Radar: Adversarial Driving Style Representation Learning With Data Augmentation

Zhidan Liu, *Member, IEEE*, Junhong Zheng, Jinye Lin, Liang Wang, and Kaishun Wu, *Member, IEEE*

Abstract—Characterizing human driver's driving behaviors from global positioning system (GPS) trajectories is an important yet challenging trajectory mining task. Previous works heavily rely on high-quality GPS data to learn such driving style representations through deep neural networks. However, they have overlooked the driving contexts that greatly govern drivers' driving activities and the data sparsity issue of practical GPS trajectories collected at a low-sampling rate. Besides, existing works omit the cold start problem, where the newly joined drivers usually have insufficient data to learn accurate driving style representations. To address these limitations, we present an adversarial driving style representation learning approach, named Radar. In addition to summarizing statistic features from raw GPS data, Radar also extracts contextual features from three aspects of road condition, geographic semantic, and traffic condition. We exploit the advanced semi-supervised generative adversarial networks to construct our learning model. By jointly considering statistic features and contextual features, the trained model is able to efficiently learn driving style representations from practical GPS trajectory data. Furthermore, we enhance Radar's representation learning for drivers owning limited training data with some basic data augmentation strategies and a novel auxiliary driver based data augmentation method. Experiments on two benchmark applications, i.e., driver identification and driver number estimation, with a large real-world GPS trajectory dataset demonstrate that Radar can outperform the state-of-the-art approaches by learning more effective and accurate driving style representations.

Index Terms—GPS trajectory, multi-source data, driving style representation, generative adversarial networks, data augmentation

1 Introduction

The advances of global positioning system (GPS) and wireless communication techniques have enhanced the ability of various systems in collecting the spatio-temporal vehicular trajectories. The massive GPS trajectories stimulate a number of trajectory mining tasks for better understanding human mobility patterns and behaviors [65], among which characterizing human driver's driving behaviors is an important yet challenging task. Similar as the bio-metrics, it is believed that each driver also has a distinguishable pattern of driving, which is referred as *driving style* [34]. Specifically, driving style reflects a driver's fine-grained behavioral habits of steering and speed control and their temporal combinations [13]. Learning drivers' driving style representations from their trajectories can benefit many intelligent applications, e.g., driving assessment and assistance [55], driver-vehicle interaction [34], autonomous driving [31], and etc. In addition, auto insurance companies have been interested in utilizing the driving style information for risk assessments and personalized insurance pricing [16], [27]. For example, a good driving style representation can be used to identify the true driver of a trip for claim fraud detection [13].

In the literature, some valuable efforts have been made to derive the driving style representations. Traditional approaches heavily rely on the data collected from automobile sensors (e.g., controller area network buses) [16], [19], [42], [49] or dedicated sensors (e.g., high-definition cameras) [23], [25], [46] for driving style learning. However, it is difficult to retrieve data from automobile sensors while dedicated devices will incur installation costs. Recent studies [7], [13], [30], [66] turn to leverage deep learning models to process GPS trajectories for learning the driving style representations. Compared to automobile and dedicated sensors, GPS sensor data are often easier to access and thus are more popular in the large-scale study [13], [65]. These works, however, require high-frequency rate of GPS data collections, which may be prohibited due to concerns of privacy and energy consumption [36]. Furthermore, these works merely focus on the feature extractions from GPS data, but have overlooked the instant driving context information, such as road conditions and traffic conditions. As a result, they are inadequate to acquire accurate driving style representations.

Despite these research efforts, it is still non-trivial to efficiently learn driving style representation from GPS trajectories, mainly due to following challenges. First, practical GPS

- *Zhidan Liu, Junhong Zheng, and Jinye Lin are with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, Guangdong 518060, China. E-mail: liuzhidan@szu.edu.cn, {zhengjun4hong2019, linjinye2021}@email.szu.edu.cn.*
- *Liang Wang is with the School of Computer Science, Northwestern Polytechnical University, Xi'an, Shaanxi 710060, China. E-mail: liangwang@nwpu.edu.cn.*
- *Kaishun Wu is with The Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China, and also with College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, Guangdong 518060, China. E-mail: wu@szu.edu.cn.*

Manuscript received 19 January 2022; revised 18 August 2022; accepted 13 September 2022. Date of publication 21 September 2022; date of current version 3 November 2023.
This work was supported in part by the China NSFC under Grant 62172284, 61872248, and U2001207, in part by the Guangdong Basic and Applied Basic Research Foundation under Grants 2022A1515010155 and 2017A030312008, in part by Shenzhen Science and Technology Foundation under Grants ZDSYS20190902092853047 and R2020A045, in part by the Project of DEGP under Grants 2019KCXTD005 and 2021ZDZX1068 and in part by the Guangdong "Pearl River Talent Recruitment Program" under Grant 2019ZT08X603. (Corresponding author: Kaishun Wu.)
Digital Object Identifier no. 10.1109/TMC.2022.3208265

1536-1233 © 2022 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.
Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

trajectory data are usually collected at a low-sampling rate, e.g., 1 sample per 30 seconds [36], and are probably sparse, i.e., their quality may not meet the rigid requirements of some deep learning models [30]. Second, driving is essentially a complex activity and the resultant driving style will be influenced by many factors. The GPS trajectory data themselves cannot capture the complete view of a driver's driving style, and hence the external context information should be taken into account. However, how to properly integrate the features from GPS trajectory data and context information into one model needs to be well designed and thus is challenging. Third, some drivers, e.g., the newly joined drivers, may not have sufficient data, and their driving style representations, especially those learned by the deep learning models that require a large amount of training data, would be under-fitting, which cannot effectively support upper applications.

In this paper, we present an adversarial driving style representation learning approach, named Radar, which extracts comprehensive features from multi-source data and builds a semi-supervised generative adversarial networks (SGAN) based model to learn driving style representations from these extracted features. To better describe a driver's driving behaviors, Radar not only transforms raw GPS trajectory data to fine-grained statistic features about driver's habits of steering and speed control, but also additionally considers each GPS trajectory's contextual features, which are captured by three aspects of road condition, geographic semantic, and traffic condition. In particular, different from specific GPS locations, geographic semantic could encode high-level geographic features of a trajectory by mapping it to the whole city area. We aggregate the three kinds of contextual features as the driving context representation, which greatly governs a driver's driving activity and thus is important for accurately learning a driver's driving style. To tackle the data sparsity issue, Radar makes use of SGAN to construct the learning model, which equally treats statistic features and contextual features as the input to learn the driving style representations. Our learning model consists of three different components: generator, discriminator, and classifier, which work together to not only classify drivers from inputted trajectories but also generate fake samples close to the training data. As a result, Radar' learning model can achieve better generalization ability through both data augmentation and the competition between generator and discriminator. To help drivers with insufficient data learn accurate driving style representations, we propose a series of basic data augmentation strategies to enrich their training data. Furthermore, we exploit the novel concept of triplet loss [50] to constrain the representation learning process of drivers who have limited data through some well-selected auxiliary drivers' driving style representations. These strategies can effectively enhance the learning capability of Radar.

Our key contributions are summarized as follows:

- To the best of our knowledge, we are the first to consider the problem of context-aware driving style representation learning from practical GPS trajectory data, which improves existing works by comprehensively considering the driving contexts.
- We propose an adversarial driving style representation learning approach – Radar, which exploits multi-source data and a SGAN based learning model to efficiently learn driving style representations from GPS trajectory data.
- We present a series of data augmentation strategies to enrich a driver's training data. In addition, we propose an auxiliary driver based data augmentation strategy for the drivers with insufficient training data to effectively learn driving style representations.
- We conduct extensive experiments with two benchmark applications, namely *driver identification* and *driver number estimation*, based on a large real-world trajectory dataset. Experimental results demonstrate that Radar outperforms state-of-the-art approaches, e.g., on average improving the accuracy of driver number estimation and driver identification by 9.6% and 5.6%, respectively. The effectiveness of our data augmentation strategies has also been validated by extensive experiments.

The remainder of the paper is organized as follows. We review related works in Section 2. The problem statement is presented in Section 3. We elaborate the design of Radar in Section 4, and present the data augmentation strategies in Section 5. We conduct experiments to evaluate Radar in Section 6, and present the discussions in Section 7. Finally, Section 8 concludes this paper.

2 Related Work

The related works can be classified into three categories: *mobility pattern analysis*, *driving behavior analysis*, and *driving style learning*. We review these related works as follows.

2.1 Mobility Pattern Analysis

Nowadays, trajectory data can be collected by various devices and location-based services, e.g., GPS sensors [13], [30], cellular base stations [28], [51], and check-in Apps [62], [67]. Such trajectory data contain implicit information about people's movements within a city [65], and thus have inspired a wide range of applications, e.g., urban traffic estimation [39] and prediction [37], personalized recommender systems [22], [64], ridesharing [35], package delivery [9], anomaly detection [8], and social relationship inference [32], [33]. To enable those novel applications, a plethora of research works have been conducted to uncover human mobility patterns from trajectory data [6], [58], [65].

Among these studies, trajectory-user linking [66], which links trajectories to users who produce them, is quite relevant to our work. The high-level idea of solving this problem is to capture people's mobility patterns across different application domains by exploiting representation learning models. Instead of relying on some statistical models [28], most of recent works mainly analyze mobility trajectories by building various deep learning models to learn the semantic trajectory representations [17], [41], [47], [51], [62], [66], [67]. For example, Feng et al. present a deep learning framework to link heterogeneous mobility data, which are collected from different check-in services, to the users [17]. Ren et al. build a spatio-temporal Siamese network model to predict whether an incoming set of trajectories belong to a certain agent based on historical trajectory data [47]. In addition, Miao et al. utilize recurrent networks with attention mechanism to solve the

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

trajectory-user linking problem [41]. Song et al. present *HERMAS* to learn mobility representations from large-scale cellular signaling data [51]. Furthermore, Zhou et al. firstly introduce deep meta-learning to improve the generality of mobility prediction and classification models trained based on both labeled and unlabeled trajectories [67]. Different from those works, we aim to learn drivers' driving style representations from real-world GPS trajectory data, which involves of capturing more complicated human driving behaviors from noisy and sparse trajectory data.

2.2 Driving Behavior Analysis

Driving behavior analysis aims to study the relationship between the driver's behaviors (e.g., accelerating or steering) and current driving state (e.g., the vehicle's speed, driving direction, and operating reactions) [5], [7]. Extensive studies have been conducted by utilizing different data as the input. Previous works primarily rely on the explicit data collected from automobile sensors, e.g., on-board diagnostic (OBD) systems [14], [16], [27], controller area network (CAN) buses [19], [42], [44], [49], and dedicated devices [2], [15], [25], [63], to analyze drivers' driving behaviors. For example, Ezzini et al. utilize OBD data to realize driver identification and fingerprinting [14]. He et al. propose the *PPP* framework that fuses both OBD data and insurance data to profile driver behaviors and customize the insurance pricing model [27]. Based on the same types of data, Fang et al. present *MoCha* to model and predict a user's driving patterns in terms of three key metrics, i.e., distance, time, and speed, for the usage-based insurance [16]. Rich driving maneuver data can be derived from the CAN buses, and thus researches have built different deep learning models, e.g., long short-term memory [49] and convolutional recurrent neural network [42], to analyze driving behaviors. By deploying high-definition dash-cameras in the vehicles, video frames about the driving environment can be captured. Combined with other driving signals, a variety of computer vision techniques are proposed to not only recognize some basic driving actions, but also identify the driver's visual attention [2], [15] or even intention [25]. In particular, Zhang et al. study the multi-vehicle interaction patterns in the lane change scenarios by analyzing lane-change video frames [63]. However, it is relatively difficult to collect data from these automobile sensors, while dedicated devices like cameras bring installation costs. These constraints greatly limit their usability.

