Learn Travel Time Distribution with Graph Deep Learning and Generative Adversarial Network

Xiaozhuang Song, Chenhan Zhang, *Student Member, IEEE*, and James J.Q. Yu, *Senior Member, IEEE*

***Abstract*— How to obtain accurate travel time predictions is among the most critical problems in Intelligent Transportation Systems (ITS). Recent literature has shown the effectiveness of machine learning models on travel time forecasting problems. However, most of these models predict travel time in a point estimation manner, which is not suitable for real scenarios. Instead of a determined value, the travel time within a future time period is a distribution. Besides, they all use grid structure data to obtain the spatial dependency, which does not reflect the traffic network's actual topology. Hence, we propose GCGTTE to estimate the travel time in a distribution form with Graph Deep Learning and Generative Adversarial Network (GAN). We convert the data into a graph structure and use a Graph Neural Network (GNN) to build its spatial dependency. Furthermore, GCGTTE adopts GAN to approximate the real travel time distribution. We test the effectiveness of GCGTTE with other models on a real-world dataset. Thanks to the fine-grained spatial dependency modeling, GCGTTE outperforms the models that build models on a grid structure data significantly. Besides, we also compared the distribution approximation performance with DeepGTT, a Variational Inference-based model which had the state-of-the-art performance on travel time estimation. The result shows that GCGTTE outperforms DeepGTT on metrics and the distribution generated by GCGTTE is much closer to the original distribution.**

I. Introduction

Recently, the continuously increasing number of vehicles and the expansion of road length have led to many critical challenges in the transportation system, e.g., the complex traffic scheduling and potential traffic congestion [1]. Thus, a robust and reliable Intelligent Transportation System (ITS) is urgently needed. One of the most critical issues of building ITS is to construct a precise travel time forecasting model for the roads in traffic networks, as it can provide an essential data guarantee for the decision-making of ITS [1].

However, building a robust travel time forecasting model is non-trivial. The main challenges in solving this problem are twofold. First, it is crucial to choose the proper approach to process the raw data and build the spatial dependency since each road's real-time traffic status is highly related to its topology structure. Existing works mainly use the position information of trajectory to model the spatial dependency [2], or partition the data into grids, and use spatial features of grids to obtain their spatial dependency [3]. Typically, spatial features of grids are input into a Convolutional Neural Network (CNN)-based model to construct its spatial dependency. However, traffic network has a graph topology structure rather than a grid structure. Since that, the grid structure data can not fully reflect the traffic network's underlying topology, and it possibly leads to a performance degradation [4].

Second, in existing literature on travel time forecasting, the proposed models primarily present point estimation results. However, the travel time within a period of time shall be formulated as a distribution [5]. For example, if we want to predict the travel time from campus to the subway station at 7 am every Monday, it is evident that the travel time can be influenced by various factors, e.g., bad weather and traffic congestion. Nonetheless, the point estimation models only provide a single value output, which can not show travel time's underlying distribution. In this case, forecasting travel time with a prediction model that produces travel time distribution results is more reasonable. Therefore, the point estimation model has a natural limitation for its lack of distribution estimation ability. Unfortunately, it is difficult to approximate the travel time distribution entirely since it is intractable. For instance, a classical way for approximating distribution is to use Monte Carlo sampling method [6]. However, distribution approximation methods based on Monte Carlo sampling are computationally expensive and may possibly lead to an overfitting problem [6].

To solve the above problems, we propose GCGTTE, a deep learning model based on the Graph Neural Network (GNN) and Generative Adversarial Network (GAN). The main contributions of this paper are as follows:

- We propose a novel travel time distribution estimation model built with graph deep learning and GAN. The results show that our model achieves better performance over baselines for travel time estimation task on a real-world dataset.
- Rather than a point estimation method, we approximate the underlying actual travel time distribution in an adversarial training style. Moreover, we conduct a case study by comparing GCGTTE with other models to explain the effectiveness of GCGTTE.
- We propose a practical embedding learning technique, which makes the model robust when handling sparse data.

