded that of the image-to-image model at the expense of increasing the computation time.

The two extended GCNs commonly had the best performance when adopting the input data from three 15-minute intervals (=45 min). Adopting longer time lag for input did not clearly enhance the forecasting accuracy. Unlike the prior expectations, lessening the interval from 15 min to 5 min did not improve the model performance. This could have been due to the insufficiency of using probe taxis within a 5-minute interval. Also, some data were missing from the raw data of 5-minute intervals. Aggregating these incomplete data into 15-minute intervals must have reduced the variance in the probe taxi speeds, which could be explained using the central-limit theorem.

Regarding the computation efficiency, the time taken to evaluate the forecasting model is more critical than that to train the model, because practical computing time is the most important prerequisite for onsite real-time traffic forecasting. Table 1 shows that

Table 2
Model performances for multi-period predictions.

Performance	Time horizon	Extended GCN 1	Extended GCN 1 + FCs	Extended GCN 2	Extended GCN 2 + FCs
MAE	15 min.	1.485	1.396	1.520	1.409
	30 min.	1.716	1.372	1.787	1.407
	45 min.	1.930	1.491	1.997	1.503
	60 min.	2.088	1.648	2.138	1.661
%RMSE	15 min.	8.477	8.035	8.598	8.085
	30 min.	9.890	7.903	10.053	8.134
	45 min.	11.847	8.587	11.215	8.691
	60 min.	13.981	9.379	12.010	9.483

the ranking for the model evaluation time was almost identical to that for the model training time. It should be noted that the evaluation time for the extended GCNs was much shorter than that for any other reference models, which means the proposed model shows promise for application to a real-time traffic forecasting system.

The influence that the addition of a generative adversarial framework exerted on the model performance was investigated. Caution should be used when judging whether a generative adversarial network (GAN) structure benefited the model performance, because how realistic the forecasting results are cannot be measured directly. The criterion used to minimize the discrepancy between observed and forecasted results was included in the loss function of the proposed GAN model, but it could not represent the entire performance that includes a measure of how realistic the forecasting results are. Nonetheless, a MAE and PRMSE had to be used to compare model performances, since there was no way to measure the reality of traffic speed patterns in a transportation network. The two indices for models with a generative adversarial framework were almost the same as those for models without this framework. Previous studies verified that a GAN model outperformed the vanilla leaning models such as DNN, LSTM, and CNN (Lei et al., 2019; Lin et al., 2019). However, the framework provided little improvement in the models incorporating the network topology intrinsically. It should be noted, however, that this conclusion was based solely on the prediction accuracy. A generative adversarial framework could result in better performance in securing prediction reality.

The performance of multi-period forecasting was investigated for the extended GCN models. As shown in Table 2, the performance deteriorated gradually as the forecasting horizon increased. When the time horizon increased to an hour, which was four-fold that of the original time horizon, the deterioration in performance was acceptable in that the two performance measures were much less than twice that of a future single period.

Fig. 8. Road links with a poor prediction accuracy.