Compared to the in-vehicle sensors, GPS sensor data are much easier to collect, and thus GPS trajectory data based driving behavior analysis has attracted much attention [55], [60]. For example, Yang et al. analyze GPS traces of peer vehicles to proactively alter drivers of the vehicles with dangerous behaviors nearby [60]. By jointly modeling the peer and temporal dependencies of driving trajectories, Wang et al. enable the applications of driving score prediction and risk area detection [55]. Recently, many works [3], [4], [23], [46], [59], [61] resort to collect driving data using the internal sensors of smartphones and analyze such data for monitoring drivers' behaviors. These works mainly utilize sensing data collected by smartphone sensors (e.g., accelerometer, gyroscope, microphone, and speaker) to detect the driver's maneuver [4], [46], [61] and predict possible driving risks [3], [23], [59]. In particular, Chan et al. have summarized the recent smartphone sensing based driver behavior analysis works [5], and interested readers can refer to this survey. These works, however, mainly concern about driving safety, rather than learning a driver's latent driving style.

2.3 Driving Style Learning

Different from driving behavior analysis, the driving style learning generally aims to obtain the latent representation of a driver's fine-grained driving habits [34]. The learned driving style representation acts as the "driver DNA" [20], which can be used to support intelligent transport applications [31], [52], [55]. For example, Sun et al. combine drivers' driving style information to provide personalized estimated time of arrival for users, who may spend different time to travel the same route due to their diverse driving styles [52].

Earlier works usually build conventional machine learning models, e.g., rule based classifier [26] and random forest model [54], to learn driving style representations from CAN bus data, and make use of such representations to identify drivers for the given trajectories. Chowdhury et al. also build random forest models for driver identification based on the hand-crafted features extracted from GPS trajectories [10]. However, these works suffer from either the difficulty of retrieving CAN bus data or the limited representation ability of conventional machine learning models.

Due to the superior representation ability of deep learning techniques [37], recent works usually utilize various deep learning models to learn driving style representations from trajectory data [7], [12], [13], [30]. For example, Chen et al. propose a multi-task learning framework that consists of graph representation learning and semi-supervised learning to identify driving styles from GPS trajectory data in the temporal dimension [7]. This work, however, does not consider the problem of representation learning for drivers with limited historical data. In addition, Dong et al. propose an autoencoder regularized deep neural network and a trip encoding framework to learn drivers' driving styles directly from GPS trajectories [12], [13]. Besides, Kieu et al. propose a trajectory-to-image representation framework that encodes both geographic features and driving behaviors of trajectories into multi-channel images [30].

Although existing works on driving style learning could achieve remarkable performances, they are still not sufficiently efficient and practical. First, they require high-quality GPS trajectories that are collected at a high-sampling rate such as 1Hz, while most practical GPS trajectories are collected at a low frequency, e.g., 1 sample per 30 seconds, due to the concerns of energy consumption and privacy [36]. Second, they merely extract features from GPS data while overlooking the driving contexts, within which a trajectory has been generated. Third, they do not consider the *cold-start problem* for newly joined drivers, who have insufficient historical data to learn a good driving style representation. Different from prior researches, we utilize multi-source data to construct comprehensive driving contexts and exploit the advanced SGAN modeling [45] to learn more effective and accurate driving style representations. Furthermore, we improve the concept of triple loss [50] by selecting suitable auxiliary drivers to constrain the representation learning for drivers with limited training data.

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

Compared with the earlier version of Radar [38], we further improve the design by addressing the cold start problem. Specifically, we enhance Radar's driving style representation learning for drivers who own limited training data with some basic data augmentation strategies and a novel auxiliary driver-based data augmentation method. In addition, we review more related works and perform extra experiments to validate the effectiveness of the proposed data augmentation strategies and the enhanced design of Radar. We discuss the design choices and privacy protections of Radar, which may inspire future studies.

3 Problem Statement

In this section, we introduce some important definitions and notations, and then formally define the context-aware driving style representation learning problem.

3.1 Definitions and Notations

The GPS trajectory data are collected when a set of drivers $\mathbb{U} = \{u_1, \cdots, u_{|\mathbb{U}|}\}$ drive their vehicles, which have been equipped with GPS sensors, on a road network. The GPS trajectory set $\mathbb{T}_{u_i}$ generated by driver u_i implicitly encodes u_i's driving style. Accurately learning the driving style representation can benefit many potential applications, such as driving assessment and assistance [2], [55], driver-vehicle interaction [34], [63], autonomous driving [31], and so on.

Definition 1. *(**GPS trajectory**) Let $\mathcal{T}_j^i \in \mathbb{T}_{u_i}$ denotes the j-th trajectory generated by driver u_i. Specifically, $\mathcal{T}_j^i = [g_1, \cdots, g_{|\mathcal{T}_j^i|}]$ is a time-ordered sequence of GPS records, where each GPS record is a tuple $< ts, lat, lng, v, dir >$, indicating that u_i's vehicle located at latitude lat and longitude lng at time ts, with instant travel speed v in direction dir.*

Due to GPS localization errors, we have to map raw GPS locations to their actual locations on the roads through map matching techniques [43]. Therefore, a trajectory $\mathcal{T}_j$ [1] could be mapped to a travel route $\mathcal{R}_j$ on the road network $\mathbb{G}$.

Definition 2. *(**Road network**) A road network is modelled as graph $\mathbb{G} = \{\mathbb{V}, \mathbb{E}\}$, where $\mathbb{V}$ represents the set of road intersections and $\mathbb{E}$ represents the set of road segments in a city. In addition, each road segment has following attributes: ID of road segment, road type, number of lanes, and one-way indicator.*

Definition 3. *(**Travel route**) The travel route $\mathcal{R}_j$ for a GPS trajectory $\mathcal{T}_j$ is denoted by a sequence of road segments, i.e., $\mathcal{R}_j = [e_1, \cdots, e_{|\mathcal{R}_j|}]$, on road network $\mathbb{G}$, where $e_i \in \mathbb{E}$ is a road segment in route $\mathcal{R}_j$ and $|\mathcal{R}_j|$ is the number of all traveled road segments. Note that end point of e_i is the start point of e_{i+1}.*

3.2 Problem Statement

We define the problem we consider in this paper as follows.

Definition 4. *(**Context-aware driving style representation learning**) Given a set of GPS trajectory data generated by drivers in $\mathbb{U}$, we aim to learn driving style representations for drivers in $\mathbb{U}$ by exploiting both the trajectories and some necessary context information that govern the driving activity, so as to support intelligent applications like* driver identification *and* driver number estimation.

Different from previous works [7], [10], [12], [13], [30] that heavily rely on high-quality GPS trajectory data, we should devise an approach that works well for practical trajectory data and incorporates contextual information for much better driving style representation learning. To that end, we have to address the following challenges.

(1) *GPS data sparsity*. This challenge is raised from two aspects. On the one hand, in practice GPS data are usually collected at a low-sampling rate, e.g., 0.1Hz. On the other hand, trajectories are of different lengths and may contain deficient driving behavior information, resulting in insufficient qualified trajectories. These factors together lead to low-quality data for training deep learning models, and thus impair their performances.

(2) *Balanced integration of features from GPS data and contextual information*. Although driving contexts would benefit driving style representation learning, how to efficiently encode these contextual information and further gracefully integrate features extracted from raw GPS data and driving contexts should be wisely designed. The driving contexts involve various information, and the resultant feature vectors may be of different dimensions.

(3) *The cold-start problem for drivers with limited historical data*. Recent advances usually prefer to build deep learning models to learn driving style representations from massive data [5], [7], [12], [13], [30], while some drivers, especially the newly joined drivers, do not have sufficient trajectory data for the model training, resulting in the under-fitting representations. How to improve the driving style representation learning of such drivers is necessary yet challenging.

4 Design of Radar

In this section, we present the overview of Radar, and then describe the detailed design of each module.

4.1 Overview

Fig. 1 illustrates the architecture of our approach Radar, which consists of three major modules: *GPS data transformation*, *driving context representation*, and *learning model*. At high-level, Radar takes raw GPS trajectories and road map as the input, and exploits the modules of *GPS data transformation* and *driving context representation* to extract features from a GPS trajectory and the corresponding contexts. The integrated feature tensors are then fed into *learning model* to compute the driving style representation, which can support many intelligent applications, e.g., driver number estimation and driver identification.

Specifically, the *GPS data transformation* module utilizes a sliding window to calculate various statistics of the GPS data, which finally form the statistic feature matrix. For the *driving context representation* module, it firstly applies map matching technique to transform each GPS trajectory to an actual travel route. With this route, Radar derives context information from three aspects of road conditions, geographic semantic (i.e., geographical distribution of the route over the area of interest), and traffic conditions. These context information are fused to form a contextual feature

1. We omit the upper-script if the context is clear.

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

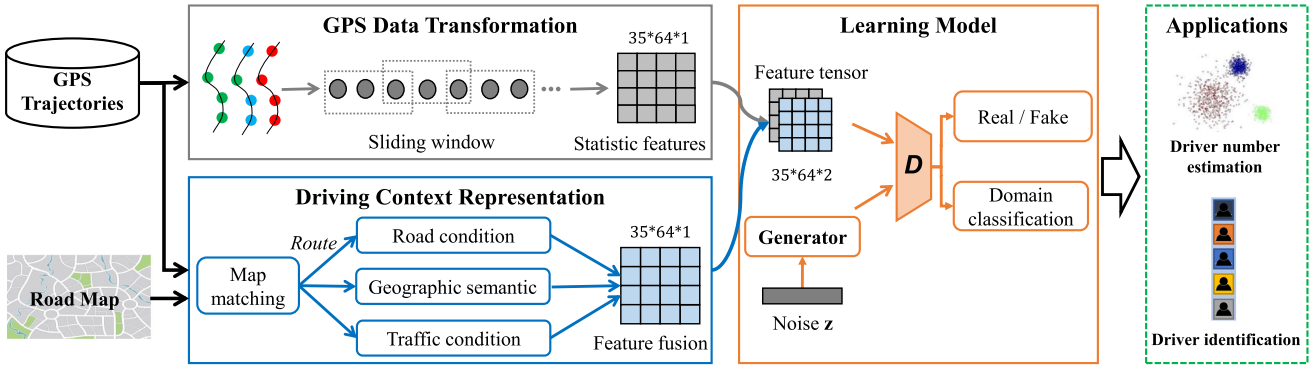


Fig. 1. The architecture of our approach Radar, which takes the input of GPS trajectory data and road map, and computes the driving style representations through three modules, i.e., *GPS data transformation*, *driving context representation*, and *learning model*. The final driving style representations learned by Radar can effectively support various intelligent applications like *driver number estimation* and *driver identification*.

