Online Traffic Speed Estimation for Urban Road Networks with Few Data: A Transfer Learning Approach

James J.Q. Yu, *Member, IEEE*

***Abstract*— Online traffic speed data of urban road networks serve as the foundation of modern intelligent transportation systems. Much research has been conducted on developing methods, mostly model-based or machine learning ones, to estimate the data with GPS record for one, few adjacent roads, or the entire vehicular transportation network. While the machine learning methods generally yield satisfactory estimation accuracy, their accomplishments are established on a plethora of historical GPS records which may not be readily available for many urban transportation systems. In this paper, we investigate a transfer learning approach to provide speed data estimations with few data. We ground this work on a graph convolutional generative autoencoder that can generate the estimations for an entire transportation network in one go, and modify its internal computation graph to reduce the size of network topology-dependent model parameters. Subsequently, pre-trained models from road networks with massive historical data can be re-used in other networks with few data, which are only employed to adjust a small number of parameters. To assess the effectiveness of the proposed approach, comprehensive case studies are conducted, in which outstanding speed estimations can be obtained with significantly shorter training time.**

I. Introduction

TIMELY and accurate estimation of the traffic speed in urban road networks is among the most important features of the intelligent transportation system (ITS) in modern smart city developments [1]. It serves as the foundation of higher-level ITS applications, since such data helps governments, companies, and research institutes to better understand human behaviors, plan transportation system constructions, and operate city transits [2]. For instance, real-time route planning of contemporary urban city services heavily rely on the real-time traffic speed to develop optimal routes, which at the same time also contributes to the social welfare, e.g., less pollutant emission and traffic congestion [3], [4].

Motivated by the importance of real-time traffic speed estimation to urban road networks, much effort has been devoted to developing solutions for providing such information [5]. In this context, industry and academia take different approaches. The former – Google and Uber for example – makes use of its massive users and customers to gather real-time vehicular GPS records containing the position and speed of crowd-sourcing vehicles [6]. The record data can be easily fused to construct a traffic speed map, which is later employed to provide real-time routing services. However, such approaches are established on the availability of huge crowd-sourcing vehicle information, which is not always possible for other companies or services.

On the other hand, recent research effort focused on employing either traffic-model-based or black-box statistical methods to provide speed estimation with relatively scarce speed measurements, e.g., [7]–[10]. Either class of methods has its own advantages over the other. While model-based approaches can be better adopted in what-if reasoning due to their transparency and interpretability, black-box approaches are generally more robust to data noise and perturbations. Recent approaches in either class demonstrated speed estimations with satisfactory accuracy and timeliness, even compared with industrial crowd-sourcing speed data. Nonetheless, these approaches concentrate on only one or few adjacent roads at one time. In the context of modern large urban transportation networks, such approaches become highly computationally inefficient, and the transportation network topology information cannot be utilized, which leads to potential performance loss. To overcome these drawbacks, a recent work proposed a graph convolutional generative autoencoder (GCGA) model based on recent advance of deep learning techniques for real-time traffic speed estimation [11]. By adopting graph convolution operations, the approach is capable of developing urban road speed data considering the overall traffic condition according to the road topology. In addition, its generative adversarial design empowers the model to generate the speed estimation of an entire transportation network in one go.

However, there exists a significant problem to such black-box models. Most existing models are developed based on machine learning techniques, which require a large volume of historical traffic data to adjust model parameters. Such data may be available to a few large cities across the world, but can be uncommon for the rest due to infrastructure or information and communication technology limits. How to construct and fine-tune black-box models for such cities with few-shot data, i.e., data in a small volume, is a critical problem that hinders the deployment of these speed estimation approaches.

To solve this problem, in this work we propose a transfer learning-driven online traffic speed estimation approach for urban road networks, based on the GCGA deep learning model. We particularly consider the historical traffic data characteristics of different urban transportation and identify the topology-agnostic model parameters in GCGA. Different from existing work in which the speed estimators are trained with data from the same road networks, the proposed approach first pre-trains the model with data from the large

James J.Q. Yu is with the Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, China.

cities with sufficient historical records. Then selective parameters in the model is further fine-tuned using the few-shot data from the target city. As far as we are concerned, this is the pioneer work on transferrable online traffic speed estimation for entire urban road networks.

