CTTE: Customized Travel Time Estimation via Mobile Crowdsensing

Ruipeng Gao, Fuyong Sun, Weiwei Xing, *Member, IEEE*, Dan Tao, Jun Fang, and Hua Chai

***Abstract*— Estimating the origin-destination travel time is a fundamental problem in many location-based services for vehicles, e.g., ride-hailing, vehicle dispatching, and route planning. Recent work has made significant progress to accuracy, but they largely rely on GPS trajectories which are too coarse to model many personalized driving behaviors, e.g., differentiating novice and veteran drivers. In this paper, we propose Customized Travel Time Estimation (CTTE) that fuses GPS trajectories, smartphone inertial data, and road network within a deep recurrent neural network. It constructs a road link traffic database with topology representation, speed statistics, and query distribution. It also calibrates inertial readings, estimates the arbitrary phone's pose in car, and detects multiple aggressive driving events (e.g., bump judders, sharp turns, sharp slopes, frequent lane shifts, overspeeds, and sudden brakes). Finally, we demonstrate our solution on two typical transportation problems, i.e., predicting traffic speed at holistic level and estimating customized travel time at personal level, within a multi-task learning structure. Experiments on two large-scale real-world traffic datasets from DiDi platform show our effectiveness compared with the state-of-the-art.**

***Index Terms*— Travel time estimation, traffic speed prediction, aggressive driving behaviors, mobile crowdsensing.**

I. Introduction

THANKS to the explosion of sharing economy, we can easily hail a ride at anywhere and anytime in most urban cities. Such ride-hailing platforms, e.g., Uber, Lyft, and DiDi, benefit our everyday travel and ensure efficient use of vehicles. However, the riding experience differs a lot among drivers, and sometimes they may even adopt aggressive driving behaviors to arrive earlier. To get rid of such dangerous events,[1] we are curious about how much time they can save for their specific driving behaviors.

Origin-Destination Travel Time Estimation (ODTTE) is pivotal to many location-based services, including ride-hailing, vehicle dispatching, and route planning. It estimates the travel time of a specific driver through a given route at the departure time. Central to this issue relies on the balancing art between large-scale crowdsourced traffic information and a variety of personalized driving behaviors. A solution must consider both general traffic speed at public level and specific driving patterns at personal level.

Recently, a series of efforts have been undertaken to address this problem. The route-based solutions [3]–[5] estimate the driving time on each road segment and intersection, then summarize them as the origin-destination travel time. However, precisely modeling of dynamic transportation systems is difficult, especially via sparse and low-quality crowdsourced traffic data. The data-driven solutions [6]–[11] are mainly based on machine learning techniques and attract much attention in both research and industry, formulating a multivariate time series prediction problem via the spatial-temporal traffic data. However, a solution must consider both holistic traffic conditions and driver's specific riding patterns.

In this paper, we propose Customized Travel Time Estimation (*CTTE*), a novel multi-source heterogeneous data fusion approach that can predict both holistic traffic speed on each road link and travel time for each individual, via one multi-task learning model. Such a data fusion approach entails a series of non-trival challenges. First, how to eliminate sensor noises and extract the most effective features from multi-source heterogeneous traffic data. Second, how to identify different aggressive driving events for crowdsourced drivers, despite noisy inertial data from smartphones with arbitrary poses in car. Finally, how to elegantly balance the holistic traffic speed at public level and distinct driving behaviors at personal level.

Our solution consists of several components to deal with the above challenges, producing accurate predictions simultaneously for both holistic traffic speed and customized travel time. It utilizes GPS trajectories, road network, query amount, and auxiliary information (e.g., weather and holiday) to learn general traffic features, and fuses them within a Recurrent Neural Network (RNN) to predict future traffic speed on each road segment. To extract the most effective road traffic features, we learn the topology representation over large-scale road networks, and involve historical traffic statistics and query distribution. To identify personalized driving behaviors,

Manuscript received September 7, 2021; revised January 24, 2022; accepted March 14, 2022. This work was supported in part by the Beijing NSF under Grant L192004; in part by NSFC under Grant 62072029, Grant 61872027, and Grant 61876018; in part by the DiDi Research Collaboration Plan; and in part by the Open Research Fund of the State Key Laboratory of Integrated Services Networks under Grant ISN21-16. An earlier version of this paper was presented at IJCAI'19 [1] [DOI: 10.24963/ijcai.2019/234]. The Associate Editor for this article was Y.-S. Huang. *(Corresponding author: Ruipeng Gao.)*

Ruipeng Gao, Fuyong Sun, and Weiwei Xing are with the School of Software Engineering, Beijing Jiaotong University, Beijing 100044, China (e-mail: rpgao@bjtu.edu.cn; fysun12@bjtu.edu.cn; wwxing@bjtu.edu.cn).

Dan Tao is with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing 100044, China (e-mail: dtao@bjtu.edu.cn).

Jun Fang and Hua Chai are with the Maps and Public Transportation Department, DiDi Corporation, Beijing 100089, China (e-mail: fangjun@didiglobal.com; chaihua@didiglobal.com).

Digital Object Identifier 10.1109/TITS.2022.3160468

[1]According to the statistics, approximately one-third of all traffic fatalities in the US occur due to "aggressive driving" [2].

we explore a series of methods to calibrate the crowdsourced inertial readings on smartphones, then harness them to identify six types of aggressive driving events for each driver, i.e., bump judders, sharp turns, sharp slopes, frequent lane shifts, overspeeds, and sudden brakes. Finally, such common traffic features and individual driving behaviors are further fused within a multi-task learning structure for customized travel time estimation.

Specifically, we make the following contributions:

- We produce accurate inertial measurements for smartphones via automatic calibration. Specially, we estimate the placement of smartphones inside vehicles, calibrate gyroscope readings on straight roads, and explore an LSTM-based inertial tracking method to impute missing trajectories in GPS blocked environments.
- We combine GPS trajectory and inertial measurements to identify six aggressive driving behaviors for each individual driver, i..e., bump judders, sharp turns, sharp slopes, frequent lane shifts, overspeeds, and sudden brakes.
- We propose an unsupervised graph embedding method to capture road topological relations over large-scale road networks. We also extract traffic speed statistics and query distribution via historical traffic database.
- We fuse GPS trajectories, inertial readings, and road network within a deep recurrent neural network. We also explore a multi-task learning structure to predict both traffic speed on each road segments and customized travel time for individual drivers.
- We conduct extensive experiments on two large-scale real-world datasets at Beijing and Shanghai, collected by the DiDi ride-hailing platform. Results have shown our effectiveness compared with the state-of-the-art.

Next, we give a brief overview (Section II), calibrate inertial readings (Section III), identify aggressive driving behaviors (Section IV), and learn road traffic features over road networks (Section V). We further adopt our approach in two applications (Section VI) and report experimental results (Section VII). Finally, we review related work (Section VIII), discuss limitations, and conclude the paper (Section X).

II. Overview

In this section, we present important definitions in Origin-Destination Travel Time Estimation (ODTTE), and depict the overview of our approach.

A. Definitions

Definition 1 (Road Network): Given a fixed region on the map, the road network is defined as the set of underlying road links (i.e., road segments). Each road link includes its geographical location, length, direction, speed class, lane number, and other attributes. Since road intersections sometimes cause a very long queuing time in rush hours, we also treat them as links. Note that this road network model is different from others who define the map as a directed/undirected graph with nodes and edges.

Definition 2 (Path and Trajectory): A driving path is defined as a sequence of points, and each point contains the location

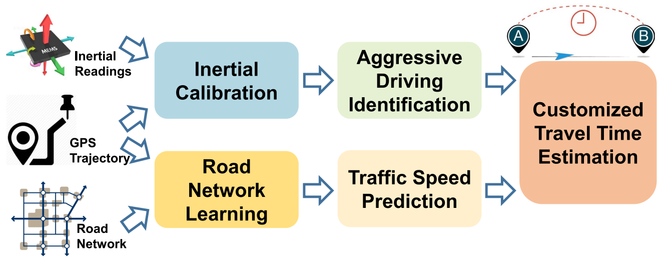


Fig. 1. We take inertial readings, GPS trajectories, and road network as inputs, and there are five components to produce the customized travel time for each driver.

(e.g., latitude and longitude), time in the day, and road link index indicating which road segment it locates on. In addition, a vehicle trajectory $x^{(i)}$ is defined as a tuple with three components, i.e., $x^{(i)} = (u^{(i)}, P^{(i)}, \lambda^{(i)})$, where i is the trajectory ID, $u^{(i)}$ is the driver's ID, $P^{(i)}$ is the driving path, and $\lambda^{(i)}$ is the auxiliary information for this trajectory, including the day index in the week, holiday index, and weather index.