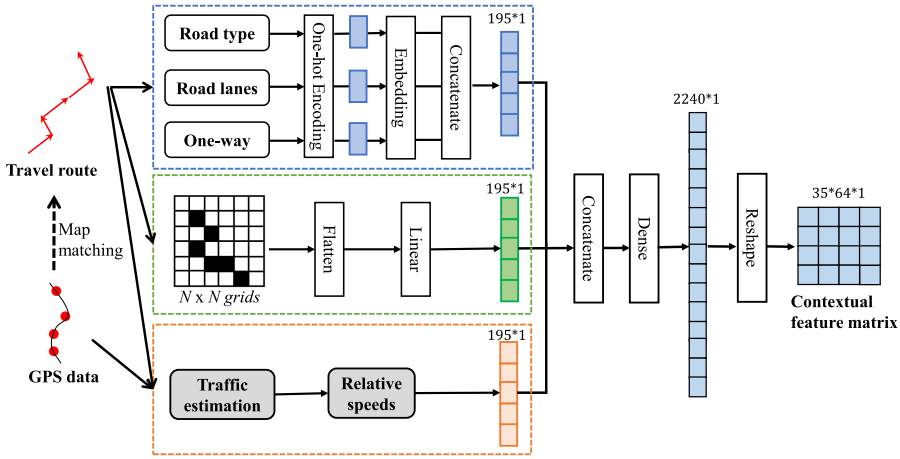


Fig. 2. The driving context representation module can capture driving contexts of road conditions, geographic semantic, and traffic conditions.

matrix. Lastly, both statistic feature matrix and contextual feature matrix are integrated as the input for the *learning model* module. In particular, we adopt the emerging semi-supervised generative adversarial network architecture [45] to construct our learning model for deriving effective and accurate driving style representations.

4.2 GPS Data Transformation

Instead of inputting raw GPS data to deep learning models, we will transform each GPS trajectory into more stable statistic features. Similar as previous work [12], [13], we divide a GPS trajectory into segments of a fixed length L_s, with a shift of $\frac{L_s}{2}$ to avoid much information loss between any two adjacent segments. We employ five basic features to capture the instantaneous vehicular movement features, namely *speed norm*, *difference of speed norm*, *acceleration norm*, *difference of acceleration norm*, and *angular speed*. To reduce the possible impact of outliers, we further divide a segment into frames of a fixed size L_f, with a shift $\frac{L_f}{2}$. For each frame, we calculate seven statistics for each basic feature, including mean, minimum, maximum, 25%, 50% and 75% quartiles, and standard deviation. For each trajectory $\mathcal{T}_j$ consisting of a sequence of time-ordered GPS records in the form of $g_i = < ts, lat, lng, v, dir >$, we can easily calculate speed statistics using travel speed v, acceleration statistics with location (lat, lng), and angular statistics with travel direction dir, respectively. As a result, we can derive a set of statistic feature matrices, each of which consists of $5 \times 7 = 35$ rows and $2 \times \lfloor \frac{L_s}{L_f} \rfloor$ columns. A statistic feature matrix encodes the driving behavior information of a trajectory segment, and serves as partial input to the learning model with its class label (i.e., the driver identifier) as the original trajectory $\mathcal{T}_j$.

In our implementation, we set $L_s = 195$ and $L_f = 6$ for the best performance. Therefore, we can obtain a set of statistic feature matrices of size 35×64 for each trajectory. In particular, if a trajectory segment is shorter than L_s, we will pad zeros into the matrix, so that to unify the size of all statistic feature matrices. In principle, long trajectories contain more information about the driving behaviors, and thus are more preferable for the model training.

4.3 Driving Context Representation

Since driving activities will be implicitly governed by the surrounding driving environment, thus Radar also takes driving context information into consideration to let machines deeply "understand" drivers' behaviors especially under certain circumstances. In the design of Radar, we particularly consider the three contexts of road conditions, geographic semantic, and traffic conditions.

Fig. 2 illustrates how Radar processes each raw GPS trajectory to generate the contextual features. For each GPS trajectory $\mathcal{T}_j$, we firstly recover the travel route $\mathcal{R}_j$ through map matching techniques [43]. Since GPS data transformation module outputs one statistic feature matrix for each

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

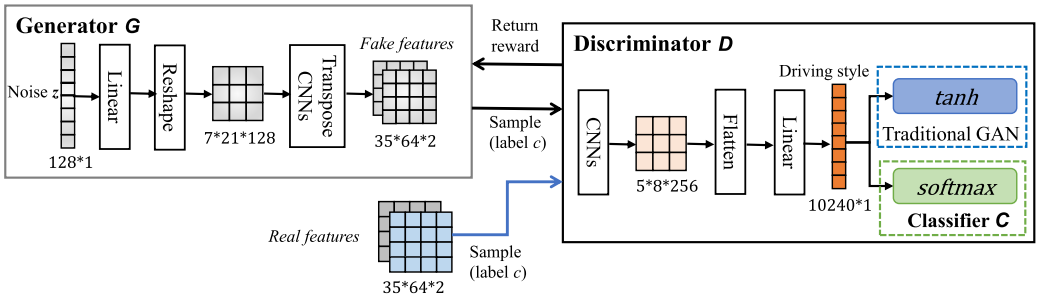


Fig. 3. The design of learning model, which consists of the generator $\mathcal{G}$, the discriminator $\mathcal{D}$, and the classifier $\mathcal{C}$.

trajectory segment, thus the driving context representation module will operate on trajectory segment and its associated travel route segment as well, and accordingly produces one contextual feature matrix. Based on road network $\mathbb{G}$, the s-th trajectory segment $\mathcal{T}_{js}$ and its travel route segment $\mathcal{R}_{js}$, we derive each context representation as follows.

Road Condition. We utilize static road attributes of road type, number of lanes, and one-way indicator to characterize road conditions. Let n_t, n_ℓ, and n_o to represent the numbers of possible values in the three types of categorical attributes, we thus employ three attribute vectors of length n_t, n_ℓ, and n_o, respectively, to encode the attributes of each road segment, respectively. Specifically, one-hot encoding is adopted to generate the attribute vectors. Given a travel route $\mathcal{R}_{js}$, we derive road type vectors of road segments covered by $\mathcal{R}_{js}$, and sequentially connect them into one vector, which describes the road types a vehicle had traveled when generating trajectory $\mathcal{T}_{js}$. In addition, we adopt an embedding layer to reduce the dimensionality of the sparse attribute vector. Similarly, we apply the same operations to the attributes of road lanes and one-way, and derive their attribute vectors for route $\mathcal{R}_{js}$, respectively. Finally, we concatenate the three embedding vectors into one vector of size 195×1, which represents the context of road conditions.

Geographic Semantic. The GPS data only reflect the instantaneous driving statuses, but not capture the high-level geographic semantic of a trajectory, e.g., origin, destination, and traveled regions. Thus, `Radar` maps each GPS trajectory segment to the whole city area to derive its geographic semantic representation, which is formally defined as follows.

Definition 5. *(**Geographic semantic representation**) We partition the city area into $N \times N$ grids. For each trajectory $\mathcal{T}_j$, we compute a geographic semantic representation matrix $\mathbf{M}_j$, where we set $\mathbf{M}_j[a, b] = 1$ if the travel route $\mathcal{R}_j$ of $\mathcal{T}_j$ intersects with the grid $[a, b]$; otherwise $\mathbf{M}_j[a, b] = 0$.*

As shown in Fig. 2, we further flatten the matrix $\mathbf{M}_j$ as a vector, which is fed into a linear layer for reducing the dimensionality. In our design, we set the final geographic semantic vector of size 195×1. Noting that we generate such a vector for each trajectory segment $\mathcal{T}_{js}$ as well.

Traffic Condition. In addition to road conditions, another factor that has great impact on driving activities is the real-time traffic condition. Considering both vehicle's instantaneous movements and surrounding traffic conditions can better define a driver's driving behaviors. Therefore, we use *relative speed*, which is calculated as the ratio between vehicle's travel speed and average travel speed of the vehicle's locating road segment, to represent traffic condition context.

To that end, we make use of all available GPS data to estimate the real-time traffic conditions. For each road segment, its traffic condition can be approximated as the average travel speed of all vehicles passing by within a time slot Δt. Therefore, we classify all GPS records to road segments according to their map matching results. For a given road segment, we calculate its average travel speed of a specific time slot using the GPS records falling into that time slot. Due to data sparsity, we may not derive a complete traffic conditions of the whole road network $\mathbb{G}$ over all time slots. For simplicity, we directly apply temporal-spatial interpolations to infer the traffic conditions of uncovered road segments by leveraging the inherent traffic correlations among roads. In fact, some advanced traffic estimation methods [36] can be adopted to compute the complete real-time traffic conditions. Once we obtain the traffic conditions of all road segments, we calculate relative speeds for road segments covered by travel route $\mathcal{R}_{js}$ of a trajectory segment $\mathcal{T}_{js}$. These relative speeds then form $\mathcal{T}_{js}$'s traffic condition representation.

As shown in Fig. 2, when the three representations of driving contexts are ready, `Radar` concatenates them into one vector, which is then fed into a dense layer to derive a vector of size 2240×1. To be compatible with the statistic feature matrix, we reshape it into a contextual feature matrix of size $35 \times 64 \times 1$.

4.4 Learning Model

To tackle the poor data quality issue, we employ generative adversarial networks (GAN) [24] to construct the learning model. Essentially, GAN operates by training two neural networks that play a *min-max* game: a *discriminator* is trained to discriminate real samples from fake ones, while a *generator* tries to generate fake training data to fool the discriminator. Therefore, GAN is able to generate samples very similar to the real GPS trajectory data and thus improves the generalization ability of the derived model.

In particular, we adopt the emerging semi-supervised GAN (*SGAN*) architecture [45] to build our learning model, which mainly consists of a generator $\mathcal{G}$ and a discriminator $\mathcal{D}$, as shown in Fig. 3. In SGAN, discriminator $\mathcal{D}$ can also act as a classifier $\mathcal{C}$ to classify each input sample into one of the predefined $(k+1)$ classes, where k is the number of classes and the additional class label is added for a new "*fake*" class. The competition and interaction (via *reward*) between generator and discriminator will improve the quality of resultant driving style representations. Therefore, our model can not only classify drivers according to the learned driving styles, but also for a given class c generates corresponding fake

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

driving style features, which are similar to training samples belonging to class c. To achieve this goal, the model training involves both traditional unsupervised GAN task and supervised classification task simultaneously. Training in unsupervised mode allows our model to learn useful feature extraction capabilities from unlabeled samples, whereas training in supervised mode allows the model to use extracted features and apply classifications. Therefore, SGAN can outperform traditional GANs on efficiently generating higher quality samples [11].

Discriminator $\mathcal{D}$ (and classifier $\mathcal{C}$). As shown in Fig. 3, discriminator takes either real samples, generated from GPS data and context information, or fake samples, produced by generator $\mathcal{G}$, as the input, which is further processed by a neural network to derive driving style representations. Discriminator $\mathcal{D}$ is trained in both unsupervised mode and supervised mode.