The rest of this paper is organized as follows. In Section II, we present the mathematical formulation of the traffic speed estimation problem. In Section III, the original design of GCGA is introduced, and how we make the model utilize the transferrable latent information from other road networks is elaborated. Section IV presents multiple case studies for performance assessment, and this paper is concluded in Section V.

II. Traffic Speed Estimation for Urban Road Networks

We model the investigated road network as a direct graph $\mathcal{G}(\mathcal{N}, \mathcal{E})$, in which $\mathcal{N}$ is the set of road intersections in the network, and $\mathcal{E}$ is the set of roads connecting the intersections. Given the topology of an arbitrary transportation system, the graph can be easily constructed by first identifying the locations of intersections, and then connecting them when there is a road. For each road $e \in \mathcal{E}$, we use $\mathrm{fr}(e)$ and $\mathrm{to}(e)$ to represent its starting and ending intersections, respectively.

Let $\mathcal{T} = \{\cdots, -2, -1, 0\}$ be the discrete time horizon of the past until the current time constituted by consecutive time instances, in which $t = 0$ refers to the current one. For each road $e \in \mathcal{E}$, we use $v_{e,t}$ to denote its average traffic flow speed at time t. Besides this time-variant data of roads, they also have a collection of time-invariant properties, e.g., driving speed limits and number of nearby point of interests, denoted by $\mathcal{P}_e$ for $e \in \mathcal{E}$. These properties greatly enhanced the information richness of the urban road network.

In traffic speed estimation, the most prominent objective is to obtain the speed map of urban road networks $\mathcal{V}_0 = \{v_{e,0} | \forall e \in \mathcal{E}\}$. While the speed values can be directly obtained from stationary speed sensors, it is economically inefficient to equip them at all available roads in the networks. It is rather common that the rare speed sensor data are integrated with the real-time GPS records of moving vehicles to construct the speed map. Let $\mathcal{E}^S \subseteq \mathcal{E}$ be the set of roads with stationary speed sensors, and $\mathcal{E}_t^+ \subseteq \mathcal{E}$ by the set of roads whose real-time speed at time t can be obtained from vehicular GPS records. Since GPS records that can contribute to inferring road speeds are typically generated on the move, $\mathcal{E}_t^+$ is dynamic with respect to a changing t. Let $\mathcal{E}_t^- = \mathcal{E} \setminus (\mathcal{E}^S \cup \mathcal{E}_t^+)$ be the set of roads whose speed at time t cannot be directly obtained. Consequently, the traffic speed estimation problem for urban road networks can be expressed as follows:

$$\text{minimize MAPE} = \frac{1}{|\mathcal{E}_t^-|} \sum_{e \in \mathcal{E}_t^-} \frac{|\hat{v}_{e,0} - v_{e,0}|}{v_{e,0} + \varepsilon},$$

where ε is a small positive value[1] preventing the divide-by-zero issue, and $\hat{v}_{e,0}$ is the estimated speed of road e at time t, which is obtained by

$$\hat{\mathcal{V}}_0 = \mathrm{EST}(\mathcal{V}_t^+, \{\mathcal{P}_e | \forall e \in \mathcal{E}\}),$$

where $\hat{\mathcal{V}}_0 = \{\hat{v}_{e,0} | \forall e \in \mathcal{E}\}$, $\mathcal{V}_t^+ = \{v_{e,t} | \forall e \in \mathcal{E}^S \cup \mathcal{E}_t^+\}$ and EST is the speed estimator.

[1] $\varepsilon \equiv 0.01\,\mathrm{km\,h^{-1}}$ in the case studies.

III. Transferable Graph Convolution Generative Autoencoder

As introduced in Section I, the ever-increasing size of urban transportation networks is introducing great computational burden to city-wise road speed estimation. In this section, we first introduce a graph-based convolutional generative neural network to effectively develop speed estimation data in real-time for urban road networks. Then we investigate a latent information transfer mechanism for the neural network in order to further alleviate the computational effort required. Finally, the detailed model training and testing implementations are discussed.