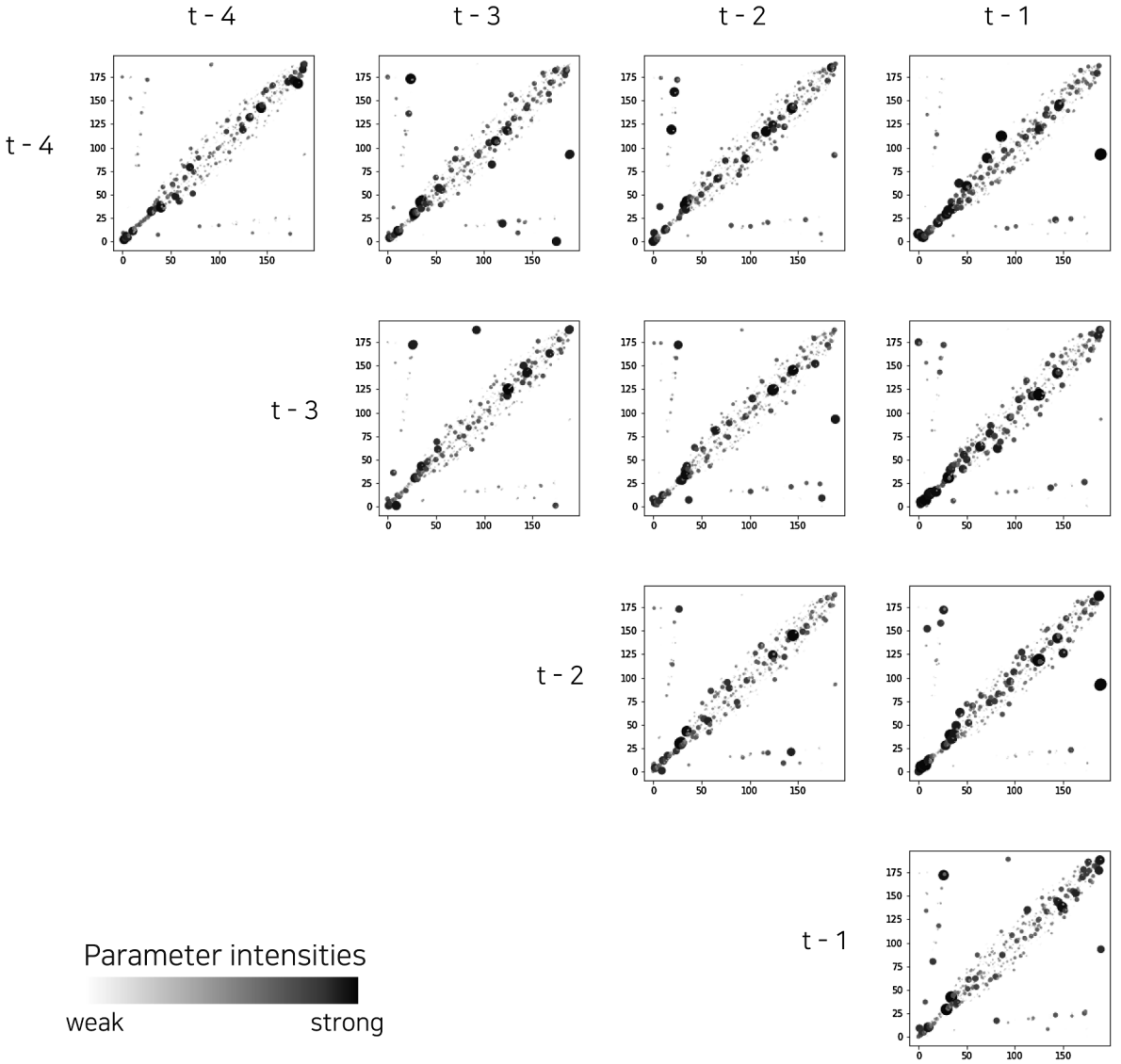


Fig. 9. Estimated intensities of adjacency matrices.

The worst 30 road links with poor forecasting results are indicated in the map shown in Fig. 8. A color that approaches bright red indicates a poorer prediction accuracy. Most links were located near a boundary of the testbed network. Omitting information from links outside the network resulted in a deterioration of the prediction accuracy for fringe links. This verified that spatial correlations clearly contributed to forecasting traffic states. Some links of higher congestion also had poor prediction results.

5.5. Estimates adjacency matrices

The extended GCN I had a great advantage in that elements of the adjacency matrix can be estimated from the data. For an explanation of the estimated adjacency matrices, the model adopted a 15-minute interval and 4 time-lags for input. Thus, 10 different adjacency matrices were derived. Fig. 9 shows plots of softmax function values for the estimated adjacency parameters of the 10 matrices. For better visibility, the connection intensity was represented by dot size and color strength. The top row stream for the longest time lag had different adjacency matrices. This implies that the distribution of each adjacency matrix represented different traffic propagation patterns across time lags with regards to forecasting the next state of traffic. The matrix located in the bottom position accounted for how the previous traffic state affected the very next traffic state.

The relative influence of the links neighboring a specific link located in the middle of the testbed was investigated (see Fig. 10). A yellow horizontal line on adjacency matrices corresponds to the row of a chosen link. Neighbor links with higher attention intensity are highlighted in dark blue. Nonetheless, interpreting the estimated elements of adjacency matrices was not straightforward.

