Research Article

Travel Time Estimation by Learning Driving Habits and Traffic Conditions

Ling Yang,1,2 Shouxu Jiang,1 Fusheng Zhang,3 and Ming Zhao2

1Faculty of Computing, Harbin Institute of Technology, Harbin 150006, Heilongjiang, China
2School of Informatics, Harbin Guangsha College, Harbin 150025, Heilongjiang, China
3Key Laboratory of Elevator Intelligent Safety in Jiangsu Province, Changshu Institute of Technology, Changshu 215500, Jiangsu, China

Correspondence should be addressed to Shouxu Jiang; jsx@hit.edu.cn

Received 1 March 2022; Revised 6 April 2022; Accepted 20 April 2022; Published 28 June 2022

Academic Editor: Muhammad Arif

Copyright © 2022 Ling Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Travel time estimation (TTE) is widely applied for ride dispatching, ride-hailing, and route navigation. There are many factors affecting the travel time of a driver on a given trajectory, including the distance, road type, driving habits, traffic congestion, etc. Existing works fail to model the complex relationships of these factors for TTE. To fill this gap, in this paper, we first analyze how these factors work together in determining the travel time. In particular, the travel time depends on the distance and driving speed on each road segment of the trajectory, where the driving speed depends on the driving habits and the environment, including the static factors like the road type (highway or byway) and speed limit and the dynamic factor like the time of the day and congestion. Among these factors, driving habits and traffic conditions (e.g., jam) are the most difficult ones to model. Second, we propose to learn the driving habits of each driver via meta-learning and estimate the conditions based on the current and historical traffic conditions (via recurrent neural networks) of this road and its connected road segments (via graph convolutional neural network). The experimental results on two real taxi trajectory datasets show that our approach outperforms three state-of-the-art methods significantly.

1. Introduction

Travel time estimation (TTE) in urban cities plays a key role in route planning [1], vehicle dispatching [2], and ride-hailing [3] applications, such as Uber, Lyft, and Didi. For example, results in [4] show that inaccurate travel time estimation leads to 28.4% car-booking cancellation rate. In this paper, we study the problem of estimating the travel time on a trajectory for a (taxi) driver at a specific time. For example, when we use ride-hailing Apps, we input the origin and destination; the Apps find a driver for us and then show the trajectory, the driver, and the estimated travel time.

Many approaches have been proposed in the literature [5–9], [10] on TTE. Factors like traffic flow, weather condition, road type, etc. have been exploited in estimating the travel time. However, the existing works fail to analyze the complex relationships among them and combine them together for TTE. For example, [6] considers the traffic flow, weather, and incidents but ignores the road type and driving habits. Reference [11] integrates the weather, day of the week, time of the day, and driver information but ignores the traffic conditions. There are two challenges in combining all factors: (1) analyzing the relationship of these factors in affecting the travel time, e.g., how does road type affect the estimation of travel time; (2) modelling each factor. Some factors can be modelled explicitly like weather conditions (“rainy” or “snowy”); some factors are implicit like the driving behavioral characteristics. Inadequate understanding of these factors may cause inaccuracy in the estimation.

In this paper, we first analyze the relationships of the factors that may affect TTE. Generally, the travel time on one road segment depends on the length and the driving speed. The length is easy to get, whereas the speed depends on many factors, which can be put under two categories, namely, the
static factors and dynamic factors. The static factors include the road type, e.g., highway or byway, road width, speed limit, and in-degree and out-degree, etc. The dynamic factors include the driver, weather, incidents, traffic flow, time of the day, and day of the week, etc. Some factors are known at the start of the trip like the information of all road segments of the trajectory, the driver, the date, and time. Some factors are impossible to get like the incidents; some factors can be estimated like traffic flow (conditions). Among all these factors, the traffic conditions and driver habits are the two most difficult ones to model.

Second, to learn the traffic conditions (flow) of a road segment on the trajectory, on the one hand, we exploit the traffic conditions at the start of the trip and the traffic conditions from previous days using a recurrent neural network to estimate the conditions when the driver reaches the road segment. This is to model the temporal correlation with the assumption that the current and historical conditions will affect the future conditions. On the other hand, we consider the traffic conditions of the nearby road segments through graph convolutional neural networks. This is to model the spatial correlation with the assumption that the traffic conditions (e.g., jam) are influenced by the nearby road segments. The two aspects are illustrated in Figure 1, where the vertical axis is the time dimension and each plane shows the road network at one time point.

Third, we propose to learn the driving habits of each driver via meta-learning. An embedding vector is learned for each driver, which is considered as the metadata of the driver. During the travel, this embedding vector is adjusted according to the dynamic environment to reflect the driver’s driving habits. The intuition is that the driver may behave differently under different situations as shown in Figure 2. For example, after a heavy traffic jam, the driver who typically drives slowly may drive very fast.

With the proposed key techniques, we learn a speed vector on a road segment by combining the traffic conditions and the driver habits. The speed vector and the road static factors are then fed into a regression network to estimate the travel time. By learning over all road segments of a trajectory together, the whole model is a multitask learning model. The contributions of this paper include the following:

(i) We propose to learn an embedding for every driver, which is adjusted dynamically via meta-learning to represent the driving habits of the driver at real time.

(ii) We propose to learn the traffic conditions by exploiting the spatial correlation with nearby road segments via graph convolutional networks and the temporal correlation with historical traffic conditions via recurrent neural networks.

(iii) We conduct extensive experiments to evaluate the performance of the proposed techniques. Results on two real datasets demonstrate that our solution can achieve better performance than the state-of-the-art methods, including DeepTTE [11], TEMP [12], and traditional statistical methods like ARIMA.

The rest of the paper is structured as follows. The state-of-the-art solutions for TTE and related deep learning algorithms are reviewed in Section 2. The preliminaries, as well as the precise problem definition, are introduced in Section 3. The employed methodology and the computational framework are depicted in Section 4 and evaluated in Section 5. Finally, Section 6 concludes this work and discusses future directions.

2. Related Work

Travel Time Estimation is a basic functional component of the Intelligent Transportation System (ITS). In recent years, the prediction accuracy of traveling time estimation has been improved by various methods. The traveling time estimation can be effectively extracted from the traffic data features by using deep representation learning. The accuracy of the traveling time estimation of the assigned driver depends on his/her driving speed. The traffic and individual driving behavior affect the personal driving speed on the road.