A. Graph Convolution Operation

When estimating urban road speeds, there is an intuition that roads shall have more prominent influence on their connecting neighborhoods than distant ones, and the influence may decay rapidly when diffusing over the network. This intuition has been justified in the previous literature, and can be realized by the concept of graph convolution [12]. This operation aims at extracting features from graphs, and puts extra emphasis on the data correlation along graph arcs. By *convolution*, graph convolution operation performs neighborhood data fusion on the input source data with a receptive field design according to the adjacency matrix of the graph. Therefore, the output data corresponds to an arbitrary vertex in the graph is in fact the result of shared information from its graph neighbors. Furthermore, multiple sequential graph convolution operations correspond to larger receptive fields based on the self-multiplying adjacency matrices, leading to a data fusion process of a wider range [13].

We adopt the graph convolution operation in [12] to perform the convolution on urban road networks. While the manipulation agent of this operation is the vertices of graphs, we are more interested in the arc features, i.e., road data, when estimating the traffic speed. This incompatibility can be resolved by transforming the original urban road network into its equivalent road-connectivity graph, in which each vertex represents a road, and the connectivity indicates road adjacency information. Specifically, a new undirected graph $\mathcal{R}(\mathcal{E}, \mathcal{A})$ can be constructed by

$$\mathcal{A} = \{(e_1, e_2) \big| \big|\{\mathrm{fr}(e_1), \mathrm{to}(e_1)\} \cap \{\mathrm{fr}(e_2), \mathrm{to}(e_2)\}\big| \neq 0, \\ \forall e_1, e_2 \in \mathcal{E}\}. \tag{1}$$

Subsequently, graph convolution operation takes the $|\mathcal{P}| + 1$ road features (speed and properties) of $|\mathcal{E}|$ roads as input and develops F latent data features of the roads using the adjacency matrix A of $\mathcal{R}$ according to the following propagation rule:

$$Z = \mathrm{GC}(X, A) = \sigma(\hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} X W + b), \tag{2}$$

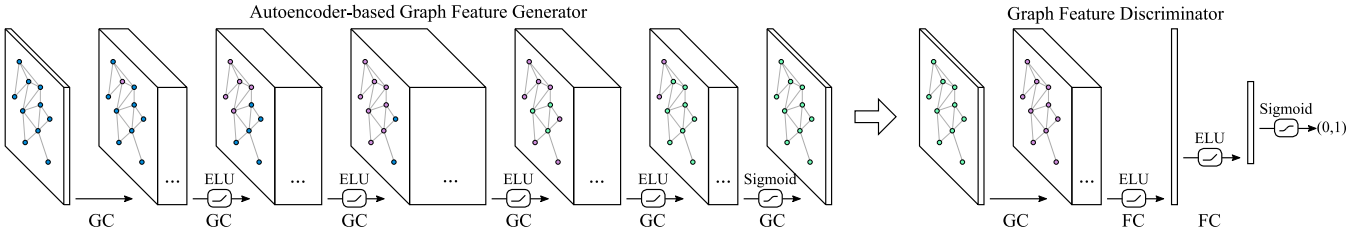

Fig. 1. Illustration of GCGA model. Adopted from [11] with modifications.

where $X \in \mathbb{R}^{|\mathcal{E}|\times(|\mathcal{P}|+1)}$ and $Z \in \mathbb{R}^{|\mathcal{E}|\times F}$ are the input and output matrices, $\hat{A} \in \mathbb{B}^{|\mathcal{E}|\times|\mathcal{E}|} = \{\hat{a}_{ij}\} = A + I$, $\hat{D} = \text{diag}\{\hat{a}_{11}, \hat{a}_{22}, \cdots\}$, and $\sigma(\cdot)$ is a non-linear activation function. In the propagation, two sets of tunable parameters are adopted, namely $W \in \mathbb{R}^{(|\mathcal{P}|+1)\times F}$ and $b \in \mathbb{R}^{|\mathcal{E}|\times F}$. When multiple graph convolution operations are chained, the number of input features from the second operation should be set to the number of output one of the previous operation. For instance, if the output of the first operation is of size $|\mathcal{E}| \times F$, then the number of input features for the second graph convolution operation should also be F. Note that in this case, there is no hard limit on the number of output features for the second operation. This propagation rule is motivated by the first-order approximation of Chebyshev polynomials of eigenvalues in the spectral domain [14]. It has been shown that such approximation can result in highly competitive performance for graph feature learning [12], [13].