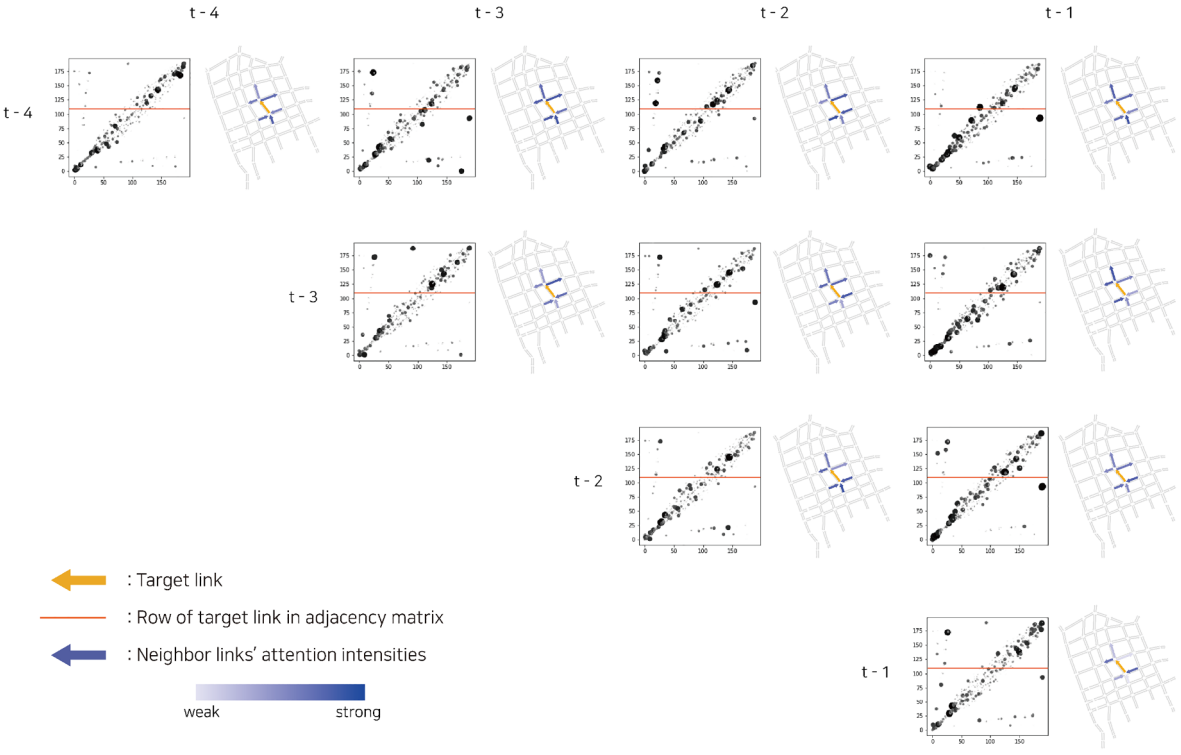


Fig. 10. Attention intensities of a link located in the middle of the testbed.