The rest of this paper is organized as follows. Section II gives a brief introduction of literature related to this work. Section

This work is supported by the Stable Support Plan Program of Shenzhen Natural Science Fund No. 20200925155105002, by the General Program of Guangdong Basic and Applied Basic Research Foundation No. 2019A1515011032, and by the Guangdong Provincial Key Laboratory (Grant No. 2020B121201001). The authors are with the Guangdong Provincial Key Laboratory of Brain-inspired Intelligent Computation, Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen 518055, China. James J.Q. Yu is the corresponding author.

III presents the details of the proposed GCGTTE. Section IV presents the experimental setting, the results on real-world dataset, and a detailed analysis on the experimental results. The conclusion and future work are presented in Section V.

II. Related Work

In traditional models, travel time is treated as a numerical value or distribution derived from existing historical trajectory data. Statistic models like Historical Average (HA) and Auto-Regressive Integrated Moving Average (ARIMA) [7] take the travel time forecasting as a time sequence forecasting task. They build temporal relation on the historical travel time data to predict its future value. However, while this method is useful in the short-term forecasting scenario, it is vulnerable when the model needs to predict in a non-stationary and complex environment [7]. Also, these methods suffer from a performance decline when the data is sparse [8]. There are also works attempting to estimate the distribution of travel time for a given route. For examples, [9] adopts Gaussian Markov Random Field to forecast travel time distribution of arterial networks. Spatio-Temporal-based Route Recovery System (STRS) [10] derives the distribution by trajectory regression, it comprises a forecasting module of travel time distribution and an inference module based on its spatial transition relation.

Recently, data-driven deep learning models are widely adopted in this field [2]. Deep learning models pay more attention to use neural network structure to model spatial and temporal dependencies. For spatial dependency modeling, Previous work mostly capture the correlation with a CNN structure. For example, [11] leverages the convolutional model based on grid structure data to capture spatial dependency between close regions for traffic condition forecasting. CNN's framework simplifies the data preprocessing and makes the training easier to implement as it relaxes the constraints of spatial relations, but this comes at a cost on fine-grained information loss. For instance, this method can not distinguish the turnings if the related locations are mapped into some regions. To tackle this problem, [2] proposed $Geo\text{-}Conv$ layer which adopts CNN layers for mapping longitude and latitude of GPS points in a route. However, this method still has limitations. As $Geo\text{-}Conv$ layer performs convolution operation directly on geographic information, this method relies highly on the amount of data. When the data is sparse or there are significant missing data, the performance is notably undermined. Modeling of temporal dependency also occupies an important position for these deep models trained by historical data. [12] constructs temporal dependency from temporal closeness and periodicity of traffic status. However, this empirical temporal dependency modeling method also brings noise when the dataset is not suitable for its prior assumption about temporal dependency. Since that, [13] uses Recurrent Neural Network-based models to build a more general model by using Long Short-Term Memory (LSTM) cells [14] and Gated Recurrent Unit (GRU) cells [15] to memorize the temporal status. In a word, all these temporal dependencies are based on the assumption that the spatial dependency is temporally invariant in the same time slot (e.g., 6:00 pm-7:00 pm every Monday). However, This assumption is not in line with the practice because many random variables such as the weather, accidents, etc. also affect the traffic status.

Considering this fact, [11] proposes DeepGTT, which build temporal dependency based on the real-time traffic condition. DeepGTT adopts the idea of Bayesian inference to forecast the travel time distribution of a given route as a whole. Though it effectively forecasts time distribution through Bayesian modeling, there are still two major problems in DeepGTT. The first problem relates to its spatial extraction methods. As we discussed before, a pipeline of grid structure process does not meet the real road traffic structure. As an alternative, Graph Neural Network (GNN) [16], which is derived from graph spectral theory [17], is a favorable deep learning structure that extracts features in the way that better suits road network topology. The second problem is its training objective. Though DeepGTT can learn distribution through a deep Variational Inference learning method. The training of Variational Inference process focuses more on better point-wise prediction rather than approximating the ground truth distribution [18]. Besides, this is also one reason why the images generated by Variational Autoencoder (VAE) are often fuzzy compared with the images generated by GAN [19]. It does not fully understand the distribution but pays more attention to the point-wise numerical approximation [20].