Some researchers solved the travel time estimation problem along the path inference [13, 14], path selection [15], and path component [16]. However, they assume the path not given in the TTE problem. It is different from the problem we are solving where the path is known. Papers [1, 17] model for the traveling time of road segment distribution learning, and [18–20] focus on the distribution or probability of path travel time or the path selection, which are different input from our problem. The input of travel time estimation was the origin and destination two points without the given path [12, 14, 21, 22], which did not work much in navigation and ride-sharing many specific fields. The travel time estimation problem for assigning drivers is a very important function in specific applications but has been neglected in the existing works.

It is difficult to model the transportation system through a physical model in an explicit form since it is a dynamic system. For example, it is difficult to model the whole traffic condition that is too complex, since conditions such as weather, traffic lights, vehicle breakdowns, and traffic accidents can affect the traffic. Therefore, there is no guarantee that the estimation of each time period in each road segment has a higher estimation accuracy. Papers [12, 16] attempt to accurately estimate travel time by carefully modelling the traffic condition of the objective world. Paper [12] leverages the trips from a similar origin and destination locations to estimate the travel time. Paper [16] proposes the model P-TTE to estimate the missing element in three-dimension tensor with each road segment during each time slot by each driver. However, they used historical data which makes hard to monitor real-time traffic. Papers [19, 23] compute the travel time histogram to a large extent depending on the road segment types with the real-time speed information or path [18] and turn costs information; however, the independence between different road segments is assumed. Paper [17] considers the space-time dependency between different road segments to develop a deep generation model, DeepGTT, for learning the travel time distribution of arbitrary paths using the CNN method to obtain real-time
Figure 1: The spatial-temporal correlation on each road segment and interdependence between road segments.

Figure 2: Continued.

- **Assigned Driver (ID:100322191)**
- **Taxi Driver (ID:100324511)**
- **Average Speed**
road conditions. However, this method does not consider other complex factors that can affect the traffic. Papers [6, 24] consider complex factors and characteristics, such as spatial-temporal dependency, traffic flow, weather, and events and use multiple source datasets for designing data-driven regression methods to understand and predict travel times. The whole path is estimated directly in [11, 24]. Paper [11] proposes a multitask learning model called DeepTTE which can effectively overcome the complex factors affecting the travel time estimation of an entire path with only GPS points. Paper [24] presents an auxiliary supervision model called DEEPTDIGITAL, which can extract multiple features that effectively capture different dynamics for estimating the travel time of a path, such as short-term and long-term traffic features for estimating the travel time. Travel time estimation in [11, 13] uses GPS data without road network or road segment information. Paper [13] investigates ST-NN, a deep neural network, to jointly estimate the travel time with raw GPS data. However, the geographical features [25, 26] or images [14] do not consider the dependency among road segments. Papers [21, 22] provide only origin location and destination location. Travel time estimation based on the origin-destination (OD) [21] path using a multitask learning framework aims to produce a meaningful representation of properties in the road network structure and the spatial-temporal prior knowledge from the traces. Paper [22] formulates the travel time estimation as a regression problem and developed wide-deep-recurrent model with origin-destination, OD pairs, which included popular OD. However, travel time estimation from these works is the average travel time by all the drivers, instead of individual driver. Papers [11, 12, 21, 24] formulate the travel time estimation as a multivariate time-series prediction problem; however, they fail to consider general traffic conditions and personalized driving behaviors.

Papers [27–29] consider driving habits that focus on a few main styles, and the classification is not obvious. They embedded the features of drivers ID into a fixed-size vector as the label of drivers, which is very coarse-grained and static as a label. However, such limited information can only provide weak distinction for the assigned drivers or even lead to error results in real-world applications. Paper [28] proposes an end-to-end STDR deep learning network with road type to estimate travel time based on historical trajectories and external factors without the driver information. Paper [29] proposes Customized Travel Time Estimation (CTTE) with topology representation, speed statistics, and query distribution focused on the special personal trails such as aggressive driving which need to be learned from features such as speeding, sudden braking, and frequent lane changing of drivers that are not considered. To improve the quality of service, it is therefore vital to provide a personality assessment based on the personality of assigned drivers, not only based on a label.

Clearly, existing works on travel time estimation aim to estimate the average travel time of a path. We take a step further by estimating travel time based on the assigned driver on a given path. To this end, we propose a hybrid neural network with graph diffusion convolutions and the gated mechanisms to estimate travel time of a given path for an assigned driver.

3. Preliminary

There are three main data sources: road network data, trajectory data, and auxiliary attribution data. The road network data includes road segment information, e.g., road segment ID number, direction, road class, length, and the topology of the road network. The trajectory data includes driver ID number, road segment ID number, time, date,
speed, and the loading state. The auxiliary attribution data is the weather. Obviously, the hidden data (e.g., spatial-temporal correlation, traffic, driving habits) cannot be captured by any observable data sets directly, which need to be learned.

3.1. Definitions

Definition 1 (Directed Graph for Road Network): We represent a road network as a directed graph \( G = (V, E) \), where \( V \) is the vertex set that represents road segments, and \( |V| = N \) is the number of road segments in the road network. \( E \) is the edge set that represents the connectivity between road segments. \( A \) is a \( N \times N \) adjacency matrix that captures how the directed edges are connected. In particular, \( A_{i,j} = 1 \) if it is possible to travel from edge \( v_i \) to edge \( v_j \); otherwise, \( A_{i,j} = 0 \). \( A_{i,j} \neq A_{j,i} \) because it is a directed graph. \( F \in R^{N \times N \times T} \) is the feature tensor of the graph \( G \), where \( T \) is the number of time steps in different hours of the day (peak vs. nonpeak traffic) and \( M \) is the number of features on the road segment. For example, \( F \in R^{N \times N \times M} \) represents the feature matrix of the road segment at time \( t \).

Road network is described by the real road in map. However, the map is consecutive data set. We put the trajectory data into the map which turns to discrete data set. Feature matrix \( S \in R^{N \times D} \) corresponds to the features of each road segment based on the time (in the fixed time), where \( D \) is the number of features on the road segment after graph convolutional operation. For example, \( S_i \) represents that the \( i \)-th row corresponds to the features vector of road segment \( v_i \) at time \( t \). The key features we consider are shown in Figure 3 from the observed datasets. They include road segment ID, road segment length, direction, road segment class, in-degree, and out-degree, etc. Driving habits and traffic are hidden features, which need to be derived from the observed data, e.g., road segment data, trajectory data, and auxiliary data. Among the abovementioned features, the characteristics of road segments, e.g., road length and road level, are time-invariant while the characteristics of trajectory, e.g., speed and loading state, are time-variant. The travel time of the given path for the assigned driver is estimated by the traffic and the speed of the assigned driver.