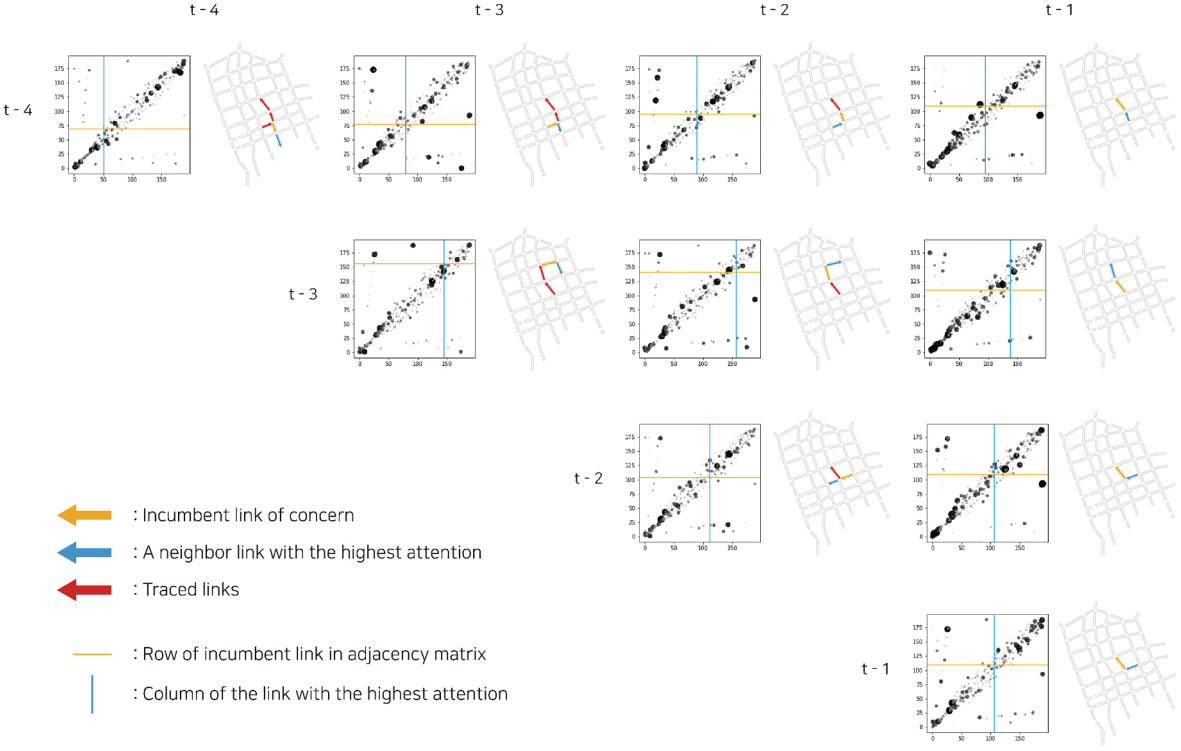


Fig. 11. Tracing a path or tree of traffic propagation.