Definition 3 (Aggressive Driving Events): We consider six aggressive driving events as dangerous behaviors: 1) bump judders; 2) sharp turns; 3) sharp slopes; 4) frequent lane shifts; 5) overspeeds; and 6) sudden brakes. When sensing with a smartphone, such events will cause distinct signal patterns in smartphone inertial data.

Given the above definitions, we conceive the customized travel time estimation problem as:

Definition 4 (Problem Statement): During the training phase, we learn: 1) how to estimate the traffic speed on each link via the road network and GPS trajectories, and 2) how to identify and represent aggressive driving behaviors for each individual via inertial readings. During the test phase, given a driver ID, an origin, a destination, and a departure time, our goal is to estimate the travel time for this specific driver, with the path generated by other route planning techniques.

B. System Overview

We utilize smartphone's inertial readings, GPS trajectories, and large-scale road map as inputs for the origin-destination travel time estimation. As Figure 1 shows, there are five components in our approach. First, we produce accurate inertial measurements via automatic calibration, and identify six types of aggressive driving behaviors. Next, we explore an effective topology representation method over large-scale road networks, and fuse with historical traffic features and auxiliary attributes to predict holistic traffic speed on each road link. Finally, we combine the general traffic features and individual driving behaviors to estimate customized travel time for specific drivers.

III. Inertial Calibration

Driving behaviors differ significantly among drivers, especially for rookie and aggressive drivers. Rookie drivers may drive relatively slow, wait longer when queuing, and make violent brakes. Aggressive drivers may shift lanes frequently to overtake others. Thus, driving behavior analysis for each

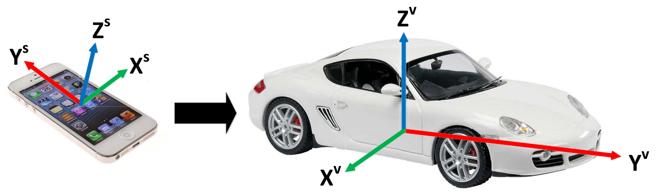


Fig. 2. Smartphone pose estimation, with (X^s, Y^s, Z^s) for the phone's coordinate system and (X^v, Y^v, Z^v) for the vehicle's.

individual is crucial to assess driving performances and estimate personalized travel time more precisely.

Measuring the driving behavior is not trivial. Although some recent applications (e.g., Waze) have already used drivers' preferred routes to provide better path navigation, they largely rely on GPS trajectories with relatively poor accuracy and low sampling rate, e.g., $\sim 5m$ position errors at $1Hz$ for commercial smartphones,[2] thus are too coarse to model many fast driving events such as lane shifts and sudden brakes. Our intuition comes from the inertial data which describes both linear accelerations and rotations of a smartphone at fine-grained level.

However, due to the arbitrary placement of smartphones, potentially large sensor noises, and severely environmental interferences, the inertial readings could be hardly used as-is. In this section, we aim to produce accurate inertial movements via automatic calibration. Specially, we identify the placement of smartphones inside vehicles, devise an angle calibration algorithm to improve orientation estimates, and propose an LSTM-based dead-reckoning method for speed inference in GPS blocked environments. Such calibrated inertial data will be used to identify different aggressive driving behaviors in next section.

A. Pose Estimation

The smartphone inertial data are measured from its built-in IMU (Inertial Measurement Unit) sensors, i.e., 3-axis accelerations by accelerometer and 3-axis angular rate by gyroscope, both in the smartphone's coordinate system.

However, since smartphones may be placed with arbitrary placements in the car, the phone's coordinate system is not always consistent with the vehicle (shown in Figure 2). Thus, we explore a Principal Component Analysis (PCA) algorithm to compute the phone's pose in the car, i.e., estimating the vehicle's coordinate system in the phone's coordinate system.

Our pose estimation solution consists of three steps. Step 1) When the car is static, the gravity direction (i.e., Z-axis of the vehicle) can be computed via a low-pass Butterworth filter to remove high frequency components. Step 2) We use the gravity direction to deduct 3-axis accelerations onto the horizontal plane, thus the vehicle's forward direction (i.e., Y-axis of the vehicle) is caused by accelerating and decelerating, which can be computed as the maximum acceleration direction by a 2D PCA algorithm. Step 3) The rest X-axis of the vehicle is calculated as the cross product of the other two axis directions.

[2]https://www.gps.gov/systems/gps/performance/accuracy

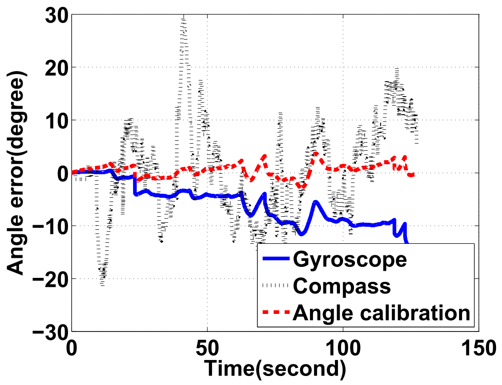


Fig. 3. Real time orientation errors.

Finally, we formulate the vehicle's coordinate system as a 3×3 transmission matrix, and use it to transform the motion data from the phone to the vehicle.

B. Angle Calibration

When tracking a vehicle via the smartphone, driving orientations could be obtained either from the compass, or from the gyroscope. However, we observe that neither sensor provides accurate angle measurements for crowdsourced smartphones: the compass frequently suffers from interferences by surrounding electromagnetic objects including vehicles, while the gyroscope is accurate only for short-time observations and has long-time drifts. Figure 3 shows orientation measurements for a 2-minute driving. Here we omit errors when driving in road turns, since their angle ground truth are difficult to measure.

Although there are some work [12], [13] calibrate angles from the sensor fusion perspective, they focus on walking scenarios which are with unique walking patterns, and they leverage a short time window to provide real time angle estimates. Even though, the angle calibration is still challenging for smartphone-based pedestrian tracking.

Observing that the gyroscope is accurate in short durations and has linear drifts [13], we aim to calculate that constant drift from other cues, so as to track vehicles with the calibrated gyroscope. Our intuition comes from the common sense that vehicle's orientations are consistently on the road, thus we employ the road information to estimate the drift. Figure 3 shows that there are no drifts for the calibrated angle, with the maximum angle error around 4° after 2 minutes.

C. LSTM-Based Inertial Dead-Reckoning

Besides using GPS for speed measurement under open sky, we observe that there are many GPS blocked environments such as in a tunnel or under an overpass. Thus, we need to recover vehicle's speed in case GPS signals are unavailable. A naive method is to conduct double integrations on vehicle's forwarding accelerations a^Y (after pose estimation). However, the low-quality accelerometers inside smartphones are always plagued by heavy noises, and are easily accumulated to unbounded velocity errors.

We aim to explore an inertial dead-reckoning method to infer vehicle's speed in GPS blocked environments. Specially,

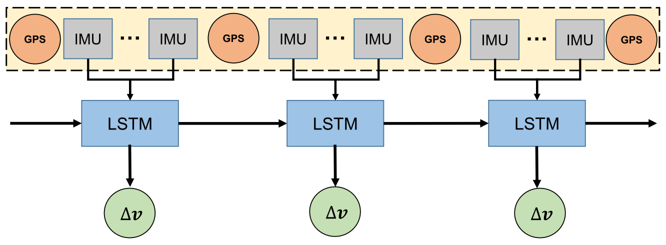


Fig. 4. The training process of inertial dead-reckoning.

we propose an LSTM-based inertial tracking model with only vehicle's forwarding accelerations as inputs, sampling at 50 Hz. When vehicles are driven outdoors, we leverage their GPS speed as the ground truth to train the model, and further deploy the pre-trained model to infer vehicle's speed in GPS blocked environment.

As shown in Figure 4, our model directly produces vehicle's realtime velocity instead of integration, and its output is the speed variance during each time interval, e.g., 1 second in our application, i.e.,

$$f_\Phi : a^Y_{t:t+T} \mapsto \Delta\hat{v}_T \tag{1}$$

where T is the window length, and Φ represents the parameter in the model. Next, we approximate the actual vehicle velocity difference Δv_T from the GPS trajectory, thus our objective is to minimize the total prediction errors, i.e.,

$$\Phi^* = \arg\min_{\Phi} \sum \|\Delta\hat{v}_T - \Delta v_T\|_2^2 \tag{2}$$

In implementation, we choose BasicLSTMCell to establish the LSTM network, the number of hidden nodes is 256, and the batch size is 30. We use the Adam optimizer with learning rate of 0.0001.