Definition 2 (Path and Trajectory): A path is the sequence of road segments \( p = < v_1, v_2, ..., v_i, ..., v_j > \), with \( |p| = n \) and \( 1 \leq i \leq j \leq n \). Given the path, we define the set of trajectories as \( TR \) and a trajectory of a driver \( u \) as \( TR(u) \), which consists of a sequence of points, where \( TR(u) \subset TR \). Each trajectory is a 6-tuple: \( (u, v_i, x'_t(u), t, d, ls) \), where \( u \) is the driver ID, \( x'_t(u) \) is the speed of driver \( u \) at the time \( t \) in the road segment \( v_i \), \( t \) is the timestamp, \( d \) is the date, and \( ls \) is the loading state, where \( ls = 1 \) means there are passengers in the taxi, otherwise 0.

Definition 3 (Driving): We consider three trajectory features as the driving habit: (1) Driver ID number \( u \); (2) speed in the road segment \( v_i \) at the time \( t \), \( x'_t(u) \); (3) loading state \( ls \). Driving habits \( DR \) is generated from these features. The driving habit of the assigned driver \( u \) on the road segment \( v_i \) in the time \( t \) is denoted as \( DR(u) \).

Definition 4 (Traffic): Intuitively, the average speed of all drivers could be regarded as an indicator of the traffic condition \( C \in R^{N \times T} \). It consists of real-time CR and periodicity traffic condition \( CP \).

3.2. Problem Statement. Problem Definition: In our problem, we perform a travel time query for the assigned driver through the given path with three inputs: \( p \) is a given path, \( t_s \) is the departure time, and \( u \) is the driver. Our model, named as DRTTE (Deep dRiver TTE), returns the travel time \( t_i \) for road segment \( v_i \) and \( t_m \) for the entire path. We model the travel time of the entire path and each road segment simultaneously using a multitask learning framework.

Figure 3 shows the multisource observed and derived features for our problem. The travel time estimation of the given path for the assigned driver \( t_m \) depends on the travel time of each road segment \( t_i \). The travel time of the road segment depends on the information of the road segment \( ST_i \) and the speed for the assigned driver \( SP_i(u) \). The speed of the assigned driver depends on the driving habits \( DR(u) \) and the traffic on the road segment \( CR_i \). The traffic depends on the real-time traffic \( CR_p \) and the periodicity traffic \( CP_p \). The real-time traffic depends on the spatial-temporal correlation \( ST_i \) and the auxiliary data, e.g., weather \( w \). The spatial-temporal correlation depends on the spatial correlations \( S_i \) and temporal correlation. The details for the notations and definition are showed in Table 1.

4. Methodology

Travel time estimation can deal with deep learning approaches for the complex and dynamic system and effectively extracting features of traffic data. Our method utilizes trajectory data, road network data, and weather data as input. Figure 4 presents the details of the framework, which is comprised of four major components, namely, Road Segment Component, Path Component, Driving Habits Component, and Travel Time Estimation Component.

During the training phase, given a road network \( G \) and a historical trajectory \( TR \), we learn (1) how to predict the public traffic \( C \) in the road network and (2) how to learn the speed \( SP \) for the assigned driver \( u \) via inertial data. During the test phase, given a driver ID \( u \), a given path \( p \), and a departure time \( t_s \), our goal is to estimate the travel times \( t_i \) and \( t_m \) for the assigned driver. The details for the input and output of these components are showed in Table 2.

Road Segment Component: We first embed the public average speed on the road segment with time into vectors using a convolutional layer with filters so that the spatial characteristics can be captured. After that, the output matrix vectors of conventional operation are used as the input of GRU. The real-time traffic consists of spatial-temporal information and the auxiliary
Figure 4: The architecture of DRTTE: GRU: gated recurrent unit; LSTM: long short-term memory.

Figure 3: The graphical model for the observed data and the hidden data on one road segment.

Table 1: Notations and definition.

| Symbol | Meaning |
|--------|---------|
| $G(E, V)$ | A directed graph of road network in map |
| $A \in \mathbb{R}^{N \times N}$ | The adjacency matrix of graph $G$ |
| $F \in \mathbb{R}^{N \times M}$ | The adjacency matrix of graph $G$ |
| $F^t \in \mathbb{R}^{N \times M}$ | The feature matrix of a graph $G$ at time $t$ |
| $S \in \mathbb{R}^{N \times D}$ | The feature tensor of a graph $G$ after convolution operation at time $t$ |
| $S_t \in \mathbb{R}^{N \times D}$ | The feature matrix of graph $G$ after convolution operation at time $G$ |
| $N$ | The number of road segment, $N = |E|$ |
| $M$ | The number of features in each road segment |
| $D$ | The number of features in each road segment after convolutional operation |
| $k$ | The number of time steps in road segment |
| $p$ | Given path |
| $n$ | The number of road segments for the given path |
| $u$ | The driver $u$ |
| $ts$ | Departure time. Generally, the current time is $ts - 1$ |
| $w$ | The weather |
| $C$ | The public traffic |
| $CR$ | The real-time traffic |
| $CP$ | The periodicity traffic |
| $DR(u)$ | The driving habit of the assigned driver $u$ |
| $t_i$ | The travel time of road segment $v_i$ by driver $u$ |
| $ten$ | The travel time of the entire path $p$ by driver $u$ |
information, e.g., weather. The final traffic contains real-time traffic and weekly periodicity traffic.

Path Component: We embed the traffic from the road segment component into the path component using the GAT with time to get the traffic in the next time step along the path. The attention coefficients from the GAT show the level of dependency between road segments.

Driving Habits Component: We explore the speed of the assigned driver with traffic and driving habits. The driving habits are learned with meta-learning.

Travel Time Estimation Component: The multitask learning structure is composed of the public traffic, the speed of the assigned driver, the travel time estimation of each road segment, and the travel time of the entire path.

4.1. Road Segment Component. We build the road segment component to predict the traffic \( C^i_t \) in the road segment \( v_i \) at time \( t \). The traffic \( C^i_t \) depends on the real-time traffic \( CR^i_t \) and history periodicity traffic \( CP^i_t \). The real-time traffic \( C^i_t \) depends on the spatial-temporal features \( ST^i_t \) and auxiliary information \( w \) weather. We build the GCN-GRU model to capture the spatial-temporal features \( ST^i_t \) in the road segment \( v_i \) with the graph convolution from the spatial view and GRU from the temporal view on the road segment \( v_i \) at departure time \( t \). The history periodicity traffic shared the same model structure with the real-time traffic, which is trained offline with history weekly trajectory data.