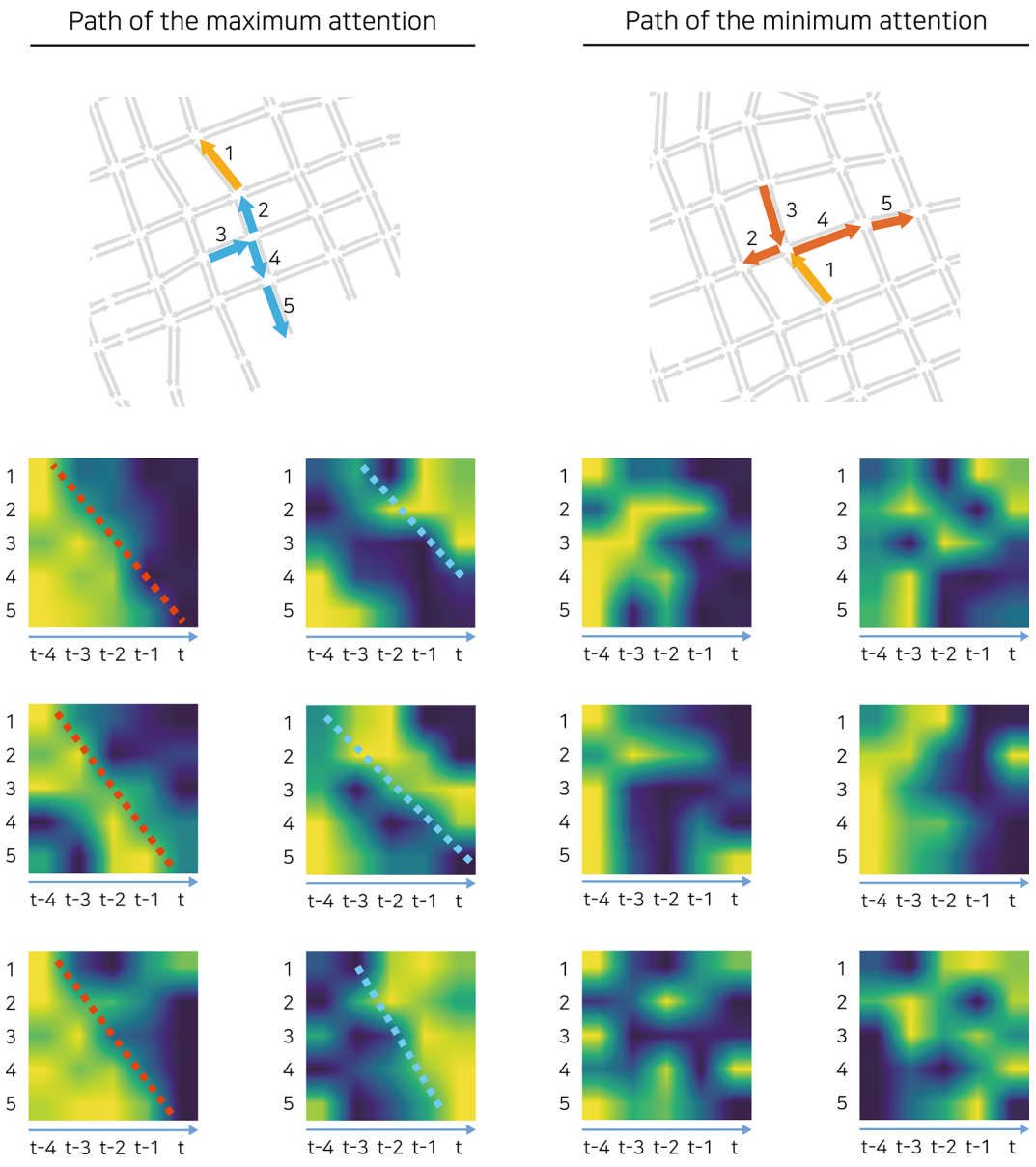


Fig. 12. Traffic propagation patterns along paths traced by the maximum and minimum of attention values.

To more intuitively explain how the traffic state of the chosen link propagates in a time–space, a series of links was traced across adjacency matrices for different time periods by selecting the link with the greatest attention value among the first-order neighbors. The traced links constituted a path or tree, which formed traffic propagation patterns. Fig. 11 shows how the path was traced back using the map of the testbed. For the top row stream of Fig. 11, the process began with the most recent time interval (=t − 1), and the column corresponding to the link with the maximum attention (=blue line) was found for the row corresponding to the link of concern (=yellow line). In the next time interval (=t − 2), the column with the maximum attention was chosen again for a row of the link that had been previously found to have the maximum attention. This process was repeated until the maximum time lag (=t − 4) was reached. In the same manner, paths of traffic propagation were found for lower row streams, as shown in Fig. 11.

To find the relationship between real traffic propagation pattern and the trace of the maximum attention, time–space diagrams along the path of the maximum attention were drawn using real speed data in a randomly chosen date (see Fig. 12). The horizontal axis represents time, and the vertical axis represents space (=traced links). The time axis is confined to the largest time lag (=t-4) as in the top row of Fig. 11. In the diagrams, the blue color represents low speed, and the yellow color represents high speed. Six diagrams for the maximum attention in the left two columns of Fig. 12 show some regular patterns of forming and releasing traffic congestion. Traffic congestion propagates through the traced links of the maximum attention in turn. Traffic congestion first takes place in the target link (=link 1) and then propagates into the next link of the maximum attention along the red dotted line, and

traffic congestion tends to be released from the target link to the next link of the maximum attention along the blue dotted line. Although these traffic propagation patterns are not observed every time traffic congestion is formed or released, it is obvious that there are some correlations between actual traffic propagation and the change in traffic states along the path of the maximum attention. On the other hand, no regular pattern of traffic propagation was found for the path of the minimum attention during the same periods (see diagrams in the right two columns of Fig. 12). This also supports that there are some correlations between actual traffic propagation and that along the path of the maximum attention.

6. Conclusions

The present study devised graph-based learning models for traffic forecasting that mimicked real traffic propagation based on variable adjacency matrices. The connectivity intensity of neighbor links was parametrized, and the intensity was estimated from a large amount of observed data. Furthermore, properties of road links such as capacity and length were also incorporated via a graph convolution model. To the best of our knowledge, effort to incorporate the prior domain knowledge into deep learning is rare in traffic forecasting studies.

In the present study, such an effort resulted in considerable improvement in forecasting accuracy when compared with the existing graph convolution models previously adopted in the field of traffic forecasting. Moreover, the computing time for the model evaluation and training was much shorter than that for any other reference models. This means that the proposed model conveys a great advantage when applied to a real-time traffic forecasting system. Unfortunately, a generative adversarial framework did not enhance the model performance, which was based solely on prediction accuracy, although the possibility to increase the prediction reality was expected.