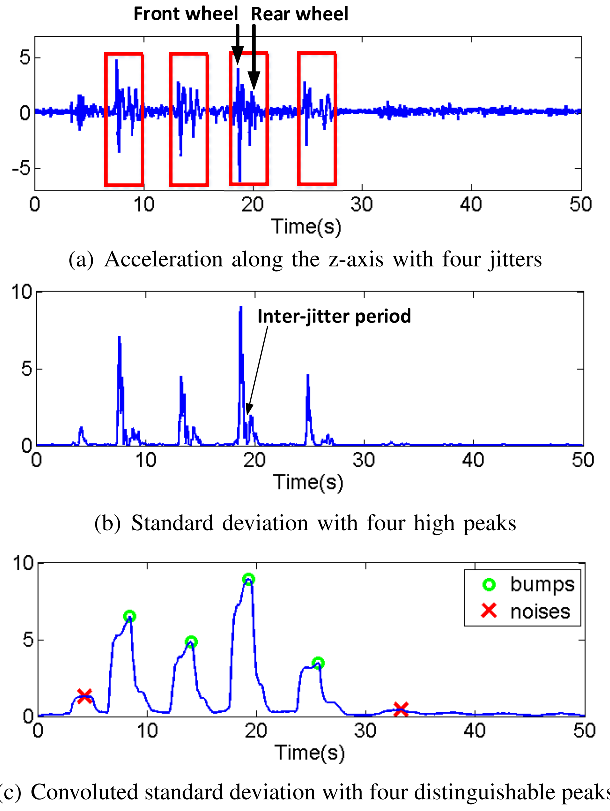
IV. Aggressive Driving Identification

In this section, we aim to identify six aggressive driving behaviors, i.e., bump judders, sharp turns, sharp slopes, frequent lane shifts, overspeeds, and sudden brakes.

Intuitively, our PCA algorithm has transformed the inertial dynamics from the smartphone to the vehicle (elaborated in Section 3.1), and produced vehicle's forwarding acceleration a^Y, vertical acceleration a^Z, and angular speed ω^Z for driving behavior identification. Specially, a^Y corresponds to vehicle's velocity and can be used to detect speeding events (e.g., overspeeds and brakes), a^Z indicates vehicle's vibration and can be used to detect bump judders, while ω^Z refers to vehicle's orientation and can be used to detect rotation events (e.g., turns and lane shifts).

A. Bump Judders

Bumps are used to limit the vehicle speed for sake of safety. When a vehicle passes over a bump with high speed, it will cause severe judders with an unique inertial pattern. Note that some potholes on the road may cause inertial jolting patterns similar as bumps, thus they are also categorized as bumps in this paper.



(a) Acceleration along the z-axis with four jitters

(b) Standard deviation with four high peaks

(c) Convoluted standard deviation with four distinguishable peaks

Fig. 5. Bump detection: (a) four jitters (circled ones in the figure) in the acceleration on the z-axis; (b) four high peaks in the standard deviation of the z-axis acceleration; (c) four distinguishable peaks after applying convolution to the standard deviation with a Gaussian filter, where green (or red) markers indicate the local maximums for detected bumps (or false alarms).

1) Traditional Method: When a vehicle passes over a bump, jitters occur in vehicle's vertical direction (z-axis) can best characterize such jitters. Some previous work [14] calculate the standard deviation of the z-axis accelerations, and use a pre-determined threshold to detect bump-passing. Figure 5(a) shows the vertical acceleration for a car driving over four bumps along a straight road, and the standard deviation is shown in Figure 5(b). The first jitter at the 4th second is from a sudden change of the phone's pose, which is less significant than other four caused by the bumps. However, due to the variation of road conditions, vehicle models and driving styles, it is difficult to find a universal threshold that fits all cases. Moreover, occasional phone pose changes during driving may also cause false alarm (e.g, the one at the 4th second in Figure 5 (b)).

2) Our Algorithm: We propose a robust detection algorithm that is less sensitive to the choice of the threshold and occasional pose changes of the phone. We observe that when driving straight, each bump can incur two consecutive car jolts when its front wheels pass over a bump followed by rear ones. Note that the first jolt usually is more significant than the second as the phone is often placed in the front of a car, and it is also because that the vehicle's speed usually has been reduced when the front wheels pass over the bump. Or a bump may cause four jitters at a turning corner because each of wheels' pass over the bump at different times. These consecutive jitters lead to two or four peaks in the standard

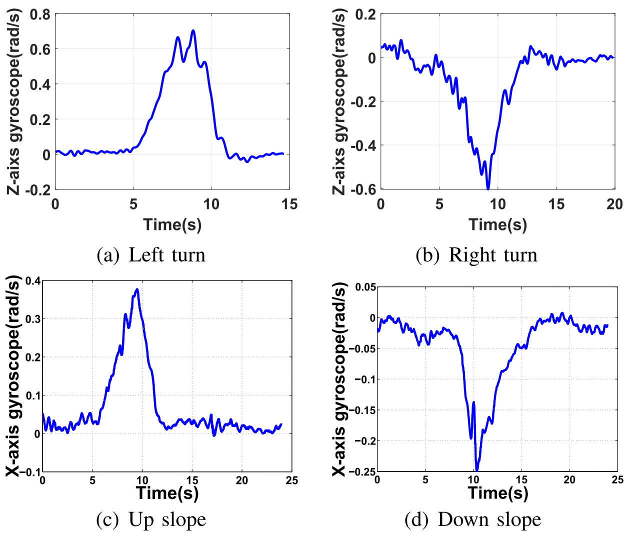


Fig. 6. Inertial sensing data used for turn and slope detection. (a)(b) show turns using gyroscope signal along the z-axis. (c)(d) show slopes using gyroscope signal along the x-axis.

deviation of the z-axis acceleration. We smooth the standard deviation curve by convolving it with a Gaussian filter. The two jitters caused by a bump in Figure 5(b) is merged into a single distinguishable peak in Figure 5(c). The time stamp of the peak is the time when the center of the vehicle is passing over the bump. After the convolution, it is much easier to distinguish jolts caused by bumps from those incurred by sudden phone pose changes, because the latter does not have subsequent followers.

B. Sharp Turns and Slopes

Sharp turns and slopes may trigger dizziness feeling for both drivers and passengers. They can be detected by gyroscope readings when rotation rates are measured: horizontal turns result in rotation rates along the z-axis as shown in Figure 6(a)(b), and slopes lead to rotation rates along the x-axis as shown in Figure 6(c)(d). Based on this observation, we calculate the accumulated rotation angles as the detection statistic, i.e.,

$$C(t; \mathbf{s}, k) = |\sum_{\tau=t-k}^{t} \mathbf{s}(\tau)|, \tag{3}$$

where $\mathbf{s}(\tau)$ is the gyroscope reading at time τ along an axis, and k defines the size of the data collection window. The magnitude of C depends on the angle that the vehicle turns. Based on our observation on raw signals via crowdsensing, we define $k = 150$ (i.e., with 3-second time window) and different thresholds for turn and slopes, i.e.,

- **A sharp turn** is detected at time t if and only if

$$C(t; \mathbf{s}_z, k) > \theta_1 \tag{4}$$

where $\mathbf{s}_z$ is the gyroscope signal along the z-axis. In our implementation, θ_1 is set as $\frac{\pi}{6}$ since the vehicles always take longer time to travel through a turn.

- **A sharp slope** is detected at time t if and only if

$$C(t; \mathbf{s}_x, k) > \theta_2, \tag{5}$$

where $\mathbf{s}_x$ is the gyroscope signal along the x-axis. In our implementation, θ_2 is set as $\frac{\pi}{18}$ since the slope signal is always fast and sharp.

C. Frequent Lane Shifts

Drivers always shift their driving lanes to overtake other vehicles, but frequent lane shifts are regarded as aggressive driving behaviors which may incur traffic accidents, e.g., rear-end collisions. Thus we automatically detect lane shift events and count their frequency.

Specially, lane shifts correspond to a reciprocating motion, e.g., first turn left then immediately turn right within a short time. In addition, the gyroscope readings are known to be accurate within a short time, but suffer from a linear drift. We further explore a complementary filter algorithm [15] to fuse ω^Z with the long-term GPS speed bearing observations and eliminate the drift errors.

D. Overspeeds and Sudden Brakes

Driving at overspeeds or taking sudden brakes are well-known dangerous behaviors at expressway, thus we compare vehicle's GPS speed with the speed constraints on the map to identify such events. In addition, when driving in GPS blocked environments such as in tunnels or under overpasses, we leverage our LSTM-based inertial dead-reckoning method (Section 3.3) to infer the vehicle's velocity via inertial readings, with its customized learning model.

V. Road Network Learning

In this section, we extract the most effective road traffic attributes over large-scale road networks, including road topology relations, historical speed statistics, and user query distribution.