B. Graph Convolution Generative Autoencoder

Empowered by the graph convolution operation, a GCGA model was proposed to provide real-time estimation of traffic speed data in urban road networks in [11]. The model inherits the design principle of the Generative Adversarial Network (GAN) [15] and autoencoder, in which two sub-neural-networks form a two-player minimax game to generate artificial speed data of networks that is indistinguishable from the ground truths. Fig. 1 presents an illustration of GCGA model. An autoencoder-based graph feature generator is employed to develop $\hat{\mathcal{V}}_0$ with input available data at the current time instance, i.e., $\mathcal{V}_0^+$ and $\{\mathcal{P}_e|\forall e \in \mathcal{E}\}$[2]. This can be achieved thanks to the autoencoder-like feature encoding-decoding architecture of the generator. At the beginning of feature generation step, the normalized available data is input into the network with missing data entries set to zeros. The three consecutive up-sampling graph convolution operations enrich the data by producing 128, 256, and 512 features for each road, respectively, which receives diffused information from its neighborhood road(s) along the graph. Subsequently, the $|\mathcal{E}|\times 512$ feature matrix is decoded by three other down-sampling graph convolution operations, each of which reduces the number of features to 256, 128, and finally 1, respectively. The final $|\mathcal{E}| \times 1$ output matrix, after being activated by an element-wise sigmoid function, resembles the normalized complete road network speed data $\hat{\mathcal{V}}_0$.

[2]We adopt following road features as $\mathcal{P}_e$, i.e., traffic speed limit, road length and width, number of lanes, and number of point-of-interest sites along the road.

Then a graph feature discriminator is constructed to assess the authenticity of the generated $\hat{\mathcal{V}}_0$ with the prior knowledge of the speed data pattern in $\mathcal{V}_t^+, \forall t \in \mathcal{T}$. This network is a stack of a 128-feature graph convolution layer and three fully-connected neuron layers with 1024, 128, and 1 neurons. Finally, the discriminator outputs a $(0, 1)$ value indicating the discriminators opinion on the authenticity of $\hat{\mathcal{V}}_0$. When fine-tuning the parameters W and b, both sub-networks play the minimax game with the following objective function:

$$\min_{\theta^G}\max_{\theta^D} L = \mathbb{E}[\log \mathrm{D}(\mathcal{V}_0^+, \theta^D)] + \mathbb{E}[\log(1 - \mathrm{D}(\mathrm{G}(\mathcal{V}_0^+, \theta^G), \theta^D))], \quad (3)$$

where θ^G and θ^D are the sets of parameters for the generator and discriminator, $\mathrm{G}(\cdot, \theta^G)$ and $\mathrm{D}(\cdot, \theta^D)$ are the loss functions of the generator and discriminator, respectively. At the beginning of this training process, θ^G is initialized to random values, which make the discriminator reject $\mathrm{G}(\mathcal{V}_0^+, \theta^G)$ easily, rendering a large L value. The training algorithm then tries to adjust θ^G to generate realistic speed data to compromise the discriminator (minimize L), while adjusting θ^D in turn to distinguish the better data produced by the generator (maximize L). This process repeats until the generator is good enough to fool the discriminator, and the produced speed data can resemble the ground truth.

When θ^G is well-adjusted, the model can be applied to estimate the speed data of urban road networks. Similar to the training process, the incomplete available data is input into the generator, which utilizes the fixed θ^G values to calculate $\hat{\mathcal{V}}_0$. Instead of assessing its authenticity with the discriminator, this estimation is considered a close approximation of $\mathcal{V}_0$, and can be used in subsequent ITS applications [11].