In future study we plan to parametrize the number of time lags to represent the actual propagation time within a model framework. The model architecture will be varied while training in order to optimize the number of time lags. This scheme must adopt a methodology that can automatically determine hyper-parameters from the data. If successful, a more realistic traffic propagation scheme is expected to be developed by using a deep-learning model.

Acknowledgement

This research was supported by the Chung-Ang University Research Scholarship Grants in 2019 and also the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (2018R1A2B2004092).

References

Abu-El-Haija, S., Perozzi, B., Al-Rfou, R., Alemi, A., 2017. Watch Your Step: Learning Node Embeddings via Graph Attention. In: Advances NIPS 2018. Montreal, Canada, pp. 9197–9207.
Arjovsky, M., Chintala, S., Bottou, L., 2017. Wasserstein GAN. arXiv:1701.07875 [stat.ML].
Baek, J., Sohn, K., 2016. An investigation into passenger preference for express trains during peak hours. Transportation 43 (4), 623–641.
Bruna, J., Zaremba, W., Szlam, A., LeCun, Y., 2013. Spectral Networks and Locally Connected Networks on Graphs. ICLR 2014, Banff, Canada.
Cao, X., Zhong, Y., Zhou, Y., Wang, J., Zhu, C., Zhang, W., 2018. Interactive temporal recurrent convolution network for traffic prediction in data centers. IEEE Access 6, 5276–5289.
Chandra, S.R., Al-Deek, H., 2009. Predictions of freeway traffic speeds and volumes using vector autoregressive models. Journal of Intelligent Transportation Systems 13 (2), 53–72.
Chang, Daeyeol, Sohn, Keemin, 2015. Commuter dependence on expressways when travelling to work. Proc. Inst. Civil Eng. – Transp. 168 (1), 23–33. https://doi.org/10.1680/tran.13.00008.
Cui, Z., Henrickson, K., Ke, R., Wang, Y., 2018. Traffic Graph Convolutional Recurrent Neural Network: A Deep Learning Framework for Network-Scale Traffic Learning and Forecasting. arXiv:1802.07007 [cs, LG].
Defferrard, M., Bresson, X., Vandergheynst, P., 2016. Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering. NIPS 2017, Barcelona, Spain, 3844–3852.
Dell'Orco, M., Marinelli, M., 2017. Modeling the dynamic effect of information on drivers' choice behavior in the context of an Advanced Traveler Information System. Transport. Res. Part C: Emerg. Technol. 85, 168–183.
Gafni, O., Wolf, L., Taigman, Y., 2019. Vid2Game: controllable characters extracted from real-world videos. arXiv:1904.08379 [cs.LG].
Genders, W., Razavi, S., 2016. Using a deep reinforcement learning agent for traffic signal control. arXiv:1611.01142 [cs.LG].
Gori, M., Monfardini, G., Scarselli, F., 2005. A new model for learning in graph domains. IJCNN 2005, Montreal, Canada, 729–734.
Huang, W., Song, G., Hong, H., Xie, K., 2014. Deep architecture for traffic flow prediction: deep belief networks with multitask learning. IEEE Trans. Intell. Transport. 15 (5), 2191–2201.
Isola, P., Zhu, J.-Y., Zhou, T., Efros, A.A., 2016. Image-to-Image Translation with Conditional Adversarial Networks. CVPR 2017, Honolulu, Hawaii, pp. 1125–1134.
Jo, D., Yu, B., Jeon, H., Sohn, K., 2018. Image-to-image learning to predict traffic speeds by considering area-wide spatio-temporal dependencies. IEEE Trans. Vehic. Technol. 68 (2), 1188–1197.
Kamarianakis, Y., Prastacos, P., 2005. Space-time modeling of traffic flow. Comput. Geosci. 31 (2), 119–133.
Ke, J., Zheng, H., Yang, H., Xiqun, Chen, 2017. Short-term forecasting of passenger demand under on-demand ride services: a spatio-temporal deep learning approach. Transport. Res. Part C: Emerg. Technol. 85, 591–608.
Kipf, T.N., Welling, M., 2016. Semi-supervised classification with graph convolutional networks. ICLR 2017, Toulon, France.
Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., Shi, W., 2016. Photo-realistic single image super-resolution using a generative adversarial network. arXiv:1609.04802 [cs.CV].
Lei, K., Qin, M., Bai, B., Zhang, G., Yang, M., 2019. GCN-GAN: A non-linear temporal link prediction model for weighted dynamic networks. IEEE INFOCOM 2019, Paris, France, pp. 388–396.
Li, C., Wand, M., 2016. Precomputed Real-Time Texture Synthesis with Markovian Generative Adversarial Networks. ECCV 2016, Amsterdam, Netherlands.
Lee, J., Roh, S., Shin, J., Sohn, K., 2019. Image-based learning to measure the space mean speed on a stretch of road without the need to tag images with labels. Sensors 19 (5), 1227.
Liang, X., Du, X., Wang, G., Han, Z., 2019. A deep reinforcement learning network for traffic light cycle control. IEEE Trans. Vehic. Technol. 68 (2), 1243–1253.