The intuition for using GCN is that road network data is one kind of nonregular grid, which can be obtained by traditional Convolutionsal Neural Network (CNN). Graph convolution network (GCN) is commonly used to extract the spatial features on static graphs and it is suitable for the non-Euclidean structure. We adopt the graph convolution operation to obtain the spatial characteristics of the road segment given its structural information. A GCN unit takes the feature matrix \( S \) and the adjacency matrix \( A \) as the input and conducts the spectral graph convolution operation. The final output of each time step is \( S' \), which can be defined as follows:

\[
S' = \text{GCN}(F', A),
\]

\[
= \sigma(\hat{A}F'W).
\]  

\[\hat{A} = D^{-1/2} \hat{A} D^{-1/2}\]

\[\hat{A} = A + I_N\]

\(\hat{A}\) denotes the graph convolution filter, and \(\hat{A}\) is a matrix.

With self-connection structure, \(\hat{D}\) is a degree matrix, \(\hat{D} = \sum \hat{A}_{ij}\), \(W\) is the weight matrix, and \(\sigma(\cdot)\) represents the activation function. \(S_i\) is \(i\)-th row of \(S\) and denotes the learned spatial vector for the road segment \(v_i\).

The temporal information of the road segment \(v_i\) is another key problem in spatial-temporal correlation. Recurrent Neural Network (RNN) is most widely used for processing sequence data. However, the traditional RNN has limitations for long-term prediction because of the gradient vanishing and gradient explosion. The above problem has been addressed by long short-term memory (LSTM) and Gated Recurrent Unit (GRU) model, which are designed according to the basic principle that the gated mechanisms are all used to memorize as much long-term information as possible. Compared with GRU, LSTM takes a longer time to train because of its complex structure. Consequently, we obtain the temporal information using the GRU model from spatial feature \(S\) of each time step. As shown in Figure 4, GRU in DRTTE works as follows. It obtains the spatial-temporal correlation at time \(t\) in the road segment \(v_i\) by taking the spatial-temporal correlation hidden status \(ST^i_{t-1}\) at time \(t - 1\) and the current spatial information \(S^i_t\) as inputs. The mathematical formulation is shown in

\[
\begin{align*}
\underline{u}' &= \sigma(W_u \cdot [S^i_t, ST^i_{t-1}]), \\
\underline{r}' &= \sigma(W_r \cdot [S^i_t, ST^i_{t-1}]), \\
\underline{c}' &= \tanh(W_c \cdot [S^i_t, \underline{r}' \cdot ST^i_{t-1}]), \\
ST^i_t &= \underline{u}' \cdot \underline{c}' + (1 - \underline{u}') \cdot ST^i_{t-1}.
\end{align*}
\]

The traffic consists of real-time traffic and the history periodicity traffic. In Figure 5, the \(x\)-axis describes the time in one day of the week, and the \(y\)-axis describes the speed. Usually, the traffic speed on Monday has a certain similar traffic speed on Mondays in history, as shown in Figure 5, but may be greatly different from those on weekends. For example, we can find a similar speed at 7–9 am on the 6th Jan and on the 14th Jan. The two days have the same tendency. Thus, according to this observation, the history weekly periodicity component is designed to capture the weekly periodic features in traffic data. The generation of real-time traffic and history periodicity traffic shares the same network structure and each of them consists of several spatial-temporal blocks with the GCN and GRU. The real-time traffic and the periodicity traffic all have the spatial-temporal features fusion with the weather. For real-time traffic \(CR \in RN^{\times T}\), we concatenate spatial-temporal features \(ST\) and the weather \(w\) as \(CR = [ST, w]\). The operation of \([,]\) is the concatenate operator. The weekly traffic has the weekly period \(CP\) in history periodicity data for all drivers.

In the next step, we will discuss how to fuse real-time traffic and history periodicity traffic. The fusion formula is \(C^i_t = WR \odot CR^i_t + WP^i \odot CP^i_t\), where \(\odot\) is the Hadamard product. \(WR\) and \(WP\) are parameters which need to be learned. They reflect the relative importance between real-time traffic and history periodicity traffic. The intuition
comes from the observation that the relative importance of real-time traffic and history periodicity traffic differs from road segments to road segments. Consequently, when these two kinds of traffic information are fused, their weights need to be learned separately for each road segment.

The overall procedure of capturing spatial-temporal correlations of traffic is described in Algorithm 1.

### Algorithm 1: The procedure of capturing spatial-temporal correlations.

**Input:** A, S, t, w, CP, and time step k  
**Output:** C  

1. Initialize S; C; ST randomly;  
2. for i = 0; i ≤ |p|; i ≠ N do  
3. for t = ts − k; t ≤ k; t + do  
4. STt+ = GRU (S(t));  
5. CRt+ = STt+ ∘ w;  
6. C(t) = WRt+ ∘ CRt+ + Wp+ ∘ CP(t);  
7. end  
8. end  
9. return ST, C.

Figure 5: The temporal shifting of periodicity.

4.2. Path Component. The public traffic of each road segment is guided not only by their current traffic conditions but also by their neighboring road segments’ traffic conditions. This component tries to find the dependency between each road segment. In our case, we are more concerned with the dependencies between the road segments involved in the given path. We describe how we obtain traffic condition representations of the target road segment ($v_j$) from its neighboring road segments. The grey rectangle includes the history of periodicity data, which is the time-invariant feature derived from the history data. Then, we fuse this feature with the real-time traffic into multigraph attention networks to find out the traffic along the given path. In the path component, we combine the traffic of the target road segment and its neighbors into a new feature vector to represent the traffic of the target road segment in the next time period. The combined representation is a mixture of the target road segment’s traffic and its neighbors’ traffic for the next time step.

Graph Attention Network (GAT) is adapted by us to combine information about the target road segment’s neighbors. The key idea is to weight the features of the neighbors using an attention mechanism. The weight is the level of influence of neighbors on the target road segment. The details of our GAT are explained as follows.
For each road segment, we build a subgraph that consists of its neighbors. For target road segment $v_j$ with $|N(j)|$ neighboring road segments, the graph has $|N(j)| + 1$ nodes. The traffic of the $v_j$ road segment $C^t_j$ is used as its features. With the node features defined as above, we then combine features of the target road segment and its neighbors. This procedure is formalized as inference in a GAT [30] that is for node representation with semisupervised learning. In our problem, in terms of dependence, for each road segment, we only consider its one-hop neighbors. However, all neighbors are equal to the target road segment. Instead, we design a dynamic GAT to model the dependence level. The fixed symmetric normalized Laplacian is widely used as a propagation operator in existing GAT. In order to distinguish the dependency level of each neighboring road segment, we propose to use an attention mechanism to guide the dependence level.