C. Latent Information Transfer

A key issue exists in training the GCGA model is the scarcity of historical data $\mathcal{V}_t^+, \forall t \in \mathcal{T}$. While it is possible that data from a considerable number of past time instance is available, the number of roads covered in $\mathcal{E}^S \cup \mathcal{E}_t^+$ is far less than $|\mathcal{E}|$, rendering highly sparse training data. This issue can be resolved if the latent information extracted from one urban road network can be transferred to other road networks. In such a case, GCGA can be firstly pre-trained on large data-sets with "denser" traffic data of benchmark road networks. The resulting model is subsequently further fine-tuned with the much smaller data-set of the target road network. It can be expected that both the training time and the estimation performance can be greatly improved compared with directly training GCGA on the small data-set.

Nonetheless, the above analysis is established on the hypothesis that the latent information of different road networks can be shared. This is mostly true with respect to the graph convolution operation. By inspecting its propagation rule (2), one may notice that the size of parameter W, i.e., $(|\mathcal{P}|+1)\times F$ is actually unrelated to the road network size, i.e., $|\mathcal{E}|$. This implies that W is capable of extracting network topology-agnostic information from the input data. While the information is closely related to the types of connections among the roads, it shall demonstrate similar characteristics or probability distributions on different road networks.

On the other hand, the other parameter b is related to $|\mathcal{E}|$, and encloses road network topological information. Meanwhile, we may reduce the size of topology-related parameter set in order to alleviate the second-step training burden. This can be achieved by separating b into multiple sub-parameters:

$$b \equiv B\kappa\mu,$$

where $B \in \mathbb{R}^{|\mathcal{E}|\times K}$, $\kappa \in \mathbb{R}^{K\times F}$, and $\mu^{F\times F}$ are the three sub-parameters, and K is a new user-defined hyper-parameter that defines the size of the sub-parameters, similar to the definition of F. In such a way, both κ and μ is not related to the road network size, and fine-tuning B can be significantly more efficient than b, given a small K. Consequently, the propagation rule of the graph convolution operation is amended as follows:

$$Z = \mathrm{GC}(X, A) = \sigma(\hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}XW + B\kappa\mu). \quad (4)$$

D. Model Training and Transfer Learning

Before using the information-transferrable GCGA in developing traffic data, the network parameters, i.e., W, B, κ, μ of each graph convolution operation and weight/bias in the fully connected layers, need to be properly adjusted with respect to the networks of both dense (called *pre-trainig* data) and sparse traffic data (called *training* data). Specifically, both data sets comprise samples of incomplete speed data $\mathcal{V}_t^+$. We first augment the samples by artificially remove speed data of random roads from one $\mathcal{V}_t^+$ and create M new samples, denoted by $\mathcal{V}_{t,m}^-$ where m is the index of the sample among M. Then we adjust the network parameters of both the feature generator and discriminator iteratively with the pre-training data. In each iteration, $\mathcal{V}_{t,m}^-$ is first input into the generator to develop $\hat{\mathcal{V}}_{t,m}^-$, which is an estimate of $\mathcal{V}_t^+$. The mean squared error (MSE) between $\hat{\mathcal{V}}_{t,m}^-$ and $\mathcal{V}_t^+$ on those artificially removed roads is computed, denoted by $\mathrm{MSE}_{t,m}$ The averaged MSE over all samples, i.e.,

$$L^G = \frac{1}{|\mathcal{T}|M}\sum_{t\in\mathcal{T}}\sum_{m=1}^{M}\mathrm{MSE}_{t,m}$$

is considered as the loss function of the generator. Subsequently, both $\mathcal{V}_t^+$ and $\hat{\mathcal{V}}_{t,m}^-$ are input into the discriminator, and the binary cross entropy loss function is adopted to assess its capability of identifying input samples that are artificially generated instead of selected from the (pre-)training data set:

$$L^D = -\sum_{n=1}^{N}[y_n\log\hat{y}_n + (1-y_n)\log(1-\hat{y}_n)],$$

where N is the total number of tested samples, y_n and $\hat{y}_n$ are the truth and inferred authenticity assessments of tested samples, respectively. The network parameters can be optimized by Adam [16].