- *Unsupervised mode*. In this mode, discriminator $\mathcal{D}$, with parameter θ_d, predicts whether a sample is true (sampled from real trajectory data) or fake (generated by the generator $\mathcal{G}$) by calculating the probability score $\mathcal{D}(x|\theta_d)$ that the sample x is true. We train our learning model like traditional GANs by maximizing the score for real samples and minimizing it for fake ones. We achieve this objective by minimizing $\mathcal{L}^{(\mathcal{D})}$, which is defined as follows.

$$\mathcal{L}^{(\mathcal{D})} = -[\mathbb{E}_{x\sim p_r(x)}\log \mathcal{D}(x|\theta_d) + \mathbb{E}_{x\sim \mathcal{G}}\log\left(1-\mathcal{D}(x|\theta_d)\right)], \tag{1}$$

  where $p_r(x)$ represents the distribution of real samples from trajectory data.
- *Supervised mode*. In this mode, discriminator $\mathcal{D}$ acts as classifier $\mathcal{C}$ to complete a multi-class classification problem. For each sample, classifier $\mathcal{C}$, with parameter θ_c, predicts if the sample belongs to one of the predefined $(k+1)$ classes. Because the label of driving style features generated by the generator $\mathcal{G}$ is known, classifier $\mathcal{C}$ can also utilize the labels of fake samples for training. Thus the generalization ability of the model could be improved. In addition, classifier $\mathcal{C}$'s classification on both real and fake samples can be used as feedback (via reward) to improve generator $\mathcal{G}$, i.e., higher classification accuracy will bring more returns. To train the classifier $\mathcal{C}$, we aim to minimize the classifier loss $\mathcal{L}^{(\mathcal{C})}$, i.e., the cross entropy loss on true labeled samples that is computed using the overall classifier score.

$$\mathcal{L}^{(\mathcal{C})} = -\mathbb{E}_{p(x_c,c)}[\log \mathcal{C}(c|x_c,\theta_c)], \tag{2}$$

  where x is a sample of class c, and $\mathcal{C}$ should correctly classify it as class c.

We implement above two modes in one unified framework, as shown in Fig. 3. Discriminator $\mathcal{D}$ and classifier $\mathcal{C}$ share the same feature extraction layers, but have different output layers. Specifically, we use a stack of convolution layers with LeakyReLu to process each input sample. After a series of convolutions, we get a feature tensor that is flatten and inputted to a dense layer to derive the driving style representation vector. For traditional GAN task, the vector is fed into `tanh` to discriminate real samples and fake ones. For classifier $\mathcal{C}$, the vector is fed into `softmax` to obtain classification probabilities of the $(k+1)$ classes.

Generator $\mathcal{G}$. Given the distribution $p_r(x)$ of real samples and k class labels from real training data, generator $\mathcal{G}$ aims to find the parameterized conditional distribution $\mathcal{G}(\mathbf{z}, c, \theta_g)$ that is close to the real distribution $p_r(x)$. The generated fake samples are conditioned on the network parameters θ_g, noise vector $\mathbf{z}$, and class label c, which are sampled from prior distribution p_z and p_c, respectively. Label c of a fake sample y can be known when the generator $\mathcal{G}$ generates y, so that the actual classification label of each generated sample is retained for training classifier $\mathcal{C}$. Following the feature matching technique proposed to addresses the instability of GANs [48], we train $\mathcal{G}$ by minimizing loss $\mathcal{L}^{(\mathcal{G})}$ expressed as:

$$\mathcal{L}^{(\mathcal{G})} = ||\mathbb{E}_{x\sim p_r(x)}\mathbf{f}(x,\theta_f) - \mathbb{E}_{\mathbf{z}\sim p_z}\mathbf{f}(\mathcal{G}(\mathbf{z},c,\theta_g),\theta_f)||_2^2, \tag{3}$$

where $\mathbf{f}(\cdot)$ denotes activation on an intermediate layer (e.g., the stack of convolution layers) of discriminator $\mathcal{D}$, θ_f is the parameter subset of θ_d corresponding to the intermediate layer of discriminator $\mathcal{D}$, and c is the class label of real sample x. The objective of generator training is thus to minimize the discrepancy between real and generated data distributions in the feature space.

As shown in Fig. 3, generator $\mathcal{G}$ is implemented with four deconvolution layers, which transform noise vector $\mathbf{z}$ into fake driving style features. In particular, each deconvolution layer is followed by a nonlinear activation based on batch normalization and rectified linear unit (ReLU). $\mathbf{z}$ is a 128 dimensional vector sampled from a uniform distribution p_z, and it is processed by the dense and reshape layers before inputting to the deconvolution layers. Finally, generator $\mathcal{G}$ outputs a $35 \times 64 \times 2$ feature tensor as the same size of real feature tensors. The values of tensor items are shapely squashed within $[-1, 1]$ through the `tanh` function.

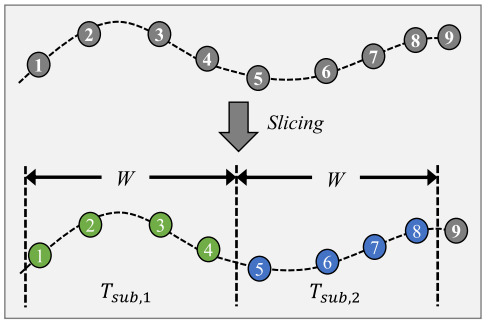
5 Data Augmentation Strategies

The superior performance of deep learning models, including GANs, heavily rely on a large number of training data to avoid over-fitting [56]. However, some drivers, especially the newly joined drivers, usually have insufficient historical trajectory data, and as a result `Radar` cannot effectively learn their driving style representations. To tackle this cold-start problem, data augmentation is a crucial technique to increase the amount of training data for such drivers. In this section, we propose a series of data augmentation strategies, which include four basic strategies adapted from time series data augmentation methods [29] and a novel auxiliary driver based data augmentation strategy.

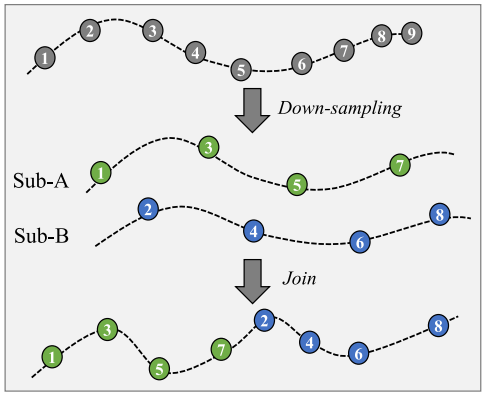
5.1 Basic Data Augmentation

GPS trajectory data are time series data as well, and thus existing time series data augmentation methods can be easily adapted to enhance the size of training data for driving style representation learning. Basic time series data augmentation methods directly manipulate the original GPS trajectory $\mathcal{T}$ by slightly modifying $\mathcal{T}$ in either time domain or frequency domain [29], [56]. Given an input trajectory $\mathcal{T} = [g_1, \cdots, g_{|\mathcal{T}|}]$ and its map matched travel route $\mathcal{R}$, the

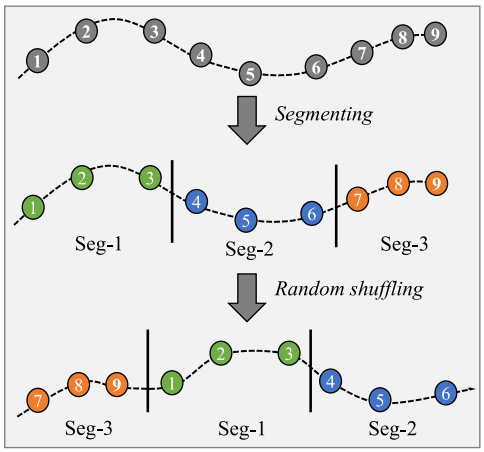
Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.



(a) Slicing



(b) Down-sampling



(c) Permutation

Fig. 4. Illustration of three basic data augmentation strategies.

label (i.e., the map matched road segment) of any GPS record in $\mathcal{T}$ is stored in $\mathcal{R}$. Then we explain each basic data augmentation strategy in detail as follows.

1) *Slicing*. The general idea of slicing is that the data can be augmented by slicing the time stamps of GPS records off the ends of input trajectory $\mathcal{T}$. Generally, a sub-trajectory $\mathcal{T}_{sub}$ sliced from input $\mathcal{T}$ with a window of size W can be represented as:

$$\mathcal{T}_{sub} = [g_i, g_{i+1}, \cdots, g_{i+W}], \tag{4}$$

where i is a random integer such that $1 \leq i \leq |\mathcal{T}| - W$. Slicing can produce multiple shorter trajectories from a long input trajectory, and increases the amount of training data. Fig. 4a illustrates the slicing based data augmentation, where a long trajectory is sliced into two shorter trajectories with a window of size $W = 4$.

2) *Down-sampling*. We down-sample the original GPS records of input trajectory $\mathcal{T}$ with a specific down-sampling rate τ to obtain a shorter GPS record series of length $\lfloor \frac{|\mathcal{T}|}{\tau} \rfloor$. The label series is also down-sampled at the same rate κ as the GPS record series. We do such down-sampling operations for τ times, and thus will derive τ shorter trajectories. Then we copy and join τ of thse shorter trajectories to form a long and new trajectory of the same length as the input $\mathcal{T}$. However, this strategy may not work well, because the join points usually have large jumps and the resultant trajectory breaks the temporal dependence of GPS records. Fig. 4b shows an example of down-sampling based data augmentation on the original trajectory.

3) *Permutation*. The GPS records of input trajectory $\mathcal{T}$ are partitioned into μ segments of equal length $\lfloor \frac{|\mathcal{T}|}{\mu} \rfloor$, and these segments can be permuted to produce new patterns. The label for each GPS record will be retained accordingly. A variant of permutation is random shuffling that rearranges individual GPS records rather than the segments. Although permutation can produce many synthetic trajectories from one single input, it cannot preserve the temporal dependency within original trajectory $\mathcal{T}$ any more. Fig. 4c illustrates the permutation based data augmentation, where the original trajectory is partitioned into three segments and these segments are randomly arranged into a new trajectory.

4) *Perturbations in frequency domain*. Different from above strategies that work in time domain, Gao et al. [21] recently propose to investigate data augmentation from frequency domain by utilizing perturbations in both amplitude spectrum and phase spectrum. Specifically, given an input time series $x_1, x_2, \cdots, x_m$, its frequency spectrum $F(\omega_i)$ can be calculated via Fourier transform as:

$$F(\omega_i) = \frac{1}{m} \sum_{t=1}^{m-1} x_t e^{-\mathrm{J}\omega_i t} = A(\omega_i) \exp[\mathrm{J}\theta(\omega_i)], \tag{5}$$

where J is the imaginary unit of Fourier transform, $\omega_i = \frac{2\pi i}{m}$, $i = 0, 1, \cdots, m-1$, is the angular frequency, $A(\omega_i)$ and $\theta(\omega_i)$ are amplitude spectrum and phase spectrum, respectively. As a result, random perturbations can be applied to $A(\omega_i)$ and $\theta(\omega_i)$ separately for data augmentation in the frequency domain. Specifically, for perturbations in $A(\omega_i)$, the amplitudes of randomly selected GPS records are replaced with Gaussian noises, which are sampled from a distribution considering original mean and variance in the input $\mathcal{T}$'s amplitude spectrum. For perturbations in $\theta(\omega_i)$, the phases of randomly selected GPS records are added by extra zero-mean Gaussian noses in the phase spectrum.

In our implementation, we set $W = 195$ for the slicing strategy, $\tau = 5$ for the down-sampling strategy, $\mu = 10$ for the permutation strategy, and $m = 195$ for the strategy of perturbations in frequency domain. These settings are tuned based on extensive experiments on real trajectory data.