Figure 6 shows the process of the road segment located in the next step and in the next time $ts + t$. Road segment $v_j$ is the target road segment, whose traffic relies on the traffic at the time $ts$ in previous road segment $e_l$ and in the neighbors of the target road segment. The process is illustrated in Figure 6. The dependency of the target road segment can be represented by the $a^{ts}_i$ using GAT. Finally, the new traffic on the target road segment $v_j$ next time $ts + t$ is combined by the activation function $\sigma$, which is shown in Algorithm 2.

In equation (3), function $f(\cdot)$ applies the LeakyReLU nonlinearity (with negative input slope $\alpha = 0.2$). Fully expanded out, by the attention mechanism the coefficients computed may then be expressed as

$$a^{ts}_{i,j} = \frac{\exp(f(C^{ts}_i, C^{ts}_j))}{\sum_{l \in N(j) \cup j} \exp(f(C^{ts}_l, C^{ts}_j))}$$  \hspace{1cm} (3)

$C^{ts}_j$ is the representation traffic of road segment $v_j$ at time $ts$. Intuitively, $a^{ts}_{i,j}$ is the level of dependency or weight of road segment $v_i$ on road segment $v_j$. We also include a self-connection edge to preserve a road segment’s revealed dependency as follows:

$$\tilde{C}^{ts}_j = \sum_{k \in N(j) \cup j} a^{ts}_{i,k} C^{ts}_k.$$  \hspace{1cm} (4)

$a^{ts}_{i,j}$ provide the weights to combine the features; $\tilde{C}^{ts}_j$ is a combination of road segments’ neighbors’ dependency at time $ts$, followed by a transformation defined in

$$C^{ts+ti}_j = \sigma(C^{ts}_i + \tilde{C}^{ts}_j),$$  \hspace{1cm} (5)

where $\sigma(\cdot)$ is active function ReLU and $C^{ts}_j$ is the traffic in the road segment $i$ which is the previous road segment for the target road segment $v_j$ at time $ts$.

4.3. Driving Habit Component. The travel time estimation for the assigned driver is impacted directly by the public traffic which has been described in the above section. In this section, we learn the driving habit for the assigned driver with history trajectory data which are prepared offline.

4.4. Travel Time Estimation Component. The travel time for the assigned driver of the road segment is related to the speed and the length of the road segment. We have got the speed feature $SP$ and the road segment $S$ from the above components. Consequently, we choose the MLP model to generate a hidden variable $h_i$ for road segment $v_i$. We then design a multitask learning framework to estimate the travel time for the given path using $h_i$. In our model, we design the multitask learning component for two main tasks. During the training phases, the tasks are accurately estimating the travel time of each road segment and the entire path. During the test phase, the tasks are travel time estimation for each road segment and the entire path.

Now, we have got the feature sequence $h_1, h_2, \ldots, h_n$ for the given path. Each $h_i$ corresponds to the feature vector of the travel time on each road segment $v_1, v_2, \ldots, v_n$. Each $h_i$ can be mapped to a scalar $t_i$ through fully connected layers, where $t_i$ is the predicted travel time of the $i$-th road segment $v_i$. 

![Figure 6: The graphical model of the road segment using the attention mechanism.](image)
In terms of the travel time for the whole path, we need to think of a way to combine the travel time for each road segment. Mean pooling or max pooling is one choice, i.e., \( h_{\text{mean}} = \frac{1}{n} \sum_{i=1}^{n} h_i \), which is effective. However, this method ignores the relative importance of each segment in estimating the travel time for the entire path. The attention mechanism is thus adopted by us instead of the mean pooling. It is essentially a weighted sum of sequence \( \{h_i\} \), where the weights are parameters learned by the model. Formally, we have that

\[
\begin{align*}
    h_{\text{att}} &= \sum_{i=1}^{n} \alpha_i \cdot h_i, \\
    \alpha_i &= \frac{e^{z_i}}{\sum_{j} e^{z_j}}, \\
    z_i &= \langle \sigma_{\text{att}}(\text{att}), h_i \rangle.
\end{align*}
\]

\( \alpha_i \) is the weight for the \( i \)-th road segment in the path, and the summation of all \( \alpha_i \) equals number 1. To learn the weight parameter \( \alpha \), we consider the traffic information of each road segment, as well as the speed for the assigned driver. \( \alpha \) is the attention correlation coefficient, which means the importance of neighbors for the road segment \( v_i \).

Finally, \( h_{\text{att}} \) passes the several fully connect layers that are connected with residual connections. In our model, we use \( \sigma_{\text{att}} \) to denote the \( i \)-th residual fully connected layer. At last, we use a single neuron to obtain the estimation of the entire path, which we denote as \( t_{\text{est}} \). The algorithm of the travel time estimation of the road segment and the entire path is shown as Algorithm 4.

\section*{5. Experiments}

In this section, we evaluate the effectiveness of our proposed DRTTE method in terms of the overall performance and effectiveness of different components on large-scale real-world taxi datasets from two cities. We compare our DRTTE model with the baseline methods including ARIMA, TEMP [12], and DeepTTE [11].

\subsection*{5.1. Experiment Settings}

\subsubsection*{5.1.1. Datasets.} We evaluate our model on two real taxi trajectory datasets, namely, Harbin and Chengdu. The two datasets have the same format, consisting of trajectory data, road network, and auxiliary data, e.g., weather. For the convenience of the calculation, continuous road networks are cut into discrete road segments. With two-dimensional GPS data consisting of longitude, dimension is transformed to the one-dimensional road segment data consisting of road segment ID by Map Matching algorithm. Table 3 shows the details for two taxi datasets. The Chengdu dataset is a public dataset generated by 14,864 taxis in August 2014 in Chengdu, China. The Harbin dataset is generated by 16,852 taxis in Harbin, China, from 2nd Jan 2017 to 26th Jan 2017. The total length of the road segments is 4,650.55 km and the number of the road segments is 28,964.

\subsubsection*{5.1.2. Baseline Methods.} The baseline methods include ARIMA, SimpleTTE/TEMP [12], and DeepTTE [11], which are explained as follows.
ARIMA: Autoregressive Integrated Moving Average, which is a statistical method for time-series problems. ARIMA depicts a suite of different standard temporal attributions.

TEMP [12]: TEMP is a simple TTE method that makes use of the travel time of neighboring trips with the same original destination OD pairs in the large amount of historical trajectory data to make the estimation. TEMP is a representative approach for calculating the travel time of the entire path.