III. Methodology

In this section, we first introduce the definition of the travel time estimation problem. Then we introduce the road representation layer and sptial dependency modeling, as GCGTTE inferences travel time based on these two mechanisms. We show how to obtain the route time estimation at the end of this section. The learning algorithm of GCGTTE is presented in Algorithm 1.

A. Definition

a) Definition 1. Road Network: Road network is a directed graph $G = (V, E)$ comprise of interwined roads. $V = \{v_1, v_2, \ldots, v_N\}$ is the set of nodes (locations in the transportation network where traffic information like speed can be detected) in road network. $E = \{e_1, e_2, \ldots, e_M\}$ is the set of roads in traffic graph network. For its sparsity, we use adjacent matrix as the data structure to store the road network. Let the adjacent matrix be $M \in \mathbb{R}^{N \times N}$. If there is a road between node v_i and v_j, $M(i, j) = 1$. Otherwise $M(i, j) = 0$. For road e_i in G, its n types categorical information is represented as $\{c_0, c_1, \cdots, c_n\}$

b) Definition 2. Trajectory: A raw GPS trajectory $traj$ is a set of sequence sample points $\mathrm{p}_i = \langle \mathrm{lat}_i, \mathrm{lon}_i, T_i \rangle$ from the underlying route of a moving vehicles, where lat_i, lon_i, t_i represents the latitude, longitude, and timestamp of the i-th point, respectively.

c) Definition 3. Route: $R_j = \{e_i, \ldots, e_n\}$ is a set of roads, which represents an ordered path sequence of a route.

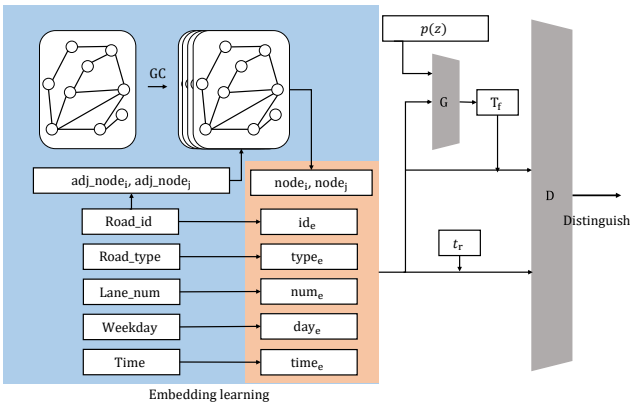

Fig. 1. Overview of GCGTTE.

Algorithm 1: GCGTTE training procedure.

Input: Train dataset: $(s_i, t_i)_{i=1}^{M}$, Embedding Layers: Embed$(\cdot;\theta_{em})$, Generator: G$(\cdot;\theta_G)$, Discriminator: D$(\cdot;\theta_D)$, GCN Network: GC$(\cdot;\theta_{GC})$, Road Network Adjacent Matrix: M, Real-time Traffic Information: X, Node Attributes C.

Output: Parameter set: $\{\theta_G, \theta_{em}, \theta_{GC}\}$

Initialize θ_G, θ_{em}, θ_D, θ_{GC} with random weights

repeat

  Select time period t, real-time traffic info X_t

  $X_{GC} = \sigma\left(\widehat{M}\,\mathrm{Relu}\left(\widehat{M}X_tW_0\right)W_1\right)$

  $\mathbf{em}$ = Embed(C)

  s = concat(X_{GC},$\mathbf{em}$)

  for $\{(s_i, t_i)\,|t_i = t\}$ **do**

    Choose s_i's adjacent node n_1 and n_2

    for *k-steps* **do**

      Sample z from noise prior $p(z)$

      Generate t_f = G$((s_i, z)\,;\theta_G)$

      Obtain $\{s_i, t_f\}$

      Update θ_D via equation 3

    Sample z from noise prior $p(z)$

    Generate t_f = G$((s_i, z)\,;\theta_G)$

    Obtain s_i, t_f

    Update θ_G via equation 3

  Update θ_{em}, θ_{GC}

until *model converges*;