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

5.2 Auxiliary Driver Based Data Augmentation

The basic data augmentation strategies enrich training data by manipulating existing trajectories of a driver, however, the derived data are still far from adequacy. In addition, the augmented data cover only partial input space (e.g., existing trajectories of a driver only cover certain driving contexts), while other part of the input space should also be explored to better characterize a driver's driving style under different driving contexts.

To this end, we propose an auxiliary driver based data augmentation strategy, which indirectly enhances the representation learning for the drivers with limited training data. The key idea is that for a given driver u, we select some auxiliary drivers and exploit the concept of triplet loss [50] to constrain u's driving style representation learning with these auxiliary drivers' driving style representations. Next, we will elaborate how to select auxiliary drivers and then make use of their driving style representations to enhance the learning process of drivers with insufficient data.

5.2.1 Auxiliary Driver Selection

Triplet loss is initially proposed to derive a unified embedding for face recognition [50], and has been widely used as a novel loss function for deep learning models, where the distance between the anchor sample and a positive sample is minimized while the distance from the anchor sample to a negative sample is maximized. The three samples of anchor, positive, and negative form a *triplet*. In our problem, we treat driver u_a with insufficient training data as the anchor, and both positive drivers and negative drivers as the auxiliary drivers to help u_a's driving style representation learning. Specifically, with respect to driver u_a, a positive driver u_p should share similar driving style as u_a, while a negative driver u_n will have a distinct driving style from u_a. It is worthy to note that both positive and negative drivers should have sufficient historical trajectory data for their own driving style representation learning.

Due to insufficient training data, however, we cannot learn a reliable representation for driver u_a. As a result, we have to seek an alternative indicator to measure the driving style similarity between drivers. The average driving speed has been frequently used as an important metric to reflect one's personalized driving style [16], [52], we thus make use of this metric to distinguish positive and negative drivers for a given anchor driver.

Specifically, for each driver u we calculate an average driving speed vector $\mathbf{v}_u$ in an offline manner. Each item v_u^i of $\mathbf{v}_u$ indicates the average driving speed on the i-th ($1 \leq i \leq n_t$) road type, where n_t is the total number of road types in the road network. Based on driver u's historical trajectory data and the corresponding map matched travel routes, we can easily obtain the sum L_u^i of travel distance and the sum T_u^i of travel time on roads of the i-th road type. Thus, v_u^i is calculated as $\frac{L_u^i}{T_u^i}$. If a driver has no data on the i-th road type, we use the average of all other drivers' driving speed on such roads as a placeholder. We will replace this value with real average driving speed, once the driver has data on roads of the i-th road type.

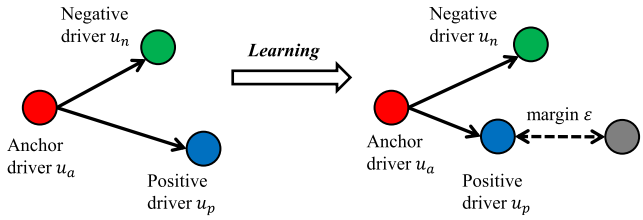


Fig. 5. Illustration of triplet loss, which aims to minimize the distance between an anchor driver and a positive driver, which share similar driving styles, while maximizing the distance between an anchor driver and a negative driver, which have distinct driving styles.

For the given anchor driver u_a, we randomly select two other drivers, and assign one driver as the positive driver u_p while the other as the negative driver u_n. The three drivers should satisfy the following constraint:

$$||\mathbf{v}_{u_a} - \mathbf{v}_{u_p}||_2^2 + \varepsilon \leq ||\mathbf{v}_{u_a} - \mathbf{v}_{u_n}||_2^2, \tag{6}$$

where ε is a pre-defined margin that is enforced between positive and negative drivers [50]. Equation (6) means that on the metric of average driving speed, anchor driver u_a is closer to positive driver u_p than to negative driver u_n. Fig. 5 further illustrates the idea of applying triplet loss for trajectory data augmentation.

5.2.2 Triplet Loss Improved Representation Learning

We assume that drivers with similar driving styles should have closer representations as well. Therefore, driving style representations of driver u_a, u_p, and u_n should also satisfy a distance constraint as follows:

$$||f_{u_a} - f_{u_p}||_2^2 + \alpha \leq ||f_{u_a} - f_{u_n}||_2^2, \tag{7}$$

where α is a hyper-parameter serving as the margin, while f_{u_a}, f_{u_p}, and f_{u_n} represent driving style representations that are learned from trajectory samples of driver u_a, u_p, and u_n, respectively. Thus, we make use of triplet loss [50] to define an auxiliary loss $\mathcal{L}_{aux}$ to further constrain the representation learning of driver u_a, i.e.,:

$$\mathcal{L}_{aux} = \frac{1}{M}\sum_{i=0}^{M}\left[||f_{u_a}^i - f_{u_p}^i||_2^2 - ||f_{u_a}^i - f_{u_n}^i||_2^2 + \alpha\right]_+, \tag{8}$$

where the operator $[x]_+ = \max(0, x)$, and M is the total number of possible triplets in the training dataset.

Therefore, the driving style representation learning process of driver u_a with insufficient training data is not only guided by the general loss $\mathcal{L}_{gen}$ that is defined in Equation (9), but also by the auxiliary loss $\mathcal{L}_{aux}$.

$$\mathcal{L}_{gen} = \mathcal{L}^{(\mathcal{D})} + \mathcal{L}^{(\mathcal{C})} + \mathcal{L}^{(\mathcal{G})} \tag{9}$$

On the one hand, the general loss $\mathcal{L}_{gen}$ ensembles the losses defined in Equations (1)$\sim$(3), and controls the learning from driver u_a's own data. On the other hand, the auxiliary loss $\mathcal{L}_{aux}$ makes driver u_a to be closer to positive drivers while to be far from negative drivers in the embedding space.

Fig. 6 illustrates the complete workflow of auxiliary driver based data augmentation for drivers with insufficient

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

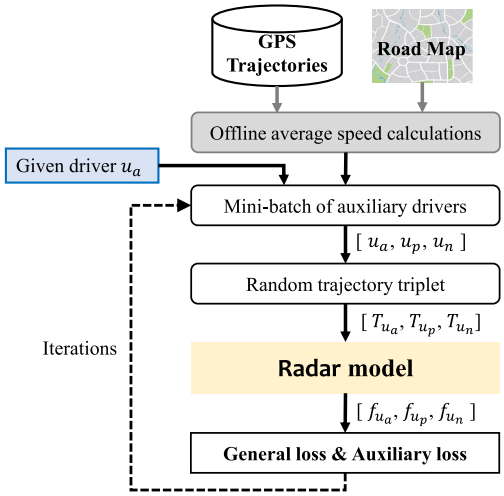


Fig. 6. The workflow of auxiliary driver based data augmentation.

training data. In practice, we compute the average driving speed vectors for all drivers based on their historical trajectory data in an offline manner before the model training. During the model training, for a given driver u_a, we firstly build a mini-batch of auxiliary drivers, which include both positive drivers and negative drivers. Instead of selecting driver triples from the whole dataset that may be computationally infeasible, mini-batch of auxiliary drivers can accelerate the training. Within the driver batch, we randomly generate a driver triplet $[u_a, u_p, u_n]$ that should satisfy the distance constraint in Equation (6). A driver triplet that does not meet the condition will be discarded. Next, we randomly extract one trajectory from the three drivers' historical trajectory datasets, respectively, to construct a trajectory triplet $[\mathcal{T}_{u_a}, \mathcal{T}_{u_p}, \mathcal{T}_{u_n}]$. We utilize the `Radar` model to learn driving style representation f_{u_a}, f_{u_p}, and f_{u_n} for anchor driver u_a, positive driver u_p and negative driver u_n based on their trajectory data, respectively. Based on the three representations, we calculate auxiliary loss $\mathcal{L}_{aux}$ and general loss $\mathcal{L}_{gen}$ using Equations (8) and (9), respectively. Furthermore, we use a hyper-parameter β, $0 \leq \beta \leq 1$, to balance the trade-off between $\mathcal{L}_{gen}$ and $\mathcal{L}_{aux}$. Therefore, the objective function for driver u_a is defined as:

$$\mathcal{L} = \beta \cdot \mathcal{L}_{gen} + (1 - \beta) \cdot \mathcal{L}_{aux}. \tag{10}$$

With the help of auxiliary drivers, the model parameters for drivers with insufficient training data can be optimized under the comprehensive loss in Equation (10). In practice, we can integrate some basic data augmentation strategies with the auxiliary driver based data augmentation strategy to achieve better representation learning performance. In our implementation, we set the margin $\varepsilon = 5$ and $\alpha = 0.1$, build a random driver batch of size 100, and set $\beta = 0.8$ to balance $\mathcal{L}_{gen}$ and $\mathcal{L}_{aux}$.

6 Performance Evaluation

In this section, we conduct extensive experiments to evaluate the effectiveness of `Radar` on learning accurate driving style representations with two applications.

TABLE 1
Distribution of Road Types Covered by GPS Trajectory Data

Road Type	*primary*	*secondary*	*tertiary*	*residential*	*service*
Percent	24.51%	12.55%	6.85%	27.54%	28.55%

6.1 Experimental Setup

Dataset. We use a large real-wold anonymized GPS trajectory dataset[2] for the experiments. This dataset contains 1.3 billions GPS records that are collected from 10000 drivers in Shanghai city, China, during a six-month period in 2015. These GPS records are collected at a low-sampling rate as 0.1Hz (i.e., one sample per ten seconds). Each GPS record includes the driver identifier, a timestamp, the location with longitude and latitude, travel speed, and travel direction.

Furthermore, we download the road network of the city area covered by GPS records from OpenStreetMap (OSM)[3], and model it as a graph $\mathbb{G}(\mathbb{V}, \mathbb{E})$, which has 159386 vertices and 30336 edges (i.e., road segments) in total. In addition, we obtain road attributes from OSM as well. After map matching, we derive the actual travel route for each trajectory. By analyzing all travel routes and the road attributes specified by the OSM file [1], we find that the trajectory data mainly cover five kinds of roads, i.e., primary, secondary, tertiary, residential and service roads. These roads are paved with asphalt or concrete. Table 1 shows the distribution of road types covered by the drivers' travel routes. We find that most of the GPS trajectories were generated within urban area, as ride-hailing services are more popular there. In addition, we analyze the driving activities of all drivers, and summarize the statistic results in Fig. 7. The statistics show that 80% drivers have completed 500 trajectories with data collection duration of 550 hours. On average, we have 430 trajectories for each driver in the dataset.

Baseline Approaches. We compare `Radar` with the following baseline approaches, which can also learn driving style representations from GPS trajectories.

- *ARNet*is one of state-of-the-art approaches. *ARNet*proposes an autoencoder regularized neural network for the driving style representation learning, merely from raw GPS data [13].
- *T2INET*is one of state-of-the-art approaches as well. *T2INET*represents one GPS trajectory as the multi-channel images that capture both geographic and driving behavior features using a sequence of convolution layers [30].
- `Radar-C`serves as one variant of our `Radar` by disabling the driving context representation module. As a result, `Radar-C`only takes the statistic features extracted from GPS records as input for the learning model to compute driving style representations.