DeepTTE [11]: DeepTTE is a typical method of deep learning for travel time estimation. DeepTTE captures the spatial features with the geo-convolution operation and captures the temporal dependencies with stacking LSTM layers. The travel time estimation of the road segment and the travel time estimation of the entire path are determined by multitasking learning.

5.2. Evaluation Metric. The evaluation metrics we adopt include mean absolute percentage error (MAPE), root-mean-squared error (RMSE), and mean absolute error (MAE). Mean absolute percentage error (MAPE) compares the value of estimation to the percentage of the ground-truth value, while root-mean-squared error (RMSE) and mean absolute error (MAE) are the gap values between estimation value and true value. Equation (9) gives the mathematical formula of the three metrics. In equation (9), $g_{ti}$ denotes the ground truth of the $i$-th road segment, $t_i$ denotes the estimation value of the $i$-th road segment from DRTTE, and $n$ denotes the number of road segments in the given path.

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{g_{ti} - t_i}{t_i} \right| \times \frac{1}{n},$$

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (g_{ti} - t_i)^2},$$

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^{n} |g_{ti} - t_i|.$$

5.3. Comparisons with Baselines. In this section, we evaluate the effectiveness of our proposed DRTTE in terms of MAPE, RMSE, and MAE. Table 4 shows the comparison results between baseline methods and our DRTTE method.

From Table 4, we can observe that ARIMA achieves the worst results. The reason is that it relies on spatial-temporal historical data to predict future travel time value without considering the spatial features, e.g., road segment class, road length, road network topology etc., and other related external features, e.g., weather. This result demonstrates that the traditional time-series prediction method cannot capture the complex spatial-temporal relationship. TEMP method displays the medium performance between the static method, ARIMA, and deep learning method, DeepTTE. The reason is that TEMP is an approximate method; it is more suitable for the highway or expressway in urban without traffic changing. It cannot cope with the problem of our complicated traffic conditions. The results of TEMEP and DeepTTE are both better than ARIMA, which reveals that the deep learning methods can deal with the large-scale
complex data better than non-deep learning models. For DeepTTE, since it adopts the convolutional operations to deal with discrete locations in order to capture the spatial characteristic, it achieves better results than TEMP. Finally, our DRTTE model significantly outperforms other methods on two datasets, respectively. The reason is threefold. Firstly, our model exploits graph convolution operations to make use of spatial information. Secondly, the graph attention networks with temporal operations are designed to find out the dependency among the road segments with road properties. Thirdly, the effect of driving habits is taken into account for the travel time estimation for the assigned driver. These innovations help preserve the spatial-temporal characteristics of the traffic and the driving habits for the assigned driver.

5.4. Effectiveness of Different Components. There are five significant components in our DRTTE model, namely, (1) “LSTM,” (2) “GCN and GRU,” (3) “GAT,” (4) “DR,” (5) “ATTENTION,” as shown in Figure 4.

In this section, we evaluate the effectiveness of these components by adding them one by one and observing the performance gain. The four models we evaluate are described as follows: The first model is an LSTM for multitask learning without the information road segments and the information of road network characters. The second model is an LSTM combined with GCN and GRU; this model is able to discover and utilize the spatial-temporal information of the road segments. The third model is an LSTM combined with GCN, GRU, and GAT. This model takes the dependency between road segments into consideration. The fourth model is an LSTM combined with GCN, GRU, GAT, and driving habits (DR). This model puts driving habits of the assigned driver into the model. The last model is an LSTM combined with GCN, GRU, GAT, DR, and attention mechanism in the multitask layer. This model is our DRTTE.

From Table 5, we have the following observations. Firstly, merely LSTM exhibits the lowest performance. Secondly, the model “LSTM + GCN + GRU” achieves similar results to the DeepTTE model shown in Table 4. This is because they employ similar spatial-temporal features. However, DeepTTE learns the spatial relationship by two consecutive locations with a fixed time gap. The convolution operation will have errors when the objects are in the same place within the sample gap. It tends to get convolution operation results that are an error when the sample data is nearly constant. To address this issue, we design GAT with time to capture the spatial dependence. Thirdly, DeepTTE is worse than the model “LSTM + GCN + GRU + GAT.” The reason is that GCN and GRU capture the spatial-temporal correlation on each road segment and GAT captures the dynamic dependency between road segments. On the contrary, DeepTTE can only find the dependency between locations of a fixed timestamp, instead of temporal dependency between locations. Fourthly, the model “LSTM + GCN + GRU + GAT + DR” is better than all the abovementioned models. The reason is that personalized driving habits have been trained and employed for every driver offline. Lastly, the attention mechanism in our DRTTE only improves the results a bit. The reason is that the error has already been reduced a lot in the previous components. The performance of our DRTTE is the best when all the components are considered. The DRTTE can interpret the generation of travel time for the assigned driver, and it could reveal the dependency among relevant variables and explore training data in a more efficient way.

5.5. Impacts of the Kernel Size. In this section, we evaluate the impacts of kernel size of the graph convolutional operation. From Figure 8, we can clearly observe that the MAPE, MRSE, and MAE have the same trend, and the best results are obtained when the kernel size is neither the biggest nor the smallest. When the kernel size is less than 4, it cannot capture the entire spatial correlation; when the kernel size is more than 4, it captures more unnecessary information that damages the true correlation between road segments.

5.6. Impacts of History Periodicity Data. In this section, we evaluate the impacts of the history of periodicity data. For the history periodicity, we feed a one-day history and three-week history, respectively, into the model and observe the results. Figure 9 reports the results on DeepTTE and our DRTTE. In general, DRTTE performs better than DeepTTE, even when only one-day history is fed. The reason is, in our model, the historical periodicity data is used by the traffic component and driver driving is learned by habit component, both of which are offline and the data is prepared. This effectively reduces the sensitivity of our model DRTTE to the data. Therefore, in the face of sparse training data, it can be effectively dealt with. Our model DRTTE has great advantages compared with other models.

Another observation is that when the amount of train data is too small or insufficient, MAPE of deep learning model (DeepTTE) (55.32%) is bad as the ARIMA (35.49%). Deep learning (DeepTTE) is a model that requires a large amount of data to train lots of parameters and weights in order to improve accuracy. However, the traditional
statistical method ARIMA is a model that predicts the data without training, which will give a result based on the data. The size of the training data impacts the performance of the model. We study the change of MAE with the different size of training data points from 3,000 to 90,000 of the assigned driver in one day and in the three weeks. DRTTE is not good enough for the baseline ARMIR (MAPE: 35.49%) with few data, but the performance will be greatly improved with training with a large amount of data.