B. Road Representation Layer

Road representation layer is used to extract the corresponding road's latent representation, which is used as one main factor to inference travel time. In the road representation layer, we compute the embedding of roads' categorical features first. Then, we perform the graph convolutional operation with real-time traffic information of nodes through road network G. Finally, by concatenating the graph learning results as the road's external features with its categorical features embedding, we obtain the final representation s_i for road r_i.

a) Embedding Learning: For categorical information features, we use embedding layers to embed different features for two purposes: (1) Convert the categorical features from a static text representation into a low-dimensional categorical numerical representation; (2) Embed the categorical features that share similar semantic representation with closer distance in the embedding space. Embedding learning layers transform categorical features to an embedding value by performing a matrix multiplication with weight matrix $W \in \mathbb{R}^{N\times E}$. Comparing with the one-hot encoding method, embedded features $\mathbf{em}$ is more computationally efficient as they effectively reduce the size of input dimension.

b) Spatial Dependency Modeling: The spatial dependency modeling of road networks is a fundamental problem in traffic forecasting-related tasks. We adopt Graph Convolutional Network (GCN) to build the spatial dependency for its superior ability on extracting graph structured data's spatial features. Inspired by work [4], we adopt a two-layer GCN structure to aggregate the spatial information, which can be formulated as follows:

$$X_{GC} = \sigma\left(\widehat{M}\,\mathrm{Relu}\left(\widehat{M}XW_0\right)W_1\right) \tag{1}$$

where $\widehat{M}$ is the symmetric normalized laplacian of $\widehat{M} = M + I$. W_0 and W_1 are the weight matrices of the first and second GCN layer, respectively. $\sigma(\cdot)$ represents the *sigmoid* activation function.

C. Distribution Approximation Algorithm

a) Theoretical Analysis: A key issue in travel time forecasting is to approximate its distribution with a proper approach. We divide the distribution methods into two types: Variational Inference (VI)-based approaches and generative adversarial training-based approaches. An example of a VI-based deep learning model is DeepGTT [11], it approximates the distribution by putting a prior distribution assumption on the data and use it as regularization. Its distribution are learned as a Gaussian with mean value μ and standard deviation value σ. The inference process of a VI-based model can be expressed as follows:

$$P\left(t_i \mid s_i\right) = \mathcal{N}\left(\mu(s_i), \sigma(s_i)\right) \tag{2}$$

$\mathcal{N}(\cdot)$ represents a Gaussian distribution, $\mu(s_i)$ and $\sigma(s_i)$ are mean value and standard deviation value learned from s_i with two fully-connected layers, respectively. In our experiment, we follow the same structure in [11] to learn $\mu(s_i)$ and $\sigma(s_i)$. We can see that VI-based models achieve distribution approximation by fitting limited data to a Gaussian distribution or a mixed Gaussian distribution in an explicit way. Though it avoids using computationally expensive methods like Monte Carlo sampling to approximate the distribution directly, this approach typically leads to two problems: (1) the utilization of VI leads to information loss inevitably and introduces redundant information in the learning process. (2) VI-based models do not fully understand the distribution as its optimization pays more attention to the point-wise numerical approximation [20]. Alternatively, adversarial training methods give a different insight into distribution

approximation by learning the distribution implicitly. Let p_z denote the PDF of a random variable and p_{data} denote the travel time distribution in dataset. Then the objective function of GAN is given by

$$\min_G \max_D V(D,G) = \mathbb{E}_{s\sim p_{\text{data}}(s)}[\log D(s)] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z})))] \tag{3}$$

By optimizing the above objective function, GAN-based model can learns data distribution implicitly [19]. Though it puts a prior knowledge on the random variable, it avoids making assumptions on the data directly, which makes it more flexible. We conduct a more in-depth comparison of these two models in Section IV.