2. The dataset is provided by a ride-hailing company that manages a fleet of vehicles to provide on-demand mobility services. With the drivers' consent, all vehicles' location and status information were transmitted back to a centralized platform for the safety and management purposes. To protect drivers' privacy, the trajectory data have been anonymized, where each driver is randomly assigned with an ID and each driver's trajectories are identified by this unique ID.

3. OpenStreetMap: https://www.openstreetmap.org/

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

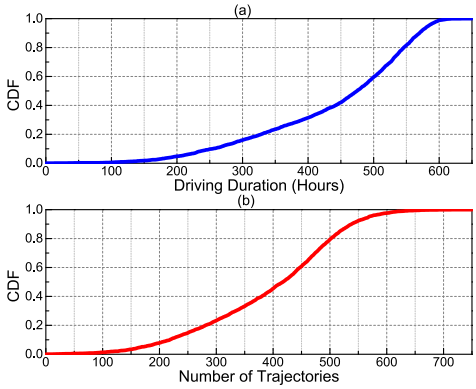


Fig. 7. Statistics about the drivers' driving activities.

Implementation. We implement Radar and all baseline approaches in Python 3.7.3 with Keras[4] 2.3.1 and TensorFlow[5] 2.2.0 for building various machine/deep learning models. In addition, we set Radar's parameters as follows. We set $L_s = 195$ and $L_f = 6$ for GPS data transformation. The city area is partitioned into 80×80 grids for geographic semantic representation. In graph $\mathbb{G}$, we have $n_t = 5$ road types, maximum number of lanes $n_\ell = 6$, and $n_o = 2$ for indicating one-way or not. We estimate traffic conditions with time slot $\Delta t = 30$ minutes. For the learning model, we use Adadelta as the optimizer, and set learning rates for generator $\mathcal{G}$ and discriminator $\mathcal{D}$ as 0.0001 and 0.0004, respectively. We set batch size as 128 and the epochs as 5000. Besides, we directly adopt the implementations of *ARNet* [13] and *T2INET* [30], which are provided by the authors respectively, and tune their parameters with our data to achieve their best performances.

We evaluate these approaches with two benchmark applications, i.e., *driver identification* and *driver number estimation*, on a server, which is equipped with Intel Core i9-9900K CPU@3.60GHz, NAVIDA GeForce RTX 2080 Ti GPU, and 32GB memory.

6.2 Driver Identification

The driver identification problem aims to identify the driver of a given unlabeled trajectory, which belongs to the supervised multi-class classification problem.

6.2.1 Extra Experimental Settings

In addition to the basic experimental setup, we have extra settings for the application of driver identification.

Training and Testing. In each experiment, we randomly select 10 drivers and use their GPS trajectories for the model training and testing. Specifically, 70% of the trajectory data are used for training, 10% for validation, and the remaining 20% for testing. The models of all approaches are trained with labeled trajectories, and for a testing trajectory the models should predict its driver identifier.

Performance Metric. We employ the top-n accuracy (denoted by $acc@n$) to evaluate the prediction performances of all approaches. In particular, $acc@n$ is calculated as the percentage of testing trajectories for which the ground truth drivers are in the top n predictions. For a testing trajectory, we rank the predicted driver identifiers in the descending order of probability values.

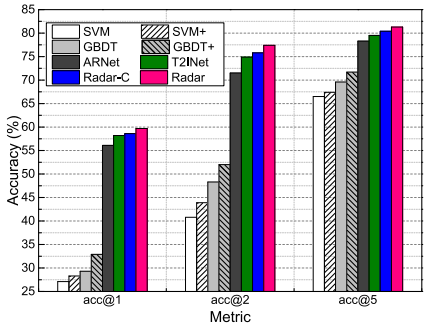


Fig. 8. Performance comparisons on Top-n accuracy with the long trajectories.

6.2.2 Experimental Results

Overall Results. In addition to the aforementioned three baselines, we also include two typical supervised learning models, i.e., support vector machines (SVM) [57] and gradient boosting decision trees (GBDT) [40], for performance comparisons. Specifically, SVM and GBDT take the statistic features produced by Radar as input for the predictions, while SVM+ and GBDT+ make use of both statistic and contextual features generated by Radar for the predictions. Furthermore, we partition drivers' trajectories into two sets: *long trajectories* (with duration more than 1950 seconds) and *short trajectories* (with duration less than 1950 seconds). We conduct experiments on each set of trajectories separately, and present the results in Figs. 8 and 9, respectively.

As shown in Fig. 8, the top-n accuracy of each approach becomes higher when n increases. Our approach achieves the highest $acc@5$ accuracy as 81.3%. These deep learning models, i.e., *ARNet*, *T2INET*, Radar-Cand Radar, always have better predictions than traditional supervised learning models, i.e., SVM and GBDT and their variants, with the largest performance gap as 36.6% on $acc@2$. It implies that deep learning models are indeed powerful at representation learning. By comparing the performances of traditional models, we find that SVM+/GBDT+ outperform SVM/GBDT, e.g., with $acc@1$ accuracy improvement by 1.2% and 3.6%, respectively. Therefore, it is necessary to include contextual features for better modeling. Compared to state-of-the-art *ARNet*and *T2INET*, Radar achieves more accurate predictions, e.g., averagely improving them by 2.6%, 4.2%, and 2.4% for $acc@1$, $acc@2$, and $acc@5$, respectively.

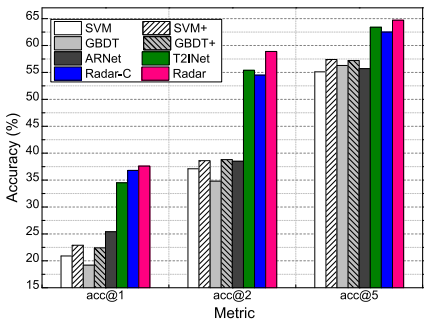


Fig. 9. Performance comparisons on Top-n accuracy with the short trajectories.

4. Keras: https://keras.io/
5. TensorFlow: https://www.tensorflow.org/

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

TABLE 2
Performance Comparisons on $Acc@1$ for Basic Data Augmentation Strategies With Different Amount of Training Data, Where *None* Means No Data Augmentation Strategy and *PFD* Represents the Strategy of Perturbations in Frequency Domain

Strategy	One week	Two weeks	One month
None	15.6%	17.8%	22.5%
Slicing	12.5%	18.5%	20.3%
Down-sampling	15.8%	17.6%	23.5%
Permutation	15.3%	20.3%	21.7%
PFD	**17.8%**	**21.6%**	**25.3%**

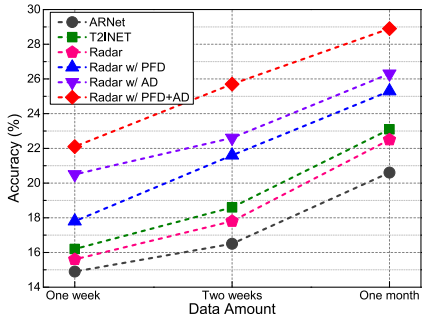


Fig. 10. Performance comparisons on $acc@1$ for various data augmentation strategies.

The prediction results on the short trajectory set are plotted in Fig. 9. Since short trajectories contain less information, and thus the prediction performances of all approaches have been seriously deteriorated. However, we find that the performance gap between *ARNet*/*T2INET*and our approach becomes even larger, i.e., on average Radar improves the two advanced approaches by 7.1%, 12.0%, and 5.2% for $acc@1$, $acc@2$, and $acc@5$, respectively. These comparisons reflect that Radar is able to extract more useful and accurate features from low-quality trajectory data, and thus can still achieve reasonably high prediction accuracy.

Effectiveness of Data Augmentation. We conduct experiments to investigate various data augmentation strategies by varying the amount of available training data for testing drivers as *one week data*, *two week data*, and *one month data*.

First, we combine Radar with different data augmentation strategies, i.e., no data augmentation and the four basic data augmentation strategies presented in Section 5.1, to learn driving styles for the drivers with insufficient training data. For simplicity, we only present the results on the metric of $acc@1$ in Table 2. When we increase the data amount, the accuracy of each combination increases as more training data generally benefit the representation learning accuracy. However, the performances of time domain based data augmentation strategies, i.e., *slicing*, *down-sampling*, and *permutation*, are not stable, since their accuracy results change around the ones of Radar with no data augmentation (i.e., *None* in Table 2). This is because these strategies manipulate existing trajectories in the temporal dimension, and inevitably break the original temporal dependency among GPS records. In contrast, we see that the strategy of perturbations in frequency domain (*PFD* for short) has the best accuracy in the three scenarios. The PFD strategy attempts to maintain the temporal dependency of trajectory data, while varying the GPS records from both amplitude spectrum and phase spectrum to produce synthetic data.

Then, we evaluate our novel auxiliary driver based data augmentation (*AD* for short) strategy, and present the results in Fig. 10. In this experiment, auxiliary drivers have been assigned with sufficient training data. From Fig. 10, we see that Radar with *AD* strategy improves the variant of Radar with *PFD* strategy, with an average improvement by 1.6%. Both variants outperform Radar without data augmentation, which implies that effective data augmentation strategies indeed improve model accuracy. More importantly, we find the two strategies, *PFD* and *AD*, are compatible, and can be adopted together to greatly improve the representation learning of drivers with limited data. According to these results, we find that the variant of Radar integrated with two strategies significantly outperforms state-of-the-art approaches, e.g., averagely improving *ARNet*and *T2INET*by 8.2% and 6.3%, respectively.

6.3 Driver Number Estimation

This application aims to estimate the number of drivers from a set of anonymous trajectories. To solve this problem, we train the driving style representation learning models with a set of labeled trajectories (i.e., with known driver identifiers), and exploit trained models to represent each testing trajectory as a driving style representation vector. Then, we employ the affinity propagation clustering algorithm [18] to classify all representation vectors into clusters. In theory, a desired model should effectively learn drivers' driving styles, and would classify the testing trajectories generated by a specific driver into the same cluster. Finally, the number of clusters is regarded as the number of drivers.

6.3.1 Extra Experimental Settings

In addition to the basic experimental setup, we have extra settings for the application of driver number estimation.

Training and Testing. We randomly select 10 drivers from the driver set $\mathbb{U}$ and take their labeled trajectories as training data. In addition, we randomly select κ drivers from remaining drivers, who are absent in the training data, to form a group, denoted by *Group* κ. We vary κ from 1 to 10. For each group, we randomly sample 50 trajectories from the κ drivers, and use these trajectories as testing data. We repeat 10 runs for each κ value, and report average results.

Performance Metrics. We compare different approaches on the following two performance metrics: (1) the mean absolute error (*MAE*), which is the difference between the ground truth of driver number and the estimation; (2) the adjusted mutual information score (*AMI*) [53] that measures the clustering quality. The AMI values fall in the range of [0,1], and larger AMI values are preferable.