Lin, Y., Dai, X., Li, L., Wang, F., 2019. Pattern sensitive prediction of traffic flow based on generative adversarial framework. IEEE Trans. Intell. Transport. Syst. 20 (6), 2395–2400.
Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F., 2015. Traffic flow prediction with big data: a deep learning approach. IEEE Trans. Intell. Transport. Syst. 16 (2), 865–873.
Mai, T., Ghosh, B., Wilson, S., 2012. Multivariate short-term traffic flow forecasting using bayesian vector autoregressive moving average model. TRB 91st Annual Meeting, Washington, D.C.
Ma, X., Tao, Z., Wang, Yinhai, Yu, H., Wang, Yunpeng, 2015. Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transport. Res. Part C: Emerg. Technol. 54, 187–197.
Owens, A., Isola, P., McDermott, J., Torralba, A., Adelson, E.H., Freeman, W.T., 2015. Visually indicated sounds. CVPR 2016, Nevada, Las Vegas, pp. 2405–2413.
Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G., 2009. The graph neural network model. IEEE Trans. Neural Netw. 20 (1), 61–80.
Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R., n.d. Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15(1), 1929–1958.
Stathopoulos, A., Karlaftis, M., 2003. A multivariate state space approach for urban traffic flow modeling and prediction. Transport. Res. Part C: Emerg. Technol. 11 (2), 121–135.
Tan, H., Xuan, X., Wu, Y., Zhong, Z., Ran, B., 2016. A comparison of traffic flow prediction methods based on DBN. CICTP 2016, Shanghai, China, pp. 273–283.
Ulyanov, D., Vedaldi, A., Lempitsky, V., 2016. Instance normalization: the missing ingredient for fast stylization. arXiv:1607.08022 [cs.CV].
Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2017. Graph attention networks. ICLR 2018, Vancouver, Canada.
Williams, B., Hoel, A., 2003. L. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: theoretical basis and empirical results. J. Transport. Eng. 129 (6), 664–672.
Xie, S., Huang, X., Tu, Z., 2015. Top-down learning for structured labeling with convolutional pseudoprior. ECCV 2016, Amsterdam, Netherlands, pp. 302–317.
Yang, Y., Yao, E., Yang, Z., Zhang, R., 2016. Modeling the charging and route choice behavior of BEV drivers. Transport. Res. Part C: Emerg. Technol. 65, 190–204.
Yu, B., Yin, H., Zhu, Z., 2017. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework For Traffic Forecasting. arXiv:1709.04875 [cs, LG].
Yu, H., Wu, Z., Wang, S., Wang, Y., Ma, X., 2017. Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks. arXiv:1705.02699 [cs.LG].
Zhang, J., Shi, X., Xie, J., Ma, H., King, I., Yeung, D.-Y., 2018. GaAN: gated attention networks for learning on large and spatiotemporal graphs. UAI 2018, Monterey, California, USA.
Zhang, Y., Yao, E., Zhang, J., Zheng, K., 2018b. Estimating metro passengers' path choices by combining self-reported revealed preference and smart card data. Transport. Res. Part C: Emerg. Technol. 92, 76–89.
Zhao, L., Song, Y., Zhang, C., Liu, Y., Wang, P., Lin, T., Deng, M., Li, H., 2018. T-GCN: A temporal graph convolutional network for traffic prediction. arXiv:1811.05320 [cs, LG].
Zhu, J.-Y., Park, T., Isola, P., Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. ICCV 2017, Venice, Italy, pp. 2223–2232.