After training GCGA model with the pre-training data-set, the network parameters can represent both topology-agnostic and topology-dependent features of the input road network. If the model is supposed to be transferred to another urban road network, we only fine-tune the topology-related network parameters, i.e., B, with the training data instead of re-training all parameters from scratch. The only difference of this second training process from the previous one is that all other network parameters except for B is set constant. In such cases, the relatively small training data set is still sufficient to adjust values of B, which is significantly smaller than $\theta^G \cup \theta^D$ according to the discussion in Section III-C. Finally, the newly tuned GCGA model can be adopted to provide estimations on the new urban road network, and the latent information from the original network is transferred to the new one.

IV. Case Studies

To fully evaluate the performance of the proposed transferrable GCGA model in estimating the traffic speed of urban road networks with sparse data, we conducted two comprehensive case studies with real-world data sets. We first investigate the performance and computation speed improvement of the latent information transfer mechanism. Then we assess the influence of the control parameters, and evaluate how the GCGA model structure impact the transfer performance.

A. Data Sets and Simulation Configurations

In this work, we adopt the real-world traffic speed data of four major cities in China for investigation, whose major road networks are shown in Fig. 2. Specifically, we adopt the traffic data of 1386 roads in Beijing during the year 2018 as the pre-training data set, which comprises 105 120 time instances and more than 100 million data entries. In addition, we also employ the traffic data of Shanghai, Guangzhou, and Shenzhen during the thirty days in November 2018 as three independent training data sets, each of which has 1522, 1024, and 400 roads in the set, respectively. Each training data set comprises 8640 time instances, which is way smaller than the Beijing data. To emulate real-world cases that stationary speed sensors and crow-sourced vehicular GPS records are relatively rare, we only retain a random number of data entries in all time instances during training. This data retention rate is denoted by α, and set to 15% unless otherwise stated. For cross-validation, all data sets are grouped into three non-overlapping sub-sets, i.e., a training set with 50% of all time instances for network parameter tuning, a validation set with 25% instances for training

Fig. 2. Illustration of latent information transfer among cities.

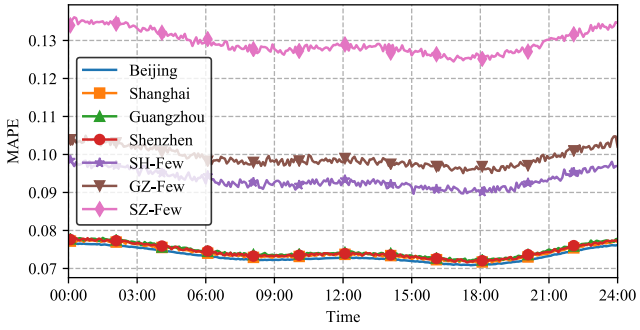

Fig. 3. Illustration of latent information transfer among cities.

process termination, and a testing set with the rest 25% instances for performance assessment, which is conducted on the removed roads in these instances using MAPE. The validation set is evaluated after each training epoch in which each instances from the training set is adopted to tune the parameters, and the training process terminates when validation set MAPE does not reduce for a consecutive three epochs.

When pre-training and training GCGA, the control parameters $K = 128$ and $M = 8$. Training data is employed in mini-batches with size 32, and batch-normalization [17] is employed after each neural layers in GCGA. The model is implemented with PyTorch [18], and all simulations are conducted on Amazon EC2 P3 instances equipped with nVidia Tesla V100 GPUs for neural network evaluation acceleration.