5.7. The Travel Times and Distances Patterns. To study the distance performance of the model DRTTE for travel time estimation, we randomly pick 400 given paths including 9,870 road segments from the validation datasets of the assigned drivers. Then we calculate the travel time and travel distances of these 400 given paths. Figure 10 presents the MAPE and MAE results over the length of the path. In this part, we focus on the impact of travel distance on the performance of DeepTTE and our DRTTE. We divide the given path into subpaths into units of two/three kilometers' lengths (in 2 KM/3 km step), \([0, 2), [2, 5), [5, 7), [7, 10]) in the end. Because the relevant literature gives some conclusions, the distance generally does not exceed 10 kilometers by taxis. Since both of our real data are from taxis, the upper limit for selecting a given path is set to 10 kilometers. In Figure 10, we compare the two models DeepTTE and DRTTE performances for the given path with different

|                     | Chengdu |                | Harbin |                |
|---------------------|---------|----------------|--------|----------------|
|                     | MAPE (%)| RMSE (s)       | MAE (s)    | MAPE (%)    | RMSE (s)     | MAE (s)     |
| ARIMA               | 35.49   | 444.42         | 357.22  | 32.32        | 413.62       | 310.47      |
| SimpleTTE/TEMP      | 26.45   | 324.18         | 213     | 22.75        | 314.08       | 193.61      |
| DeepTTE             | 19.37   | 289.51         | 191.26  | 17.61        | 267.04       | 164.23      |
| Ours DRTTE          | 13.91   | 252.32         | 155.71  | 11.64        | 242.7        | 136.29      |

| Kernel Size | MAPE (%) | RMSE (s) | MAE (s) | MAPE (%) | RMSE (s) | MAE (s) |
|-------------|----------|----------|---------|----------|----------|---------|
| 2           | 200      | 190.1    | 180     | 160.78   | 155      | 145.67  |
| 3           | 170.35   | 170.89   | 160.32  | 150.38   | 145.67   | 136.29  |
| 4           | 145.67   | 145.67   | 136.29  | 136.29   | 136.29   | 136.29  |
| 5           | 136.29   | 136.29   | 136.29  | 136.29   | 136.29   | 136.29  |
| 6           | 136.29   | 136.29   | 136.29  | 136.29   | 136.29   | 136.29  |

![Figure 8: Impacts of kernel size.](image)
Figure 9: Impacts of history periodicity data.
lengths with two city datasets. When the travel distance of the given path goes faster, DeepTTE and our model DRTTE methods face the accuracy of declining problems. The error rates increase as a consequence with the distance of the given path, which means the uncertainty of the traffic conditions increases, causing the performance degradation of the model. Estimation of travel time is useful with shorter distances which shows that the travel time estimation of each road segment is valuable. Through the two data sets, we find that the performance of the model is similar to the given path length increases. Even in the longest distance, the MAPE (27.66% in DRTTE and 27.22% in DeepTTE) value is optimistic and the MAE (276.3 seconds in DRTTE and 309.16 seconds in DeepTTE) is meaningful, which means the 5 minutes is the error time in the travel of 10 kilometers.

DeepTTE shows a better MAPE (17.61% in Harbin dataset) when the distance of the given path is around 2 km. However, it fails to handle the longer given paths (with length greater than 5 km). The effect of DeepTTE after 7 km (MAPE 32.88% and 36.52%) is not good enough with the length of the path. The effect of our model DRTTE after 7 km (MAPE 23.9% and 27.66%) is similar to the effect of DeepTTE around 5 km (MAPE 27.12%). In contrast, our model is also less sensitive to the distance of the given path than the DeepTTE.

Figure 10 also plots the MAEs values of DeepTTE and our model DRTTE about travel distance. Not surprisingly, the value of MAE increases with the travel distance which means there are many error values for the estimation value. Longer trips typically have much more travel time. It is
worth to show that the performance difference between DeepTTE and DRTTE is similar to the travel distance. This result suggests that DRTTE is more trustful for short trip estimation and DRTTE is available but does not have much advantage for long trips.

5.8. The Effects of Hyperparameters. Figure 11 shows the curve of MAPE and MAE results with epoch numbers “20, 40, 60, 80, 100,” respectively. We observe that the epoch is effective to estimate the travel time accuracy, reducing the MAPE from (Chengdu 50.75, Harbin 42.23) to (Chengdu 13.91, Harbin 11.64) and reducing the MAE from (Chengdu 320.75, Harbin 280.23) to (Chengdu 155.71, Harbin 136.29).

6. Conclusions

In this paper, we propose a novel framework, namely, DRTTE, which takes traffic conditions and driving habits into consideration in estimating the travel time for assigned drivers. The DRTTE framework is designed to find the spatial-temporal dependent traffic with GCN, GRU, and GAT, driving habits with meta-learning, and subsequently to estimate the travel time with multitask learning. We conduct experiments on two real taxi trajectory datasets to understand the dependency of spatial-temporal information for traffic and driving habits for the assigned driver and to confirm the superiority of DRTTE [33].

Data Availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by Major Natural Science Research Projects of Colleges and Universities in Jiangsu Province (no. 20KJA460011): Research on Elevator Safety Situation Cloud Awareness System Based on Multisource Sensor Data Fusion.