Furthermore, another design of GCGTTE is to make the model produce the corresponding travel time estimation with a specified input $\mathbf{y}$. For instance, if we want to obtain the travel time distribution of e_i on 12:00 am, Monday, we can feed corresponding time embedding and road representation vectors to the models and get the specific results conditioned on $\mathbf{y}$. The objective function of this conditional form can be summarized as

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}\mid \boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1-D(G(\boldsymbol{z}\mid \boldsymbol{y})))] \tag{4}$$

D. Route time estimation

For the travel time forecasting of a whole route, we sum up the travel time forecasting results for roads in set $\mathbf{R}_j$, which can be formulated as $T_{\mathbf{R}_j} = \sum^{e_i\in\mathbf{R}_j} t_{e_i}$, where $t_{e_i} = G(s_{e_i})$

IV. Experiments

In this section, we conduct experiments on a real-world dataset and compare GCGTTE with other baseline models. We first compare the performance of models on travel time estimation and discuss the results. Then we further compare GCGTTE's distribution approximation ability with DeepGTT, which is a VI-based travel time distribution estimation method. We show a visualization result to depict the distribution approximation comparison betwen GCGTTE and DeepGTT.

A. Experimental Settings

a) Experimental Environment: All experiments are performed on a Linux Server (CPU: Intel(R) Xeon(R) CPU E5-2620v4, GPU: GeForce RTX2080Ti, System: Ubuntu 18.04). All experiments are conducted with Pytorch 1.3.0.

b) Hyperparameter Settings: We use a two-layer linear neural network to learn the embedding features. The associated categorical features of roads used in our experiment are RoadID, Day, Hour, RoadType, LaneNumber, and Road Length. We embed RoadID to $\mathbb{R}^{\not 4}$ Day to $\mathbb{R}^{\not 4}$, Hour to $\mathbb{R}^{\not 4}$, RoadType to $\mathbb{R}^{\not 4}$. For Road Length, we process it with a normalization approach. For the hidden size of GCN blocks, W_0 and W_1 are set to 16 and 32, respectively. From above settings, the final unified representation for road r_i at time t_j is a vector with size $(49,)$. We choose the Adam optimizer to optimize network parameters. The learning rate μ during training is 10^{-3}. We set the batch size as 4096 as it is shown that a batch with large size benefits GAN's training, which usually suffers from model collapse issues. To avoid overfitting, GCGTTE uses an early stopping strategy when the validation loss no longer increases for three consecutive epochs. The generator of GAN is designed as a two-layer linear network, the weight matrix of each linear layer is $[49, 64]$, $[64, 1]$. The discriminator of GAN is a two-layer linear network with sizes $[49, 64]$, $[64, 1]$. We use a Gaussian distribution with mean value 0 and variance 1 for p_z. Followed the dataset split ratio adopted in [4], the proportions for the training set, validation set, and test set are 60%, 10% and 30%. For the comparison on MSE and RMSE, we sample distribution and running the point estimation models 50 times. The average value of results are used for final comparison.

c) Dataset Description: We conduct our experiment on the Didi Chuxing Gaia dataset[1], this dataset collects GPS data of Didi car-hailing trajectories in the northeastern region of Chengdu city during November 1, 2016 and November31, 2016. We obtain the road network meta information from OpenStreetMap API[2], which contains a total of 2584 points and 8432 edges.

d) Data Preprocessing: Since there is only raw GPS data in the dataset, we need to process it before training the model. The procedure of data preprocessing is as follows:

- Firstly, we calculate the motion information from its geographic location: latitude and longitude

$$\Delta t_j = t_{j+1} - t_j \tag{5}$$

$$\mathrm{RD}_j = \text{Vincenty}\left(\text{lat}_j, \text{lon}_j, \text{lat}_{j+1}, \text{lon}_{j+1}\right) \tag{6}$$

$$v_j = \mathrm{RD}_j / \Delta t_j \tag{7}$$

  where Δt_j, d_j, v_j represent the elapsed time, relative distance and velocity of GPS point p_j, respectively. Vincenty$(\cdot)$ represents the Vincenty formula [21], which is used to calculate the distance between two points on the surface of a spheroid.
- Secondly, as we acquire the graph network of the road from Open Street Map, all GPS trajectories are mapped into the graph network to get their road-mapped route representation by the map-matching algorithm proposed in [22].