B. Estimation Accuracy and Training Time

The main purpose of the information transfer mechanism is to address the few-shot training data issue for urban road networks. The accuracy of the estimated speed data is among the most important metric in evaluating the efficacy of the proposed mechanism, with training time another key concern. In particular, we compare the MAPE performance of the proposed mechanism with the direct training approach in which the small training data sets of Shanghai, Guangzhou, and Shenzhen are employed to train GCGA from scratch without the aid of Beijing data pre-training. Since traffic speed data tend to demonstrate stronger daily patterns than other common time scales, we present the MAPE of every five minutes in a day of either approach on all four urban road networks in Fig. 3. In this figure, each data point is the averaged MAPE over 30 days and all roads in the respective road network except for "Beijing", which is the averaged result over 365 days. For the plots labeled by "Shanghai", "Guangzhou", and "Shenzhen", GCGA is pre-trained with Beijing data, and further fine tuned with the traffic data of the respective city. For the other plots end with "-Few" postfix, only the few-shot data of the respective city, i.e., Shanghai, Guangzhou, and Shenzhen, are used to train a GCGA.

TABLE I
COMPARISON PRE-TRAINING AND TRAINING TIME

City	Time with pre-train	Time w/o pre-train
Shanghai	1.3 h	7.4 h
Guangzhou	1.4 h	5.7 h
Shenzhen	0.7 h	4.3 h
Beijing	-	57.1 h

From the simulation results it is obvious that the transferred information from Beijing data set greatly improves the model quality when generating traffic speed data for the other cities. On average, pre-training boosted GCGA achieves satisfactory MAPE at 7.41%, 7.46%, and 7.44% for the three few-shot cities, while the baseline MAPE for Beijing is 7.33%. The performance can be deemed outstanding when compared with the few-shot training results, i.e., 9.34%, 9.92%, and 12.95% for the three cities, respectively. Since the only difference between these two training approaches is the pre-training process, the notable improvement can be credited to the transferred latent information.

The training time for above pre-training and training processes are recorded in Table I. The comparison clearly indicates that pre-training GCGA model can help reduce the subsequent training time greatly. This is due to the significantly smaller parameter size, i.e., $|B|$ compared with $|\theta^G \cup \theta^D|$, to be adjusted after pre-training. While training a GCGA model with Beijing data costs notably more time than the others, the trained model can be both used to develop speed data for Beijing and employed as the pre-training model for other cities. To conclude, it is preferable to make use of pre-trained GCGA model on large historical data set than training the same model from scratch with few-shot data in terms of both estimation accuracy and training time.

C. Parameter Sensitivity Test and Model Structure

In previous simulations, we set the control parameters $K = 128$ and $M = 8$. In this subsection, we investigate GCGA's sensitivity on them. Specifically, we test

TABLE II
RESULTS OF PARAMETER SENSITIVITY TEST

City	Metric	K				M			
		32	64	128	256	2	4	8	16
Beijing	MAPE	8.62%	7.98%	7.33%	7.27%	10.44%	8.01%	7.33%	7.35%
	Time	48.3 h	52.0 h	57.1 h	72.4 h	18.7 h	30.8 h	57.1 h	112.2 h
Shanghai	MAPE	8.38%	7.93%	7.41%	7.39%	14.06%	8.84%	7.41%	7.40%
	Time	1.2 h	1.2 h	1.3 h	1.6 h	0.3 h	0.7 h	1.3 h	2.5 h
Guangzhou	MAPE	8.50%	7.87%	7.46%	7.45%	12.30%	8.41%	7.46%	7.45%
	Time	1.2 h	1.3 h	1.4 h	1.6 h	0.4 h	0.7 h	1.4 h	2.8 h
Shenzhen	MAPE	8.14%	7.65%	7.44%	7.41%	18.77%	8.92%	7.44%	7.37%
	Time	0.6 h	0.7 h	0.7 h	1.0 h	0.2 h	0.4 h	0.7 h	1.4 h

the generated urban speed estimation accuracy and training time on various parameter configurations, i.e., $K \in \{32, 64, 128, 256\}$ and $M \in \{2, 4, 8, 16\}$. All other settings are identical to those in Section IV-B, and the simulation results are presented in Table II. The table implies that while increasing either K or M can reduce the averaged MAPE on all cities, the training time also increases roughly linearly with respect to the parameter value. In the meantime, reducing either parameter below the default value leads to reduced training time at the expense of deteriorated estimation MAPE. Since the performance improvements for $K > 128$ and $M > 8$ is not as significant as the proportionally growing training time, $K = 128$ and $M = 8$ are a pair of best performing control parameters considering both the estimation accuracy and training time.