A. Topology Representation

Road topological relations is crucial for both traffic speed prediction and travel time estimation. Some intelligent transportation systems have already assigned successive road segments with the same traffic light lifetime, thus vehicles pass them with a high speed and do not need to stop frequently. However, representing the road topology is not trivial. Simple numerical or one-hot encoded categorical features can not reflect the entire road topology, especially for complex road networks. Graph Laplacian Regularization method [10] enhances the loss with a graph Laplacian factor, thus adjacent links are likely to be assigned with similar representations, but its optimization step fails for large-scale road networks.

Inspired by the unsupervised graph embedding approach in DeepWalk [16], we explore a road topology representation method for large-scale road networks. First, instead of using the RandomWalk algorithm which generates random node sequences to "sample" the graph, we leverage a map-matching method [17] to attach each GPS point onto a specific road link and merge continuous links, thus each trajectory can produce a sequence of link IDs.

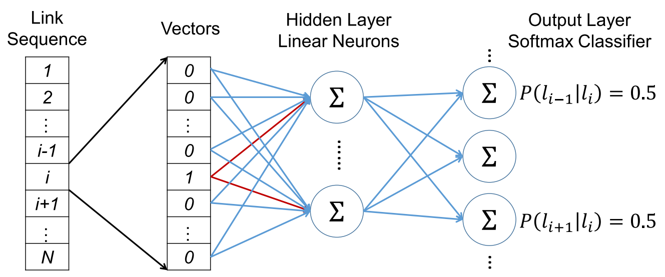


Fig. 7. Illustration of the SkipGram algorithm for a link sequence.

Next, we use the SkipGram language model [18] which maximizes the co-occurrence probability among words in a sentence. Given a link sequence $(l_1, l_2, \ldots, l_N)$, we represent each link as an $N \times 1$ vector by one-hot encoding, and define a neural network to compute the probability of each other link that it is adjacent to link l_i (shown in Figure 7).

In this model, the neural network has only one hidden layer without the activation function, and the output layer uses softmax function to ensure the output vector as a probability distribution. We only keep the weight matrix in hidden layer as link representations. However, when representing a road network with millions of links, the large-scale weight coefficients make the posterior distribution learning (i.e., $P(l_{i-1}|l_i)$ and $P(l_{i+1}|l_i)$) extremely difficult. To speed up its training process, we use the Hierarchical softmax function in DeepWalk. We assign all road links as a sequence of tree nodes, and transform the prediction problem into maximizing the probability of a specific path in the hierarchy, i.e.,

$$P(l_{i-1}|l_i) = \prod_{k=1}^{\lceil \log N \rceil} P(l_{a_k}|l_i) \tag{6}$$

where the path in tree structure from root to node l_{i-1} is denoted as $(l_{a_0}, l_{a_1}, \ldots, l_{a_{\lceil \log N \rceil}})$. In this equation, $P(l_{a_k}|l_i)$ can be learned by a binary classifier:

$$P(l_{a_k}|l_i) = \frac{1}{1 + e^{-r(l_{a_k}) \cdot r(l_i)}} \tag{7}$$

where $r(.)$ is the representation vector of each link.

Finally, we use the stochastic gradient descent (SGD) algorithm for optimization, with the learning rate of 2.5% and decreasing linearly. Figure 8 shows the representation example of a road network with 14 links.

B. Historical Speed Statistics

Historical speed statistics can reflect rush hours and traffic conditions on each link, thus has a direct impact when estimating the travel time. After map matching process, the vehicle trajectory is comprised of a series of GPS points, each with a timestamp and a road link index, thus we can easily compute the average traffic speed on each link.

Suppose a trajectory segment with GPS points $p_{k:k+n}$ locates on link l_i, starting from time t to $t + T$. The link traffic speed for l_i is computed as $l_i^v = \frac{l_i^{length}}{T}$. Notice that vehicles enter the main road on the first link and exit on the last link, actual driving length on these two links are not complete. Thus, we remove the GPS points on these two links. Finally, we summarize all historical trajectories and compute the average traffic speed on each link for each time interval (5 minutes in this paper).

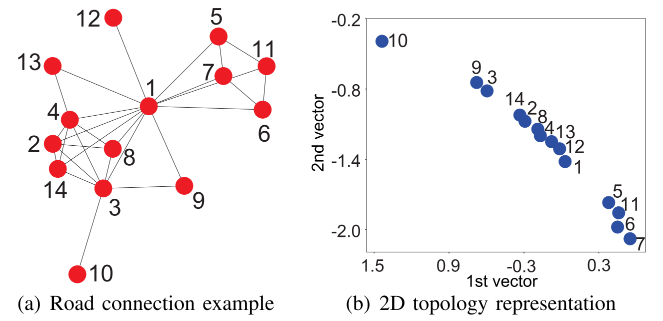


Fig. 8. Topology representation on road network, where (a) shows the connection relationship of an example road network with 14 road segments, and (b) depicts the 2D vector representation for each road segment on its topology relation, i.e., two connected road segments should be embedded as close 2D vectors.

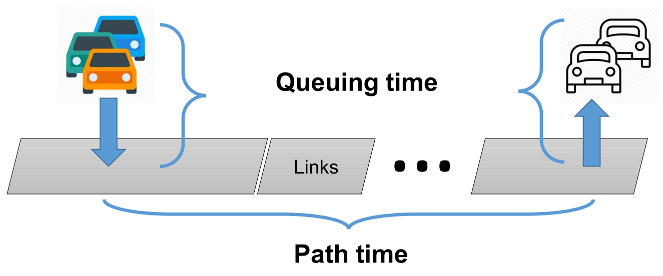


Fig. 9. Illustration of the origin-destination travel time, which consists of the path time on links and the queuing time at inlet/outlet.

C. Query Distribution

As shown in Figure 9, the origin-destination travel time consists not only the path time when driving on road links, but also the queuing time when entering/exiting the main road, especially when we visit a POI (Point of Interest).

By Queuing Theory, the queuing time can be computed via queuing length (amount) and driving speed. Thus, we also analyze the historical query distribution on each link at each time interval. For example, a trajectory starting from link l_A at time t_A and ending from link l_B at time t_B can produce two records of link query: $q_{l_A,IN}^{t_A}$ and $q_{l_B,OUT}^{t_B}$, where IN and OUT denote query direction to links.

VI. Applications

In this section, we aim to predict both public traffic speed on holistic road networks and customized travel time for individual drivers.

A. Model Architecture

The model architecture is shown in Figure 10, which combines both traffic speed prediction and customized travel time estimation via mobile crowdsensing.

First, we collect the crowdsourced traffic data and produce two databases. 1) **Road link traffic**, derived from Section V, which contains road link topology representation, historical

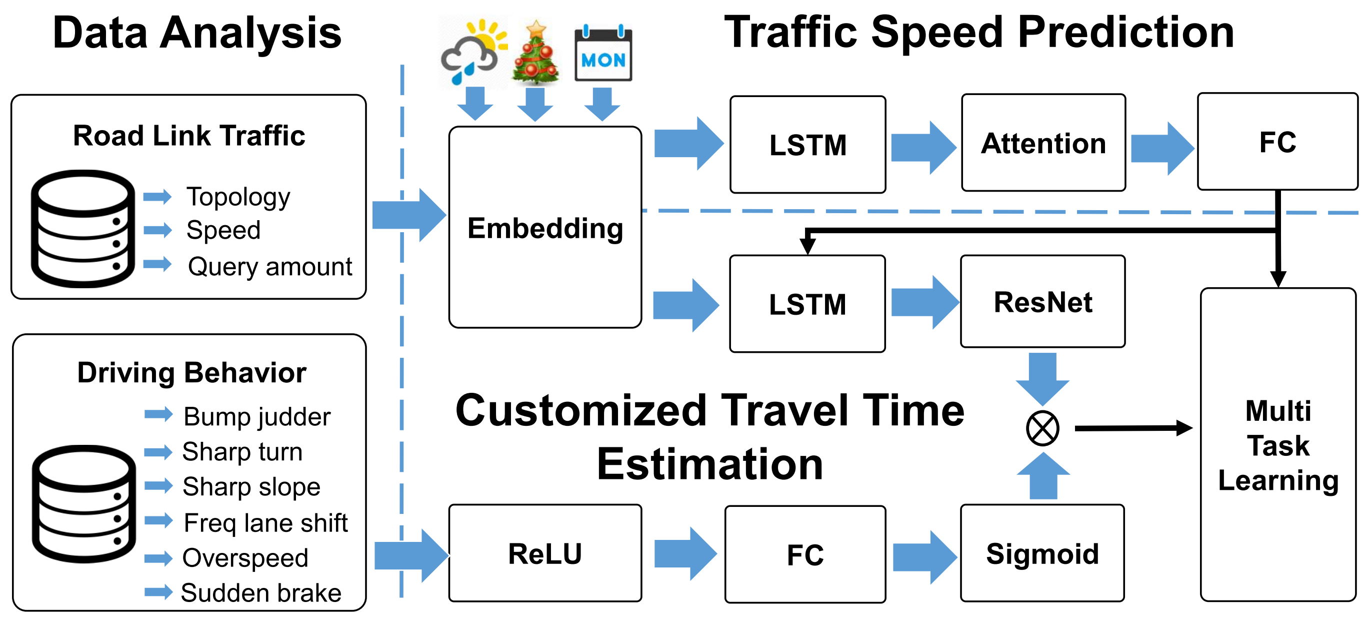


Fig. 10. In data analysis stage, we produce two databases on road link network and driving behavior. We further propose a multi-task learning structure to predict both traffic speed and customized travel time.

speed statistics during different time, and query distribution on each road segments. 2) **Driving behavior**, derived from Section IV, which identifies six aggressive driving behaviors for each driver, including bump judders, sharp turns and slopes, frequent lane shifts, overspeed events, and sudden brakes.