e) Evaluation Metrics: We use Root Mean Square Error (RMSE) and Mean Average Error (MAE) to measure models' average prediction error. They are formulated as

$$\mathrm{MAE} = \frac{1}{m}\sum_{i=1}^{m} \mid \left(T_i - \widehat{T}_i\right) \tag{8}$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(T_i - \widehat{T}_i\right)^2} \tag{9}$$

where Y_i and $\widehat{Y}_i$ are the true value and predicted value, respectively.

[1]Data available at `https://outreach.didichuxing.com/appEn-vue/dataList` upon application.

[2]`https://wiki.openstreetmap.org/wiki/API`

TABLE I

METRICS COMPARISON ON TRAVEL TIME ESTIMATION

	RMSE(sec)	MAE(sec)
TTIGAN	620.96	516.33
DeepST	450.38	348.53
DeepTTE	353.34	258.10
NeiTTE	275.63	200.65
DeepGTT	281.38	205.75
GCGTTE_{C}	271.55	197.80
GCGTTE	**203.01**	**155.77**

f) Baselines: We compare GCGTTE with five other baselines: DeepST [12], DeepTTE [2], Nei-TTE [13], TTI-GAN [23] and DeepGTT [11].

- **DeepST** adopts CNN to build its spatial dependency on the grid structure data. Besides, it builds temporal dependency from constructing the temporal closeness part, seasonal trend part, and the period part from an empirical perspective.
- **DeepTTE** models spatial-temporal dependency from a CNN-based network and an LSTM-based network. It use an attribute embedding layer to learn the embedding of given roads' categorical features. Moreover, it predicts travel time from spatial-temporal dependency and attributes embedding.
- **TTIGAN** adopts Info-GAN [24] to impute the travel times for the missing data. It uses Skip-Gram model [25] and a heuristic clustering approach as the features for model training.
- **DeepGTT** uses a Variational Inference method to model the travel time of a given route. It first learns the variance σ and means μ of posterior velocity distribution based on the road attributes and environmental attributes. Then the travel time distribution is obtained by a gaussian inverse transformation.
- **Nei-TTE** further extends the work of DeepGTT by introducing the knowledge of neighboring regions with an attention mechanism.

B. Case Study

a) Performance on Estimation: Table 1 presents the experimental results. For models using real-time traffic information, GCGTTE performs significantly better than DeepGTT. However, this is primarily due to the fact that GCGTTE uses a more fine-grained spatial dependency modeling method by adopting graph convolutional operation to extract spatial features. Thus, for a fair comparison, we design a variant of GCGTTE called GCGTT_{C}, which uses CNN for spatial information extraction. We can see that the result of GCGTTE_{C} is close to DeepGTT, which indicates that both these two models have effectively learned the data during training. As GCGTTE_{C} performs slightly better than DeepGTT on the test set, we attribute this to the fact that the Variational Inference used by DeepGTT is essentially a point-wise training process and it leads to an overfitting problem easily.

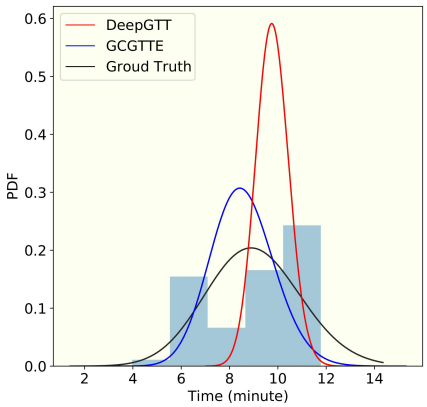

Fig. 2. PDF of DeepGTT, GCGTTE, and groud truth for estimating a route's travel time distribution on Tuesday 8:00 am.