V. CONCLUSIONS

In this work, we propose a latent information transfer mechanism for online urban road network speed estimation with few-shot historical data. The transfer can be achieved by first modifying the propagation rule of graph convolution operation in the GCGA model and then minimizing the size of network topology-dependent model parameters. Specifically, the original parameter b is expressed as the multiple of three new parameters, i.e., $B\kappa\mu$, in which the topology-dependent parameter B is generally smaller than b. This design not only reduces the computation complexity of adjusting b in GCGA, but also enables other topology-agnostic parameters to extract and represent transferrable information between road networks. The new model, when applied to estimate data for road networks with few data, can be firstly pre-trained on other road networks with large historical data sets. Subsequently, the few data from the target networks are only employed to adjust the values of new parameter B. To evaluate the proposed mechanism, we conducted a series of case studies on three cities in China with few-shot data, i.e., Shanghai, Guangzhou, and Shenzhen. The abundant historical data from Beijing, China is employed for pre-training. Simulation results demonstrate that the proposed mechanism can develop outstanding speed data estimations compared with the no-pre-training method, and the training time can also be reduced. Finally, we investigate the sensitivities of the model parameters.

REFERENCES

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

[2] J. Zhang, F. Y. Wang, K. Wang, W. H. Lin, X. Xu, and C. Chen, "Data-driven intelligent transportation systems: a survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1624–1639, Dec. 2011.

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

[4] J. J. Q. Yu and A. Y. S. Lam, "Autonomous vehicle logistic system: joint routing and charging strategy," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 7, pp. 2175–2187, Jul. 2018.

[5] R. Sundar, S. Hebbar, and V. Golla, "Implementing intelligent traffic control system for congestion control, ambulance clearance, and stolen vehicle detection," *IEEE Sensors J.*, vol. 15, no. 2, pp. 1109–1113, Feb. 2015.

[6] "Official Google blog: the bright side of sitting in traffic: crowdsourcing road congestion data," accessed Feb. 2018. [Online]. Available: https://googleblog.blogspot.hk/2009/08/bright-side-of-sitting-in-traffic.html

[7] Y. Duan, Y. Lv, Y.-L. Liu, and F.-Y. Wang, "An efficient realization of deep learning for traffic data imputation," *Transportation Research Part C: Emerging Technologies*, vol. 72, pp. 168–181, 2016.

[8] L. Li, X. Su, Y. Zhang, Y. Lin, and Z. Li, "Trend modeling for traffic time series analysis: an integrated study," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 6, pp. 3430–3439, Dec. 2015.

[9] C. P. I. J. van Hinsbergen, T. Schreiter, F. S. Zuurbier, J. W. C. van Lint, and H. J. van Zuylen, "Localized extended Kalman filter for scalable real-time traffic state estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 1, pp. 385–394, Mar. 2012.

[10] R. Wang, D. B. Work, and R. Sowers, "Multiple model particle filter for traffic estimation and incident detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 12, pp. 3461–3470, Dec. 2016.

[11] J. J. Q. Yu and J. Gu, "Real-time traffic speed estimation with graph convolutional generative autoencoder," *IEEE Trans. Intell. Transp. Syst.*, in press.

[12] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. International Conference on Learning Representations*, Toulon, France, Apr. 2017.

[13] J. Chen, T. Ma, and C. Xiao, "FastGCN: fast learning with graph convolutional networks via importance sampling," in *Proc. International Conference on Learning Representations*, Vancouver, Canada, May 2018.

[14] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Advances in Neural Information Processing Systems*, Barcelona, Spain, Dec. 2016.

[15] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Advances in Neural Information Processing Systems*, 2014, pp. 2672–2680.

[16] D. P. Kingma and J. Ba, "Adam: a method for stochastic optimization," in *Proc. International Conference on Learning Representations*, San Diego, CA, Dec. 2015.

[17] S. Ioffe and 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.

[18] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *Proc. Advances in Neural Information Processing Systems*, Long Beach, CA, Dec. 2017.