Next, we fuse the multi-source heterogeneous traffic data within a deep recurrent neural network, and explore a multi-task learning structure for both traffic speed prediction at a holistic level and customized travel time estimation at a personal level. Below we present the detailed design for each model, respectively.

B. Link Speed Prediction

Our link traffic database contains most useful traffic features for links: the link topology captures adjacency relations, the historical speed reflects busy hours, and the query amount measures queuing time. We also implement a web crawler to extract other auxiliary attributes such as the weather, the holiday, and the day in week information.

With such heterogeneous data inputs from multiple sources, we incorporate them and use the embedding method [19] to transform those categorical attributes into low-dimensional vectors, thus can feed them into the neural network.

To further capture the temporal dependencies among road links, we apply the recurrent neural network (RNN) to learn long-term temporal patterns. RNN has been widely used in sequential learning on natural language processing, machine translation, and speech recognition. In our model, the input vector for each link i is constructed as the concatenation of all attributes, i.e.,

$$x_i = relu(W_x \cdot [x_i^{topology} \circ x_i^{speed} \circ x_i^{query} \circ x_i^{auxiliary}]) \quad (8)$$

where $x_i^{topology}$, x_i^{speed}, x_i^{query}, and $x_i^{auxiliary}$ denote the embedded vector of topology, speed, query amount, and auxiliary information for each link, respectively. W_x is the weight matrix.

Next, we input the concatenated vector into a LSTM [20] structure as the RNN implementation, and obtain the current hidden variable h_t as:

$$h_t = LSTM(x_i, h_{t-1}) \quad (9)$$

We also use the soft attention mechanism [21] to capture the weights of a sequence of LSTM's output. Finally, the output of attention is connected to an FC layer, then compared with speed statistics to calculate the speed prediction loss L_{speed}.

C. Customized Travel Time Estimation

In addition to predicting current traffic speed, estimating the origin-destination travel time for each individual is also meaningful in many intelligent transportation systems and applications. However, driving skills vary obviously among different drivers, thus we should also consider their driving behaviors for customized travel time estimation.

Given the path of a trajectory, we first build an LSTM network to capture the temporal and spatial (topological) features of travel time. The hidden state output of LSTM is connected to a ResNet (Residual Neural Network [22]) module for decoding.

To identify the aggressive driving events for each driver, we use smartphone inertial data with pre-tuned filters to generate different channels of driving behaviors (details elaborated in Section 3.3). We further concatenate such driving behavior data and process them sequentially with a $ReLU$ function, an FC layer, and a $sigmoid$ function to calculate a driver-personalized scale factor.

Next, we use the scale factor to amend the output of ResNet by multiplication. Note that this driver-personalized scale factor is better than simple driver ID used in other

TABLE I
DATASET INFORMATION

Dataset	Beijing	Shanghai
Time	8.6-8.26, 2018	8.6-8.26, 2018
Traces	3.9GB (410,882)	2.6GB (270,716)
Links	513KB (12,600)	350KB (8,500)
Inertial data	2.3GB	1.22GB

methods [8], [11], since it can match similar driving behaviors among different drivers, and remain effective even for new users with few historical data.

Finally, to train our model, we formulate a multi-task learning problem, and our objective is to minimize the combination of both the speed loss and time loss, i.e.,

$$L_{time} + \alpha \cdot L_{speed} \tag{10}$$

where the coefficient α weights the speed loss item. During training, we leverage the mean square error (MSE) as the loss function for speed prediction, and use the mean absolute percentage error (MAPE) for travel time estimation.

VII. EXPERIMENTS

We evaluate our proposed method on two large-scale real-world traffic datasets, which is collected crowdsourcingly by the DiDi ride-hailing platform. We also compare it with the latest existing approaches for effectiveness.

A. Datasets and Baseline Algorithms

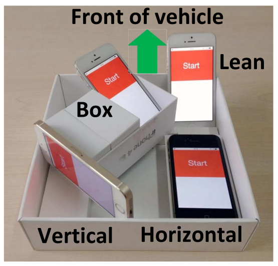
Our traffic datasets are gathered in Beijing and Shanghai, the largest two cities in China with millions of vehicles. The time period is both three weeks, from Aug. 6th to Aug. 26th, 2018. We have defined the same data format for each dataset, consisting of heterogeneous sensory data from multiple sources. Each dataset contains GPS trajectories, inertial data, road network, and auxiliary information (weather index, holiday index, the day in the week). In addition, the GPS points on each trace have been projected onto the road link via a Map Matching algorithm [17]. Table I depicts the details for each dataset.

In this experiment, we implement our model with PyTorch toolbox, and train the model on 1080Ti with 32GB memory. A typical training process takes about 30 minutes on link typology representation, and about 17 hours on the multi-task learning for traffic speed prediction and travel time estimation.

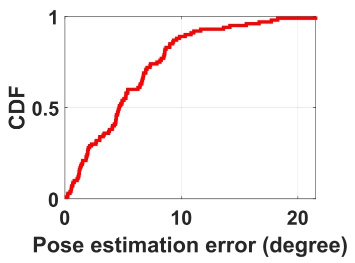
We further compare with Support Vector Regression (SVR) [23] and DeepTTE [8]. SVR uses SVM for regression, which has been widely used in sequence prediction. DeepTTE is the state-of-the-art approach for travel time estimation and it is an open source project. We leverage the mean absolute percentage error (MAPE), the mean absolute error (MAE), and the mean square error (MSE) to measure the accuracy.

B. Smartphone Pose Estimation

To measure the pose estimation accuracy, we use a mould to hold four iPhones with different poses (shown in Figure 11(a)), and measure their ground truth poses via a protractor. The Cumulative Distribution Function (CDF) curve of pose estimation errors is shown in Figure 11(b), with the 90-percentile error at 10 degrees.



(a) Mould



(b) Pose estimation

Fig. 11. Phone pose estimation for four iPhones in a mould.

TABLE II
PERFORMANCE OF LANDMARK DETECTION WITH DIFFERENT POSES

	Horizontal	Lean	Vertical	Box
Precision	93%	88%	85%	93%
Recall	88%	89%	87%	85%

C. Aggressive Driving Detection

Here we evaluate the performance of aggressive driving detection using the precision and recall as metrics. We set breakpoints when a certain driving behavior is announced to be detected, and check whether it corresponds to a correct event at that time stamp. We also compute how many aggressive driving behaviors the vehicle goes through as the ground truth number.

Figure 12 shows the detection recall and precision for six aggressive driving behaviors, i.e., bump judders, sharp turns, sharp slopes, overspeeds, sudden brakes, and land shifts. We observe that the detections of sharp turn are all correct, then comes the lane shifts and sharp slopes. The detection of bump judders has the lowest precision at 87%, and lowest recall at 83%. This is because the gyroscope is much more precise than accelerometer inside smartphones, and there are many driving activities and road conditions which are confused with bumps.

Table II further depicts the bump judder detection performance with four different poses of smartphone inside vehicles. We observe that all precisions are quite high, while twos are relatively lower. This is because that in some positions, the smartphones are also sensitive to the jolting of the car, which may falsely be detected as a bump.

D. Link Speed Prediction

1) Speed Prediction: Figure 13(a) and Figure 13(b) show the MSE and MAE results on link speed prediction, respectively. We observe that the road network is effective to link speed prediction accuracy, reducing the MSE from 22.5 to 16.9, and reducing the MAE from 2.4 to 2.3.