References

[1] J. Hu, C. Guo, B. Yang, and C. S. Jensen, “Stochastic weight completion for road networks using graph convolutional networks,” in Proceedings of the 35th IEEE International Conference on Data Engineering, pp. 1274–1285, Macao, China, April 2019.
[2] B. Zhao, P. Xu, Y. Shi, Y. Tong, Z. Zhou, and Y. Zeng, “Preference-aware task assignment in on-demand taxi dispatching: an online stable matching approach,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 2245–2252, Honolulu, HI, USA, February 2019.
[3] X. Geng, Y. Li, L. Wang et al., “Spatiotemporal Multi-Graph Convolution Network for Ride-Hailing Demand Forecasting,” in Proceedings of the AAAI Conference on Artificial Intelligence, AAAI, Honolulu, HI, USA, February 2019.
[4] D. Shah, A. Kumaran, R. Sen, and P. Kumaraguru, “Travel time estimation accuracy in developing regions: an empirical case study with uber data in Delhi-ncr,” in Proceedings of the Companion Proceedings of the 2019 World Wide Web Conference, Association for Computing Machinery, New York, NY, USA, May 2019.
[5] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, “Attention based spatial-temporal graph convolutional networks for traffic flow forecasting,” in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 922–929, AAAI, Honolulu, HI, USA, January 2019.
[6] S. Yang and S. Qian, “Understanding and predicting travel time with spatio-temporal features of network traffic flow, weather and incidents,” 2019, https://arxiv.org/abs/1901.06766.
[7] B. Yu, H. Yin, and Z. Zhu, “Spatio-temporal Graph Convolutional Neural Network: A Deep Learning Framework for Traffic Forecasting,” CoRR, 2017, https://arxiv.org/abs/1709.04875.
[8] H. Yao, X. Tang, H. Wei, G. Zheng, Y. Yu, and Z. Li, “Modeling Spatiotemporal Dynamics for Traffic Prediction,” CoRR, 2018, https://arxiv.org/abs/1803.01254.
[9] Y. Wu and H. Tan, “Short-term Traffic Flow Forecasting with Spatiotemporal Correlation in a Hybrid Deep Learning Framework,” CoRR, 2016, https://arxiv.org/abs/1612.01022.
[10] Y. Ling, Z. F. Jia, S. X. Jiang, R. X. Min, and Z. F. Sheng, “Urban Night Bus Routes Planning with Taxi traces,” in Proceedings of the 2017 12th International Conference on Computer Science and Education (ICCSE), IEEE, Houston, TX, USA, August 2017.
[11] W. C. J. L. Dong Wang, J. Zhang, and Y. Zheng, “When will you arrive? estimating travel time based on deep neural networks,” in Proceedings of the 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, December 2018.
[12] H. Wang, Z. Li, Y. Kuo, and D. Kifer, “A simple baseline for travel time estimation using large scale trip data,” CoRR, 2015, https://arxiv.org/abs/1512.08580.
[13] I. Jindal, Tony, Qin, X. Chen, M. Nokleby, and J. Ye, “A Unified Neural Network Approach for Estimating Travel Time and Distance for a Taxi Trip,” arXiv e-prints, 2017, https://arxiv.org/abs/1710.04350.
[14] W. Lan, Y. Xu, and B. Zhao, “Travel time estimation without road networks: an urban morphological layout representation approach,” in Proceedings of the 28th International Joint Conference on Artificial Intelligence, pp. 1772–1778, AAAI Press, Macao, China, July 2019.
[15] J. Dai, B. Yang, C. Guo, C. S. Jensen, and J. Hu, “Path cost distribution estimation using trajectory data,” Proceedings of the VLDB Endowment, vol. 10, no. 3, pp. 85–96, 2016.
[16] Y. Wang, Y. Zheng, and Y. Xue, “Travel time estimation of a path using sparse trajectories,” in Proceedings of the 20th SIGKDD Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, New York, NY, USA, August 2014.
[17] X. Li, G. Cong, A. Sun, and Y. Cheng, “Learning travel time distributions with deep generative model,” in Proceedings of the The World Wide Web Conference, pp. 1017–1027, Association for Computing Machinery, New York, NY, USA, May 2019.
[18] R. Waury, C. S. Jensen, and K. Torp, “Adaptive travel-time estimation: a case for custom predicate selection,” in Proceedings of the The 19th IEEE International Conference on Mobile Data Management (MDM), pp. 96–105, IEEE, Aalborg, Denmark, June 2018.
[19] R. Waury, J. Hu, B. Yang, and C. S. Jensen, "Assessing the Accuracy Benefits of On-The-Fly Trajectory Selection in fine-grained Travel-Time Estimation," in *Proceedings of the 2017 18th IEEE International Conference on Mobile Data Management (MDM)*, pp. 240–245, IEEE, Daejeon, Republic of Korea, June 2017.

[20] E. Jenelius and H. N. Koutsopoulos, "Urban network travel time prediction based on a probabilistic principal component analysis model of probe data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 436–445, 2018.

[21] Y. Li, K. Fu, Z. Wang, C. Shahabi, J. Ye, and Y. Liu, "Multi-task representation learning for travel time estimation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 2018.

[22] Z. Wang, K. Fu, and J. Ye, "Learning to estimate the travel time," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, July 2018.

[23] A. Narayanan, N. Mitrovic, M. T. Asif, J. Dauwels, and P. Jaillet, "Travel Time Estimation Using Speed Predictions," in *Proceedings of the 2015 IEEE 18th International Conference on Intelligent Transportation Systems*, IEEE, Gran Canaria, Spain, September 2015.

[24] H. Zhang, H. Wu, W. Sun, and B. Zheng, "Deeptravel: A Neural Network Based Travel Time Estimation Model with Auxiliary Supervision," CoRR, 2018, https://arxiv.org/abs/1802.02147.

[25] C. de Fabritiis, R. Ragona, and G. Valentii, "Traffic estimation and prediction based on real time floating car data," in *Proceedings of the 11th International IEEE Conference on Intelligent Transportation Systems*, October 2008.

[26] R. Tani, T. Owada, and K. Uchida, "Path travel time estimating method by incomplete traffic data," *International Journal of Intelligent Transportation Systems Research*, vol. 18, no. 1, pp. 43–52, 2018.

[27] D. J. Weiss, A. Nelson, H. S. Gibson et al., "A global map of travel time to cities to assess inequalities in accessibility in 2015," *Nature*, vol. 553, no. 7688, pp. 333–336, 2018.

[28] J. Xu, Y. Zhang, L. Chao, and C. Xing, "Strd: a deep learning method for travel time estimation, database systems for advanced applications," in *Proceedings of the Database Systems for Advanced Applications*, pp. 156–172, Springer, New York, NY, USA, April 2019.

[29] R. Gao, X. Guo, F. Sun et al., "Aggressive driving saves more time? multi-task learning for customized travel time estimation," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pp. 1689–1696, AAAI Press, Macao, China, July 2019.

[30] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proceedings of the International Conference on Learning Representations (ICLR)*, Vancouver, Canada, May 2017.

[31] J. Chen, X. Qiu, P. Liu, and X. Huang, "Meta Multi-Task Learning for Sequence Modeling," CoRR, 2018, https://arxiv.org/abs/1802.08969.

[32] D. Gupta, T. Chakraborty, and S. Chakrabarti, "Girnet: Interleaved Multitask Recurrent State Sequence Models," AAAI, 2019, https://arxiv.org/abs/1811.11456.

[33] L. Yang, S. Jiang, and F. Zhang, "Multitask learning with graph neural network for travel time estimation," *Computational Intelligence and Neuroscience*, vol. 2022, no. 1, pp. 2022–9, 2022.