6.3.2 Experimental Results

Overall Results. Table 3 shows the MAE results and deviations, where the best result of each group is marked in bold. When κ increases, the driver number estimation problem becomes harder, and thus the MAE will be larger. Among all experiments, we see that our approach (Radar and Radar-C) wins 7 best results (i.e., the smallest MAE) out of

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

TABLE 3
Performance Comparisons on MAE for Driver Number Estimation

Group κ	*ARNet*	*T2INET*	Radar-C	Radar
1	**0.64** $\pm$ 0.60	0.70 $\pm$ 0.68	0.80 $\pm$ 0.64	0.78 $\pm$ 0.65
2	**0.82** $\pm$ 0.80	0.88 $\pm$ 0.74	0.92 $\pm$ 1.20	0.84 $\pm$ 0.97
3	1.08 $\pm$ 1.26	1.22 $\pm$ 1.48	1.02 $\pm$ 1.24	**0.98** $\pm$ 1.24
4	1.18 $\pm$ 1.40	1.04 $\pm$ 1.46	**0.92** $\pm$ 1.02	1.02 $\pm$ 1.07
5	0.98 $\pm$ 1.24	**0.88** $\pm$ 1.56	1.20 $\pm$ 0.90	1.02 $\pm$ 0.88
6	1.24 $\pm$ 0.98	1.24 $\pm$ 0.96	**1.04** $\pm$ 1.24	**1.04** $\pm$ 1.06
7	1.60 $\pm$ 1.24	1.42 $\pm$ 1.64	1.42 $\pm$ 1.44	**1.24** $\pm$ 1.12
8	1.48 $\pm$ 1.46	1.46 $\pm$ 1.45	1.46 $\pm$ 1.50	**1.38** $\pm$ 1.24
9	1.74 $\pm$ 1.48	1.82 $\pm$ 1.46	1.62 $\pm$ 1.42	**1.56** $\pm$ 1.48
10	2.32 $\pm$ 1.50	2.10 $\pm$ 1.68	1.94 $\pm$ 1.54	**1.82** $\pm$ 1.46
Average	1.308	1.276	1.234	**1.168**

TABLE 4
Performance Comparisons on AMI for Driver Number Estimation

Group κ	*ARNet*	*T2INET*	Radar-C	Radar
1	**0.34** $\pm$ 0.06	0.32 $\pm$ 0.12	0.27 $\pm$ 0.06	0.25 $\pm$ 0.09
2	**0.37** $\pm$ 0.08	0.36 $\pm$ 0.07	0.25 $\pm$ 0.08	0.28 $\pm$ 0.03
3	0.21 $\pm$ 0.04	0.21 $\pm$ 0.08	0.26 $\pm$ 0.08	**0.27** $\pm$ 0.03
4	0.16 $\pm$ 0.08	0.24 $\pm$ 0.05	0.22 $\pm$ 0.05	**0.25** $\pm$ 0.04
5	0.19 $\pm$ 0.06	**0.23** $\pm$ 0.08	0.19 $\pm$ 0.08	0.18 $\pm$ 0.06
6	0.18 $\pm$ 0.05	0.22 $\pm$ 0.04	**0.26** $\pm$ 0.06	**0.26** $\pm$ 0.07
7	0.17 $\pm$ 0.07	0.17 $\pm$ 0.05	0.20 $\pm$ 0.08	**0.23** $\pm$ 0.02
8	**0.19** $\pm$ 0.06	**0.19** $\pm$ 0.06	0.14 $\pm$ 0.05	0.18 $\pm$ 0.04
9	0.15 $\pm$ 0.08	0.14 $\pm$ 0.08	0.20 $\pm$ 0.04	**0.22** $\pm$ 0.08
10	0.16 $\pm$ 0.07	0.26 $\pm$ 0.04	0.26 $\pm$ 0.05	**0.27** $\pm$ 0.04
Average	0.212	0.234	0.225	**0.239**

ten tests. For the three lost cases, our approach falls behind with marginal differences, e.g., 0.14 at most. As shown by the average experiment results in the last row of Table 3, Radar-Cachieves slightly better performance than the state-of-the-art approaches, i.e., *ARNet*and *T2INET*. It implies that our learning model is more effective on capturing driving style features from raw GPS data. By incorporating the driving context information, Radar further improves Radar-Cby reducing average MAE from 1.234 to 1.168. Overall, our approach Radar can improve *ARNet*and *T2INET*on the performance metric of MAE by 10.7% and 8.5%, respectively.

Table 4 presents the AMI results and deviations, where we also mark the best AMI of each group in bold. Similarly, we find that Radar outperforms other two baselines in most cases, with six wins out of ten tests. The results in Table 4 are in accordance with the results in Table 3. In general, a better clustering quality (i.e., a larger AMI) potentially leads to a better estimation of driver number (i.e., a smaller MAE). The average AMI values of the four approaches are 0.212, 0.234, 0.225, and 0.239, respectively. The results in both Tables 3 and 4 demonstrate that Radar is capable of learning more effective and accurate driving style representations, which thus well support the application of driver number estimation, with smaller MAE and larger AMI.

Effectiveness of Data Augmentation. We also conduct experiments to investigate our data augmentation strategies by comparing with state-of-the-art approaches under different amount of training data. In particular, we enhance Radar with two data augmentation strategies, i.e., *PFD* and *AD*. Similarly, the testing drivers are assigned with limited training data, while auxiliary drivers have sufficient data for training. In this experiment, we repeat 10 runs for each κ value, and report the average results of all κ settings.

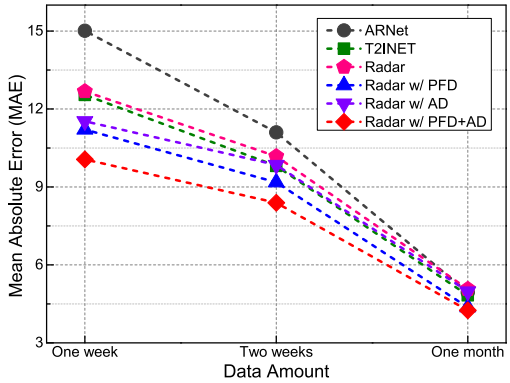


Fig. 11. Comparisons on MAE with different amount of training data.

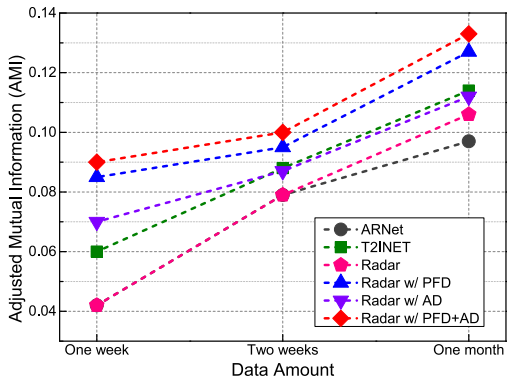


Fig. 12. Comparisons on AMI with different amount of training data.

Fig. 11 compares MAE results of different approaches for the testing drivers with varied amount of training data. With insufficient training data, the MAE results of all approaches are relatively large, while Radar enhanced with effective data augmentation can largely reduce the estimation errors. For example, Radar integrated with *PFD* and *AD* outperforms *ARNet*and *T2INET*by reducing MAE by 33% and 20%, respectively, when testing drivers have only one week of training data. Compared to Radar without data augmentation, we find that Radar integrated with *PFD* or *AD* could improve the MAE performance by introducing more training data.

We observe similar experimental results on the metric of AMI, as shown in Fig. 12. With the effective data augmentation strategies like *PFD* and *AD*, Radar can learn more robust and accurate driving style representations for drivers with limited data, and thus achieve better clustering performance based on their representations. From Fig. 12, we see that Radar with *PFD+AD* has the highest AMI score among all the four approaches.

7 Discussion

In this section, we discuss some design choices and privacy protections of Radar.

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

TABLE 5
Performance Comparisons on $Acc@1$ for the Two Average Driving Speed Vector Designs with Different Amount of Data

Speed Vector Type	One week	Two weeks	One month
Original (five dimensions)	22.1%	25.7%	28.9%
Extended (ten dimensions)	22.3%	25.6%	29.1%

TABLE 6
Comparisons on $Acc@1$ with Different Number of Drivers to Be Identified

# of Drivers	*ARNet*	*T2INET*	Radar
10	20.6%	23.1%	28.9%
30	19.6%	22.4%	24.8%

Influence of Driver's Driving Speeds. Radar exploits the driver's driving speeds, which are calculated from the GPS trajectory data, to estimate road traffic conditions and select auxiliary drivers. The involvement of drivers into Radar may have an impact on the context inference due to the variances of drivers' driving styles. To diminish the impact, we compute average travel speed of a road segment from sufficient number, e.g., ≥ 10, of vehicles passing by the road segment within a time slot. If there are insufficient vehicles, we apply temporal-spatial interpolations to infer the traffic conditions of uncovered road segments. Furthermore, we calculate the relative speeds to represent the traffic conditions while learning a driver's driving style representation (see Section 4.3). In the future, we may import traffic speeds from an independent data source, e.g., transport agency, to avoid the involvement of drivers for traffic estimations.

In Section 5.2, we calculate an average driving speed vector $\mathbf{v}_u$, which is a vector of five dimensions to represent driver u's average driving speeds on five typical road types, and make use of these vectors for auxiliary driver selection. The vector $\mathbf{v}_u$ omits the varying speeds of drivers across time of the day, and thus we conduct experiments to investigate whether such a vector design is sufficient for our auxiliary driver based data augmentation. To this end, we divide time of the day into two categories, i.e., *rush hours* (e.g., 6:00-9:00AM and 17:00PM-20:00PM) and *non-rush hours* (the remaining hours of the day), and calculate a vector of ten dimensions to represent a driver's average driving speeds on the five road types and two time categories. We compare their performance with the application of driver identification, and present the accuracy results on $acc@1$ in Table 5. We find that the two vector types achieve similar accuracy results, with minimal differences. It implies that the original average driving speed vector design is adequate for selecting the suitable auxiliary drivers. Furthermore, considering that the data sparsity issue would be even worse if we use finer segmentation on time of the day, we thus adopt the original average driving speed vector design for the auxiliary driver based data augmentation, which is proved to be simple yet effective.

Accuracy of Driver Identification With More Drivers. We also examine the effectiveness of Radar when the number of drivers to be identified becomes larger. Specifically, we conduct an extra experiment on the application of driver identification with 30 randomly selected drivers. The accuracy results on $acc@1$ are shown in Table 6, where we see that the identification accuracy of all the three methods is slightly decreased when there are more drivers to be identified. Compared to *ARNet* [13] and *T2INET* [30], Radar still achieves the highest accuracy. When the number of to-be-identified drivers becomes larger, there are more drivers who may share quite similar driving styles and thus the identification performances of the three methods drop. To further improve the identification accuracy, we could include more features, e.g., types of vehicles that generate the GPS trajectories, into the process of driving style representation learning to enhance the distinguishability of driving style representations. We leave this as a future work.

Privacy Protections. Radar has several mechanisms to protect drivers' privacy. During the model training phase, Radar will map GPS trajectories of the same driver into the same anonymized ID, and remove the beginning and ending parts of each GPS trajectory to avoid possible personal privacy leakage on the trip origin and destination, which may be inferred as the driver's home or workplace. Once the learning model is well trained, the user (e.g., an insurance company) can utilize the model to learn the driving style representation from a given anonymized GPS trajectory. The derived driving style representation is the high-level abstract of the driver's driving behavior and driving habits, which does not contain any hint about a driver's sensitive location information.

8 Conclusion

In this article, we present an adversarial driving style representation learning approach – Radar. Different from previous works, Radar not only extracts statistic features from raw GPS trajectory data, but also builds contextual features by jointly considering road conditions, geographic semantics, and traffic conditions. We further exploit an advanced semi-supervised GAN architecture to construct the learning model to compute more effective and accurate driving style representations. To address the cold-start problem, we propose several basic data augmentation strategies and an advanced auxiliary driver based data augmentation strategy, which can help drivers with insufficient training data to accurately learn their driving style representations. Experiment results from a large GPS trajectory dataset demonstrate that Radar outperforms state-of-the-art approaches on two benchmark applications. Moreover, the effectiveness of our data augmentation strategies has also been validated with extensive experiments.