TABLE II

TIME COST STUDY

	GCGTTE	DeepGTT	DeepTTE	DeepST
Training Time	**6.5h**	11h	16h	12h
Prediction Time	**15.3s**	23.6s	34.1s	27.0s

b) Grid Structure v.s. Graph Structure: The effectiveness of graph structure data used in our model can be derived from the comparison between GCGTTE and GCGTTE_{C}. It can be seen that if we replace the GCN layer in road representation layer with CNN layer, the RMSE increases from 271.55 to 203.01, while the MAE increases from 197.80 to 155.77. This result indicates the postive effectiveness of our idea to apply GNN to build spatial dependency on traffic network. As GNN constructs the learnable network follows the graph structure of road network, it has more advantages in capturing traffic spatial dependency.

c) Distribution Approximation: For distribution approximation, we compare the VI-based model DeepGTT and the GAN-based model GCGTTE. Figure 2 shows a comparison of the distribution generated by GCGTTE and DeepGTT for a route on Tuesday 8:00 a.m. We can see that DeepGTT's result presents a narrow Gaussian distribution, which does not match the actual distribution well. It shows that the learning process of DeepGTT focuses more on point estimation than the distribution approximation. Thus, it might degenerate into a point estimation model gradually. On the contrary, the distribution curve fitted by GCGTTE shows a more similar contour. This is because GCGTTE does not directly train the training set but tries to generate the distribution by itself.

d) Time Cost Study: Before we compare time cost, we first conduct a time complexity comparison between CNN and GCN, as they correspond to the spatial dependency modeling approaches of different models. The time complexity of CNN is

$$\text{Time} \sim O\left(M^2 \cdot K^2 \cdot C_{\text{in}} \cdot C_{\text{out}}\right) \tag{10}$$

where M is the spatial size of the output feature map, K is the kernel size, C_{in} and C_{out} are the input channels and

output channels, respectively. On the other hand, since GCN is a matrix multiplication process, its time complexity can be expressed as

$$\text{Time} \ \sim O\left(N^2 \cdot L\right) \tag{11}$$

where N is the graph network's spatial size, L is the size of features. From (10) and (11), we can see GCN-based methods are less time consuming. As CNN-based methods usually use a deep neural network structure, the last two terms are typically large, rendering a longer training time. On the other hand, a traffic network usually has only a few attributes, making its term L very small. The training time cost and online prediction time cost in Table II verifies our hypothesis. It shows that GCGTTE has a 40.9% decrease in time cost performance compared with DeepGTT on model training time and 35.1% decrease on online prediction time.

V. Conclusion

This work proposes a travel time distribution forecasting model GCGTTE. Rather than a point estimation model, GCGTTE approximates the underlying travel time distribution with GAN-based model. Besides, it establishes the spatial dependency through Graph Neural Network, a deep learning framework extracts spatial features more in line with the actual topological structure of the traffic network. Case studies demonstrate that GCGTTE outperforms models built on grid-structured data significantly. Besides, with a GAN-based structure, the results on a real-world taxi dataset show that it has a better performance on travel time distribution approximation compared with the state-of-the-art Variational Inference-based model, namely DeepGTT. Furthermore, we provide a theoretical analysis on model's time complexity, and experimentally demonstrate the efficiency of GCGTTE.

In future work, we will extend our existing work in the following directions: 1) apply probabilistic deep learning techniques to capture the prediction's uncertainty, 2) further improve and generalize GCGTTE by conducting reasonable knowledge transfer methods between different nodes or different road networks, and 3) design a better and efficient spatial-temporal dependency modeling method for GCGTTE.

References

[1] C. Buchanan, *Traffic in Towns: A study of the long term problems of traffic in urban areas*. Routledge, 2015.

[2] D. Wang, J. Zhang, W. Cao, J. Li, and Y. Zheng, "When will you arrive? estimating travel time based on deep neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, New Orleans, Louisiana, USA., 2018.