2) Effect of Attention: With the road network, our attention mechanism further reduces MSE from 16.9 to 16.2, and reduces MAE from 2.3 to 2.1. Figure 14 presents the weights of each attention channel, where we divide the time period with a 5-minutes interval, $t0$ presents the same time period in last week, and $t1 - t12$ denote 12 continuous time periods in last hour. Since our temporal feature is fine-grained, the time period in last 5 minutes ($t12$) has the largest weight.

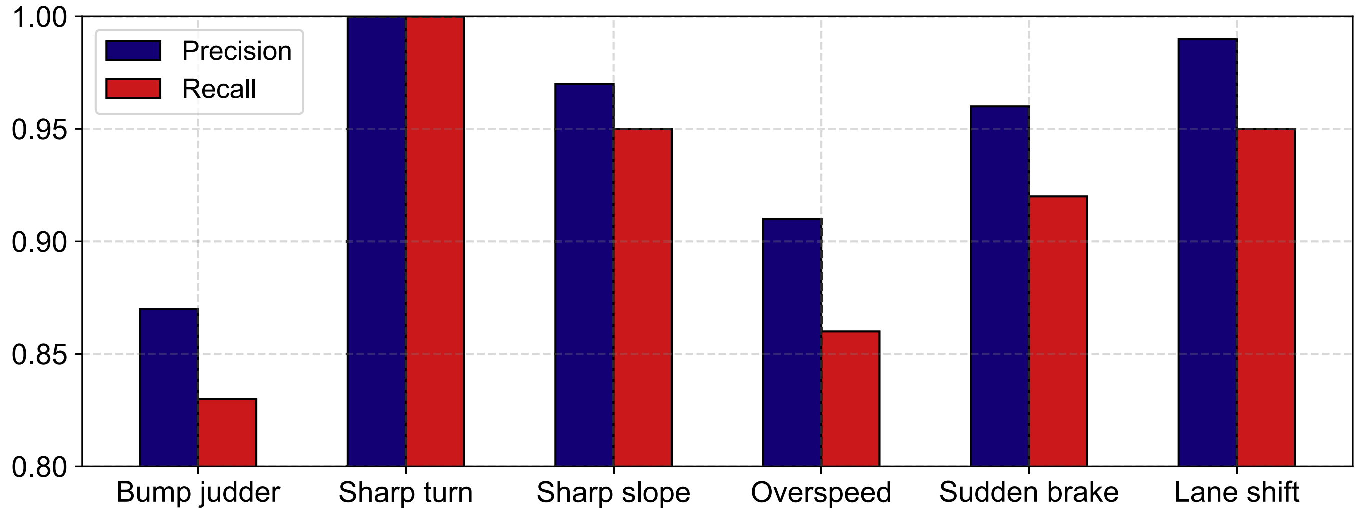


Fig. 12. Detection precision and recall on six aggressive driving behaviors.

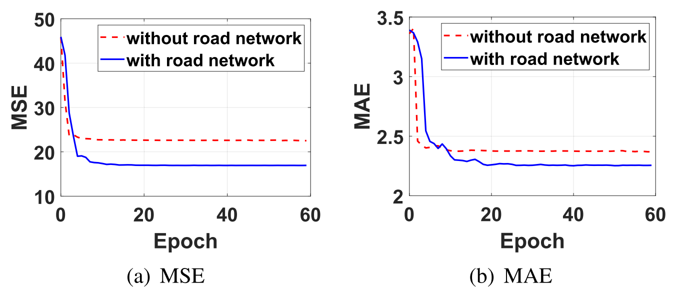


Fig. 13. Link speed prediction accuracy, with/out road network.

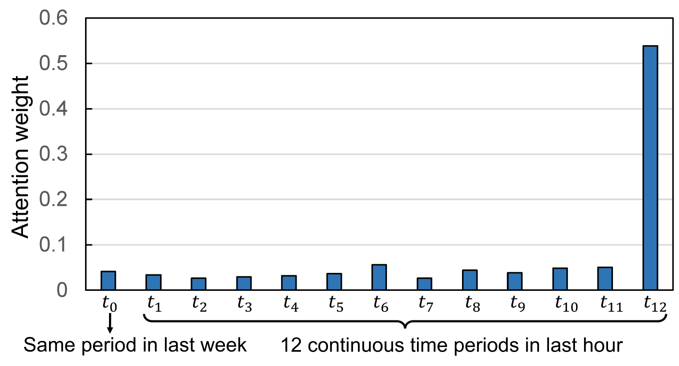


Fig. 14. Attention weights, where we divide the time period with a 5-minute interval, t0 presents the same time period in last week, and $t1 - t12$ denote 12 continuous time periods in last hour.

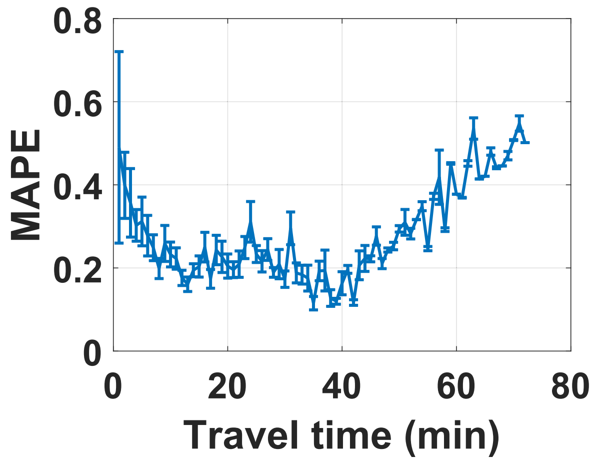


Fig. 15. MAPE w.r.t. time.

TABLE III
TRAVEL TIME ESTIMATION WITH DIFFERENT INPUTS

Road network	IMU	MSE	MAE	MAPE
ID	no	220290.5	280.4	23.9%
Topology	no	179017.5	255.5	23.1%
Topology	yes	177818.1	251.6	22.6%

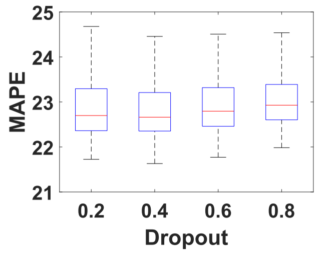
E. Travel Time Estimation

1) MAPE w.r.t. Travel Time: Figure 15 shows the error bar graph on MAPE value for each trajectory with its travel time period. The travel time period is relatively wide, varying from 1 minute to more than 70 minutes among trajectories. As consistent with our common sense, longer travel time always causes worse prediction results but lower variances. An interesting observation is that extreme short travel time (e.g., less than 10 minutes) also leads to large prediction errors, due to variety of environmental interferences besides the travel time.

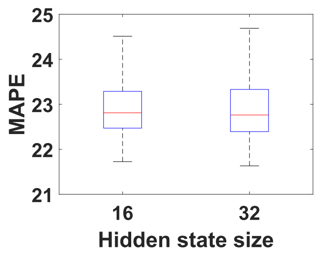
2) Effect of Network Structure: We evaluate the effects of road network and IMU data in our neural network.

a) Road network: Table III shows the comparison between ID embedding and our topology embedding on road network. The ID embedding result is based on the initial embedding API in Pytorch. We observe that our topology embedding method for road network helps to converge the loss to a much lower level, e.g., reducing the MAPE with 0.8% (first two rows).

b) IMU data: In implementation, we concatenate the inertial features and input them with an FC layer and a $sigmoid$ function to get a scale factor, then amend the output of neural network with this scale factor (multiplication). Since the IMU data is very sparse at current DiDi platform, it only reduces the MAPE loss with 0.5% (details shown in Table III).



(a) Dropout



(b) Hidden state size

Fig. 16. Effects of two hyper-parameters.

TABLE IV
TRAVEL TIME ESTIMATION WITH/OUT INDIVIDUAL DRIVING BEHAVIORS

Dataset	Driving behaviors	MSE	MAE	MAPE
Beijing	With	177818.1	251.6	22.6%
	Without	185974.3	297.5	23.2%
Shanghai	With	245917.4	304.7	24.3%
	Without	264857.2	387.5	25.2%

TABLE V
PERFORMANCE ON BEIJING DATASET

	MSE	MAE	MAPE
SVR	15081.9	100.7	83.1%
DeepTTE	939957.4	392.5	25.8%
CTTE	177818.1	251.6	22.6%

TABLE VI
PERFORMANCE ON SHANGHAI DATASET

	MSE	MAE	MAPE
SVR	16226.0	103.1	100.6%
DeepTTE	675110.2	445.8	29.1%
CTTE	245917.4	304.7	24.3%

However, this experiment proves the effectiveness of IMU data, and we will test more fine-grained IMU data in the future for more improvements.

3) Effect of Individual Driving Behaviors: Table IV shows the accuracy of travel time estimation with/out individual driving behaviors, in Beijing and Shanghai dataset. We observe that the distinct driving behaviors consistently reduces MSE, MAE, and MAPE values in both datasets. This demonstrates the effectiveness of individual driving behaviors for customized travel time estimation.