[3] W. Wei, X. Jia, Y. Liu, and X. Yu, "Travel time forecasting with combination of spatial-temporal and time shifting correlation in cnn-lstm neural network," in *Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data*, 2018, pp. 297–311.

[4] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, and H. Li, "T-gcn: A temporal graph convolutional network for traffic prediction," *IEEE Transactions on Intelligent Transportation Systems*, vol. 21, no. 9, pp. 3848–3858, 2019.

[5] M. Fosgerau and D. Fukuda, "Valuing travel time variability: Characteristics of the travel time distribution on an urban road," *Transportation Research Part C: Emerging Technologies*, vol. 24, pp. 83–101, 2012.

[6] P. Domingos, "Bayesian averaging of classifiers and the overfitting problem," in *Proceedings of the International Conference on Machine Learning*, San Francisco, CA, USA, 2000, p. 223–230.

[7] S. Oh, Y.-J. Byon, K. Jang, and H. Yeo, "Short-term travel-time prediction on highway: a review of the data-driven approach," *Transport Reviews*, vol. 35, no. 1, pp. 4–32, 2015.

[8] M. Shepperd and M. Cartwright, "Predicting with sparse data," *IEEE Transactions on Software Engineering*, vol. 27, no. 11, pp. 987–998, 2001.

[9] T. Hunter, A. Hofleitner, J. Reilly, W. Krichene, J. Thai, A. Kouvelas, P. Abbeel, and A. Bayen, "Arriving on time: estimating travel time distributions on large-scale road networks," *arXiv preprint arXiv:1302.6617*, 2013.

[10] H. Wu, J. Mao, W. Sun, B. Zheng, H. Zhang, Z. Chen, and W. Wang, "Probabilistic robust route recovery with spatio-temporal dynamics," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 1915–1924.

[11] X. Li, G. Cong, A. Sun, and Y. Cheng, "Learning travel time distributions with deep generative model," in *The World Wide Web Conference*, San Francisco, CA, USA, 2019, p. 1017–1027.

[12] J. Zhang, Y. Zheng, D. Qi, R. Li, and X. Yi, "Dnn-based prediction model for spatio-temporal data," in *Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, New York, NY, USA, 2016.

[13] J. Qiu, L. Du, D. Zhang, S. Su, and Z. Tian, "Nei-tte: intelligent traffic time estimation based on fine-grained time derivation of road segments for smart city," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 4, pp. 2659–2666, 2019.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," in *Proceedings of the 27th International Conference on Neural Information Workshop on Deep Learning, December 2014*, 2014.

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

[17] D. K. Hammond, P. Vandergheynst, and R. Gribonval, "Wavelets on graphs via spectral graph theory," *Applied and Computational Harmonic Analysis*, vol. 30, no. 2, pp. 129–150, 2011.

[18] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *International Conference on Learning Representations*, Banff, AB, Canada, April 2014.

[19] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proceedings of the 27th International Conference on Neural Information Processing Systems*, Montreal, Canada, 2014, p. 2672–2680.

[20] S. Zhao, J. Song, and S. Ermon, "Towards deeper understanding of variational autoencoding models," *arXiv preprint arXiv:1702.08658*, 2017.

[21] T. Vincenty, "Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations," *Survey review*, vol. 23, no. 176, pp. 88–93, 1975.

[22] C. Yang and G. Gidofalvi, "Fast map matching, an algorithm integrating hidden markov model with precomputation," *International Journal of Geographical Information Science*, vol. 32, no. 3, pp. 547–570, 2018.

[23] K. Zhang, Z. He, L. Zheng, L. Zhao, and L. Wu, "A generative adversarial network for travel times imputation using trajectory data," *Computer-Aided Civil and Infrastructure Engineering*, 2020.

[24] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Barcelona, Spain, 2016, p. 2180–2188.

[25] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proceedings of the 26th International Conference on Neural Information Processing Systems*, vol. 2, Lake Tahoe, Nevada, 2013, p. 3111–3119.