4) Effects of Hyper-Parameters: Figure 16(a) and Figure 16(b) show the effects on two hyper-parameters in our model, i.e., the dropout value and hidden state size, respectively. We have set the dropout value with $\{0.2, 0.4, 0.6, 0.8\}$, and the number of LSTM hidden state size with 16 and 32. The experiment for each hyper-parameter set is repeated by 10 times. The corresponding MAPE value is all around 23%. Such experiment shows the robustness of our model with such hyper-parameters.

5) Comparison With SVR and DeepTTE: Table V and Table VI depict the final accuracy of travel time estimation on two real-world datasets in Beijing and Shanghai, respectively. We observe that the SVR achieves the least MSE and MAE, but its MAPE is almost 4x than ours; the DeepTTE achieves similar MAPE as ours, but with an extreme high MSE; our approach CTTE produces the least MAPE, and maintains both MSE and MAE at very low level. Thus, our method is superior at all three loss metrics.

TABLE VII
COMPUTATIONAL PERFORMANCE

	Beijing		Shanghai	
	Training	Inference	Training	Inference
SVR	0.93 min/epoch	0.08 min	1.16 min/epoch	0.08 min
DeepTTE	2.28 min/epoch	0.19 min	3.75 min/epoch	0.17 min
CTTE	4.25 min/epoch	0.26 min	6.62 min/epoch	0.23 min

6) Computational Performance: In order to achieve a fair comparison, the training time is computed on one epoch, and the inference time is operated on test samples. Table VII demonstrates the computational performance on Beijing and Shanghai datasets, respectively. Since SVR simply uses SVM for regression, it costs the least training and inference time. Both DeepTTE and CTTE employ deep neural networks for travel time estimation, and our CTTE collects additional inertial readings to learn driving behaviors, thus it costs slightly higher computation than DeepTTE.

VIII. RELATED WORK

A. Vehicle State Estimation

There have been many research efforts using smartphones' embedded sensors to monitor the states of vehicles (e.g. instantaneous velocity and direction of travel [26], dangerous driving alert [27] and CarSafe [28]); inspect the road anomaly or conditions (e.g., Pothole Patrol [14]); and detect traffic accidents (Nericell [29] and WreckWatch [30]). The vehicle speed is a critical input in many such applications. While it is easy to calculate the speed using GPS outdoors [31], the signal can be weak or even unavailable for indoor parking lots. Some alternative solutions leverage the phone's signal strength to estimate the vehicle speed [32], and tackle a multi-target tracking problem in indoor applications via sensor networks [33]. We use inertial data only, thus getting rid of RF signals or extra sensor instrumentation.

B. Traffic Speed Prediction

Traffic speed prediction plays a fundamental role in intelligent transportation system. There are generally two kinds of methods for traffic speed prediction: the parametric and the non-parametric approaches. ARIMA [34] is a classic parametric method and models the traffic in a stationary process. However, those parametric methods are known to be not suitable for large-scale data due to the heavy computation complexity. Some supervised learning approaches, e.g., LR [35], formulate the traffic speed prediction issue as a regression problem. Currently, deep learning models (e.g., CNN [5], LSTM [24], and DLSTM [25]) are used for traffic speed prediction based on large-scale historical traffic data, but they don't consider the link road topology information.

C. Travel Time Estimation

Travel time estimation is very important to location-based services for vehicle navigation applications. The existing approaches can be classified into two categories, the route-based solutions and the data-driven solutions. The first [3]–[5] estimates total travel time as the time summation

TABLE VIII
COMPARISON WITH RELATED WORK

Model	Aim	Input	Limitation
CNN [5]	Speed prediction	GPS trajectory	Without road network
LSTM [24], DLSTM [25]	Speed prediction	Vehicle speed detector	Lack of spatial correlations
SVR [23]	Travel time estimation	Vehicle speed detector	Error accumulation
DeepTTE [8] (CNN+LSTM+Attention)	Travel time estimation	GPS trajectory and auxiliary attributes	Without road network
WDR [11] (MLP+LSTM)	Travel time estimation	GPS trajectory, road link id, and auxiliary attributes	Without road topology, without driving behaviors
Ours (Graph embedding+LSTM+Attention)	Speed prediction & Travel time estimation	GPS trajectory, road network, inertial readings, and auxiliary attributes	Slightly higher computation

on each road segment and intersection. The second [6]–[10] formulates the travel time estimation as a multivariate time series prediction problem. However, they fail to consider general traffic conditions and personalized driving behaviors. Some recent work [11] begins to use the personalized information, but it is simply driver ID and largely relies on GPS data which are too coarse to model many fast driving events, e.g., lane shifts.

D. Driving Behavior Analysis

Thanks to the widely used smartphones for driving navigation, there have been several approaches to use the smartphone's inertial data to monitor the driving behaviors, e.g., dangerous driving alert [27], traffic accidents detection [29], and road landmark detection [36]. Among them, driving speed is the only critical factor to dangerous driving, while we also consider rotations. Besides the three aggressive driving events in this paper, we aim to identify other pivotal events, e.g., not remaining aloof. We plan to explore computer vision algorithms to detect the distance to a predecessor car via a dashboard camera.

E. Summary

Detailed comparisons with the above related work are shown in Table VIII. Specially, we employ the inertial data from commodity smartphones to learn individual driving behaviors, and utilize the road link topology to capture traffic spatial correlations. We also explore a multi-task learning framework to effectively combine both traffic speed prediction at holistic level and travel time estimation for specific drivers.

IX. DISCUSSION

A. Road Map Construction

We observe that there always exists several missing road segments on commodity road maps, e.g., multi-level roads in parking structures, internal roads at residential areas, and dirt roads in the wild. A naive solution is to measure such data via dedicated human efforts, but it is time-consuming and effort-expensive. Existing SLAM approaches always adopt the occupancy grid map to continuously update the constructed map. For example, they assume the Gaussian distribution to represent the uncertainty of trajectories, and use them to compute the accessible probability for each cell on the map. We have done the map construction experiments in an underground parking structure, but found that different trajectories can be hardly integrated due to the variety of their scales and the lack of anchor points, thus its robustness remains weak especially via crowdsensing.

B. Map Topology Extraction

The map topology structure is crucial for both vehicle tracking and navigation. It can restrain GPS noises with the map matching technique, and produce detailed driving routes with distinct landmarks along the path. Existing geometry algorithms, e.g., the Voronoi diagram, can produce the skeleton of reconstructed road maps. We also plan to identify different landmarks (e.g., bumps and turns) via inertial measurements, and mark their locations on the map.

C. Personal Thresholds Production

We have set two thresholds to identify the sharp turns and slopes during driving. Such fixed thresholds are not robust for crowdsourced drivers, thus we plan to investigate personal thresholds production via a "leader-follower" mechanism in our future work. For example, the leaders manually annotate a few turns and slopes on the map, and the followers extract their corresponding inertial readings when they encounter such areas, thus they adjust their distinct thresholds by auto-annotation.

D. Data Imputation on Sparse Roads

In practice, we observe that some road links may never been traveled by any sample trajectories due to the low penetration of probe vehicles during the predict time slot, i.e., the sparsity issue in travel time estimation. In order to address the sparsity issue, we plan to capture the spatio-temporal correlations of large-scale road networks, e.g., employing the Graph Convolutional Network (GCN) with adjacent matrix and feature matrix on road network topology, and utilizing the Temporal Convolutional Network (TCN) on historical time series.

X. CONCLUSION

In this paper, we exploit GPS trajectories, smartphone inertial data, and road network to produce customized travel time estimation. It addresses one interesting problem to the ubiquitous vehicle location-based services: how much time your aggressive driving behavior saves. Our solution enables holistic traffic speed monitoring on each road segments, and customized travel time estimation with aggressive driving behaviors. We have conducted extensive experiments in large-scale real-world datasets at Beijing and Shanghai, and the results demonstrate the effectiveness of our method compared with the state-of-the-art.

In the future, we plan to address the sparsity issue in travel time estimation over large-scale road networks, produce personal thresholds for aggressive driving detection, and impute missing road segments via mobile crowdsensing.

References

[1] R. Gao *et al.*, "Aggressive driving saves more time? multi-task learning for customized travel time estimation," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, p. 1689.

[2] (2019). *Aggressive Driving*. [Online]. Available: https://www.in.gov/bmv/files/Drivers_Manual_Chapter_5.pdf

[3] R. Sevlian and R. Rajagopal, "Travel time estimation using floating car data," 2010, *arXiv:1012.4249*.

[4] B. Pan, U. Demiryurek, and C. Shahabi, "Utilizing real-world transportation data for accurate traffic prediction," in *Proc. IEEE 12th Int. Conf. Data Mining*, Dec. 2012, pp. 595–604.

[5] J. Wang, Q. Gu, J. Wu, G. Liu, and Z. Xiong, "Traffic speed prediction and congestion source exploration: A deep learning method," in *Proc. IEEE 16th Int. Conf. Data Mining (ICDM)*, Dec. 2016, pp. 499–508.

[6] M. Rahmani, E. Jenelius, and H. N. Koutsopoulos, "Route travel time estimation using low-frequency floating car data," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 2292–2297.

[7] H. Wang, Y.-H. Kuo, D. Kifer, and Z. Li, "A simple baseline for travel time estimation using large-scale trip data," in *Proc. 24th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, Oct. 2016, pp. 1–22.

[8] D. Wang, J. Zhang, W. Cao, J. Li, and Y. Zheng, "When will you arrive? estimating travel time based on deep neural networks," in *Proc. AAAI*, 2018, pp. 2500–2507.

[9] H. Zhang, H. Wu, W. Sun, and B. Zheng, "DeepTravel: A neural network based travel time estimation model with auxiliary supervision," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3655–3661.

[10] Y. Li, K. Fu, Z. Wang, C. Shahabi, J. Ye, and Y. Liu, "Multi-task representation learning for travel time estimation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 1695–1704.

[11] Z. Wang, K. Fu, and J. Ye, "Learning to estimate the travel time," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 858–866.

[12] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: Zero-effort crowdsourcing for indoor localization," in *Proc. MobiCom*, 2012, pp. 293–304.

[13] P. Zhou, M. Li, and G. Shen, "Use it free: Instantly knowing your phone attitude," in *Proc. ACM MobiCom*, 2014, pp. 605–616.

[14] J. Eriksson, L. Girod, B. Hull, R. Newton, S. Madden, and H. Balakrishnan, "The pothole patrol: Using a mobile sensor network for road surface monitoring," in *Proc. ACM MobiSys*, 2008, pp. 29–39.

[15] S. Shen, M. Gowda, and R. Roy Choudhury, "Closing the gaps in inertial motion tracking," in *Proc. 24th Annu. Int. Conf. Mobile Comput. Netw.*, Oct. 2018, pp. 429–444.

[16] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proc. KDD*, 2014, pp. 701–710.

[17] G. R. Jagadeesh, T. Srikanthan, and X. D. Zhang, "A map matching method for GPS based real-time vehicle location," *J. Navigat.*, vol. 57, no. 3, pp. 429–440, Sep. 2004.

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*.

[19] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Proc. NIPS*, 2016, pp. 1027–1035.

[20] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher, "DeepSense: A unified deep learning framework for time-series mobile sensing data processing," in *Proc. WWW*, 2017, pp. 351–360.

[21] Y. Liang, S. Ke, J. Zhang, X. Yi, and Y. Zheng, "GeoMAN: Multi-level attention networks for geo-sensory time series prediction," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3428–3434.

[22] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. ECCV*, 2016, pp. 630–645.

[23] C.-H. Wu, J.-M. Ho, and D. T. Lee, "Travel-time prediction with support vector regression," *IEEE Trans. Intell. Transp. Syst.*, vol. 5, no. 4, pp. 276–281, Dec. 2004.

[24] X. Ma, Z. Tao, Y. Wang, H. Yu, and Y. Wang, "Long short-term memory neural network for traffic speed prediction using remote microwave sensor data," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 187–197, May 2015.

[25] F. Ding, Z. Zhang, Y. Zhou, X. Chen, and B. Ran, "Large-scale full-coverage traffic speed estimation under extreme traffic conditions using a big data and deep learning approach: Case study in China," *J. Transp. Eng., A, Syst.*, vol. 145, no. 5, May 2019, Art. no. 05019001, doi: 10.1061/JTEPBS.0000230.

[26] J. C. Herrera, D. B. Work, R. Herring, X. Ban, Q. Jacobsond, and A. M. Bayen, "Evaluation of traffic data obtained via GPS-enabled mobile phones: The mobile century field experiment," *Transp. Res. C, Emerg. Technol.*, vol. 18, no. 4, pp. 568–583, Aug. 2010.

[27] J. Lindqvist and J. Hong, "Undistracted driving: A mobile phone that doesn't distract," in *Proc. ACM MobiSys*, 2011, pp. 70–75.

[28] C.-W. You *et al.*, "Carsafe app: Alerting drowsy and distracted drivers using dual cameras on smartphones," in *Proc. ACM MobiSys*, 2013, pp. 13–26.

[29] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: Using mobile smartphones for rich monitoring of road and traffic conditions," in *Proc. ACM SenSys*, 2008, pp. 357–358.

[30] J. White, C. Thompson, H. Turner, H. Turner, and D. C. Schmidt, "Wreckwatch: Automatic traffic accident detection and notification with smartphones," *Mobile Netw. Appl.*, vol. 16, no. 3, pp. 285–303, Jun. 2011.

[31] B. Hoh *et al.*, "Virtual trip lines for distributed privacy-preserving traffic monitoring," in *Proc. 6th Int. Conf. Mobile Syst., Appl., Services (MobiSys)*, 2008, pp. 15–28.

[32] G. Chandrasekaran *et al.*, "Vehicular speed estimation using received signal strength from mobile phones," in *Proc. 12th ACM Int. Conf. Ubiquitous Comput.*, Sep. 2010, pp. 237–240.

[33] D. Ciuonzo, A. Buonanno, M. D'Urso, and F. Palmieri, "Distributed classification of multiple moving targets with binary wireless sensor networks," in *Proc. 14th Int. Conf. Inf. Fusion*, Jul. 2011, pp. 1–8.

[34] B. M. Williams and L. A. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov. 2003.

[35] G. Ristanoski, W. Liu, and J. Bailey, "Time series forecasting using distribution enhanced linear regression," in *Advances in Knowledge Discovery and Data Mining*. Berlin, Germany: Springer, 2013, pp. 484–495.

[36] R. Gao, M. Zhao, T. Ye, F. Ye, Y. Wang, and G. Luo, "Smartphone-based real time vehicle tracking in indoor parking structures," *IEEE Trans. Mobile Comput.*, vol. 16, no. 7, pp. 2023–2036, Jul. 2017.

Ruipeng Gao received the B.S. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2010, and the Ph.D. degree from Peking University, Beijing, in 2016. He was a Visiting Scholar with Purdue University, USA, in 2019. He is currently an Associated Professor with the School of Software Engineering, Beijing Jiaotong University, Beijing. His research interests include mobile computing and applications, the Internet of Things, and intelligent transportation systems.

Fuyong Sun received the B.S. degree in computer science and technology from Jining University, Jining, China, in 2016, and the M.S. degree in software engineering from Beijing Jiaotong University, Beijing, China, in 2019, where he is currently pursuing the Ph.D. degree in software engineering. He was a Visiting Student with Purdue University, USA, in 2019. His research interests include deep learning and urban computing.

Weiwei Xing (Member, IEEE) received the B.S. degree in computer science and technology and the Ph.D. degree in signal and information processing from Beijing Jiaotong University, Beijing, China, in 2001 and 2006, respectively. She was a Visiting Scholar at the University of Pennsylvania, PA, USA, from 2011 to 2012. She is currently a Professor at the School of Software Engineering, Beijing Jiaotong University. Her research interests include computer vision, pattern recognition, and intelligent transportation algorithms and applications.

Dan Tao received the B.S. and M.S. degrees from the College of Computer Science and Technology, Jilin University, China, in 2001 and 2004, and the Ph.D. degree from the School of Computer Science, Beijing University of Posts and Telecommunication, Beijing, China, in 2007. She was a Visiting Scholar with the Wireless Networking Laboratory, Illinois Institute of Technology, USA, from 2010 to 2011. She currently works as a Professor with the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing. Her research interests include wireless networks, the IoT, mobile computing, and big data security.

Jun Fang received the B.S. degree from the Hefei University of Technology, Hefei, China, in 2009, and the M.S. degree from the University of Science and Technology of China, Hefei, in 2012. He is currently the Chief Algorithm Engineer with the Maps and Public Transportation Department, DiDi Company, Beijing, and in charge of estimated time of arrival and route planning. His research interests include machine learning, deep learning, and spatio-temporal forecasting.

Hua Chai received the B.S. and M.S. degrees in computer science from Tianjin University, Tianjin, China, in 2006 and 2009, respectively. He is currently the General Manager of the DiDi Maps and Public Transportation Department, DiDi Company, Beijing, China. He has rich experience in maps technology, internet advertising, large complex distributed systems (including storage, computing, and scheduling), big data, and machine learning.