References

[1] Road types in OpenStreetMap. Accessed: Aug. 2022. [Online]. Available: https://wiki.openstreetmap.org/wiki/Key:highway
[2] S. Baee, E. Pakdamanian, I. Kim, L. Feng, V. Ordonez, and L. Barnes, "MEDIRL: Predicting the visual attention of drivers via maximum entropy deep inverse reinforcement learning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13178–13188.
[3] M. R. Carlos, L. C. González, J. Wahlström, G. Ramírez, F. Martínez, and G. Runger, "How smartphone accelerometers reveal aggressive driving behavior? -The key is the representation," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3377–3387, Aug. 2020.
[4] G. Castignani, T. Derrmann, R. Frank, and T. Engel, "Driver behavior profiling using smartphones: A low-cost platform for driver monitoring," *IEEE Intell. Transp. Syst. Mag.*, vol. 7, no. 1, pp. 91–102, Spring 2015.

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

[5] T. K. Chan, C. S. Chin, H. Chen, and X. Zhong, "A comprehensive review of driver behavior analysis utilizing smartphones," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 10, pp. 4444–4475, Oct. 2020.

[6] C. Chen, S. Jiao, S. Zhang, W. Liu, L. Feng, and Y. Wang, "Tripimputor: Real-time imputing taxi trip purpose leveraging multi-sourced urban data," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 10, pp. 3292–3304, Oct. 2018.

[7] C. Chen, Q. Liu, X. Wang, C. Liao, and D. Zhang, "Semi-Traj2-Graph: Identifying fine-grained driving style with GPS trajectory data via multi-task learning," *IEEE Trans. Big Data*, vol.1, no. 1, pp.1–15, Mar. 2021.

[8] C. Chen et al., "iBOAT: Isolation-based online anomalous trajectory detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 806–818, Jun. 2013.

[9] C. Chen, D. Zhang, X. Ma, B. Guo, L. Wang, Y. Wang, and E. Sha, "Crowddeliver: Planning city-wide package delivery paths leveraging the crowd of taxis," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 6, pp. 1478–1496, Jun. 2017.

[10] A. Chowdhury, T. Chakravarty, A. Ghose, T. Banerjee, and P. Balamuralidhar, "Investigations on driver unique identification from smartphone's GPS data alone," *J. Adv. Transp.*, vol. 2018, 2018, Art. no. 9702730.

[11] Z. Dai, Z. Yang, F. Yang, W. W. Cohen, and R. R. Salakhutdinov, "Good semi-supervised learning that requires a bad GAN," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1–11.

[12] W. Dong, J. Li, R. Yao, C. Li, T. Yuan, and L. Wang, "Characterizing driving styles with deep learning," 2016, *arXiv:1607.03611*.

[13] W. Dong, T. Yuan, K. Yang, C. Li, and S. Zhang, "Autoencoder regularized network for driving style representation learning," in *Proc. Int. Joint Conf. Artif. Intell.*, 2017, pp. 1603–1609.

[14] S. Ezzini, I. Berrada, and M. Ghogho, "Who is behind the wheel? Driver identification and fingerprinting," *J. Big Data*, vol. 5, no. 1, 2018, Art. no. 9.

[15] J. Fang, D. Yan, J. Qiao, J. Xue, and H. Yu, "DADA: Driver attention prediction in driving accident scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 6, pp. 4959–4971, Jun. 2022.

[16] Z. Fang et al., "MoCha: Large-scale driving pattern characterization for usage-based insurance," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2021, pp. 2849–2857.

[17] J. Feng et al., "User identity linkage via co-attentional neural network from heterogeneous mobility data," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 2, pp. 954–968, Feb. 2022.

[18] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[19] U. Fugiglando et al., "Driving behavior analysis through CAN bus data in an uncontrolled environment," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 2, pp. 737–748, Feb. 2019.

[20] U. Fugiglando, P. Santi, S. Milardo, K. Abida, and C. Ratti, "Characterizing the "driver DNA" through can bus data analysis," in *Proc. 2nd ACM Int. Workshop Smart, Auton., Connected Veh. Syst. Serv.*, 2017, pp. 37–41.

[21] J. Gao, X. Song, Q. Wen, P. Wang, L. Sun, and H. Xu, "RobustTAD: Robust time series anomaly detection via decomposition and convolutional neural networks," in *Proc. 6th ACM KDD Workshop Mining Learn. Time Ser.*, 2020, pp. 1–6.

[22] Q. Gao, F. Zhou, K. Zhang, F. Zhang, and G. Trajcevski, "Adversarial human trajectory learning for trip recommendation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol.1, no.1, pp. 1–13, Feb. 2021.

[23] S. Gelmini, S. C. Strada, M. Tanelli, S. M. Savaresi, and V. Biase, "Online assessment of driving riskiness via smartphone-based inertial measurements," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 9, pp. 5555–5565, Sep. 2021.

[24] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2014, pp. 139–144.

[25] M. Guangyu Li et al., "DBUS: Human driving behavior understanding system," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, 2019, pp. 2436–2444.

[26] D. Hallac et al., "Driver identification using automobile sensor data from a single turn," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2016, pp. 953–958.

[27] B. He, D. Zhang, S. Liu, H. Liu, D. Han, and L. M. Ni, "Profiling driver behavior for personalized insurance pricing and maximal profit," in *Proc. IEEE Int. Conf. Big Data*, 2018, pp. 1387–1396.

[28] X. Hu, Y. Han, and Z. Geng, "Novel trajectory representation learning method and its application to trajectory-user linking," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–9, 2021.

[29] B. K. Iwana and S. Uchida, "An empirical survey of data augmentation for time series classification with neural networks," *PLoS One*, vol. 16, no. 7, pp. 1–32, 2021.

[30] T. Kieu, B. Yang, C. Guo, and C. S. Jensen, "Distinguishing trajectories from different drivers using incompletely labeled trajectories," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2018, pp. 863–872.

[31] M. Kuderer, S. Gulati, and W. Burgard, "Learning driving styles for autonomous vehicles from demonstration," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 2641–2646.

[32] J. Li et al., "Drive2friends: Inferring social relationships from individual vehicle mobility data," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5116–5127, Jun. 2020.

[33] J. Li, F. Zeng, Z. Xiao, Z. Zheng, H. Jiang, and Z. Li, "Social relationship inference over private vehicle mobility data," *IEEE Trans. Veh. Technol*, vol. 70, no. 6, pp. 5221–5233, Jun. 2021.

[34] N. Lin, C. Zong, M. Tomizuka, P. Song, Z. Zhang, and G. Li, "An overview on study of identification of driver behavior characteristics for automotive control," *Math. Problems Eng.*, vol. 2014, 2014, Art. no. 569109.

[35] Z. Liu, Z. Gong, J. Li, and K. Wu, "Mobility-aware dynamic taxi ridesharing," in *Proc. IEEE Int. Conf. Data Eng.*, 2020, pp. 961–972.

[36] Z. Liu, Z. Li, M. Li, W. Xing, and D. Lu, "Mining road network correlation for traffic estimation via compressive sensing," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 7, pp. 1880–1893, Jul. 2016.

[37] Z. Liu, Z. Li, K. Wu, and M. Li, "Urban traffic prediction from mobility data using deep learning," *IEEE Netw.*, vol. 32, no. 4, pp. 40–46, Jul./Aug. 2018.

[38] Z. Liu, J. Zheng, Z. Gong, H. Zhang, and K. Wu, "Exploiting multi-source data for adversarial driving style representation learning," in *Proc. Int. Conf. Database Syst. Adv. Appl.*, pp. 491–508, 2021.

[39] Z. Liu, P. Zhou, Z. Li, and M. Li, "Think like a graph: Real-time traffic estimation at city-scale," *IEEE Trans. Mobile Comput.*, vol. 18, no. 10, pp. 2446–2459, Oct. 2019.

[40] L. Mason, J. Baxter, P. L. Bartlett, and M. R. Frean, "Boosting algorithms as gradient descent," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2000, pp. 512–518.

[41] C. Miao, J. Wang, H. Yu, W. Zhang, and Y. Qi, "Trajectory-user linking with attentive recurrent network," in *Proc. ACM Int. Conf. Autonom. Agents Multiagent Syst.*, 2020, pp. 878–886.

[42] S. Moosavi, P. D. Mahajan, S. Parthasarathy, C. Saunders-Chukwu, and R. Ramnath, "Driving style representation in convolutional recurrent neural network model of driver identification," 2021, *arXiv:2102.05843*.

[43] P. Newson and J. Krumm, "Hidden Markov map matching through noise and sparseness," in *Proc. ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2009, pp. 336–343.

[44] M. Nice, S. Elmadani, R. Bhadani, M. Bunting, J. Sprinkle, and D. Work, "CAN coach: Vehicular control through human cyber-physical systems," in *Proc. ACM/IEEE Int. Conf. Cyber-Physical Syst.*, 2021, pp. 132–142.

[45] A. Odena, "Semi-supervised learning with generative adversarial networks," in *Proc. Workshop Data Efficient Mach. Learn.*, 2016, pp. 1–3.

[46] S.-E. Ramah, A. Bouhoute, K. Boubouh, and I. Berrada, "One step further towards real-time driving maneuver recognition using phone sensors," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 10, pp. 6599–6611, Oct. 2021.

[47] H. Ren, M. Pan, Y. Li, X. Zhou, and J. Luo, "ST-SiameseNet: Spatio-temporal siamese networks for human mobility signature identification," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2020, pp. 1306–1315.

[48] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 2234–2242.

[49] A. Sarker and H. Shen, "DeepDMC: A traffic context independent deep driving maneuver classification framework," in *Proc. IEEE Int. Conf. Mobile Ad Hoc Smart Syst.*, 2021, pp. 455–463.

[50] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 815–823.

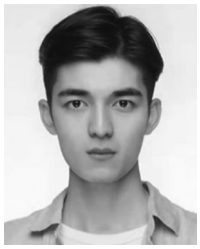
[51] Y. Song, D. Jiang, Y. Liu, Z. Qin, C. Tan, and D. Zhang, "HERMAS: A human mobility embedding framework with large-scale cellular signaling data," *Proc. ACM InterAct., Mobile, Wearable Ubiquitous Technol.*, vol. 5, no. 3, pp. 1–21, 2021.

[52] Y. Sun et al., "CoDriver ETA: Combine driver information in estimated time of arrival by driving style learning auxiliary task," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 5, pp. 4037–4048, May 2022.

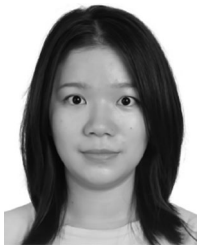
Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.

[53] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res.*, vol. 11, pp. 2837–2854, 2010.
[54] B. Wang, S. Panigrahi, M. Narsude, and A. Mohanty, "Driver identification using vehicle telematics data," Tech. Rep. 2017-01-1372, SAE Technical Paper, 2017.
[55] P. Wang, Y. Fu, J. Zhang, P. Wang, Y. Zheng, and C. Aggarwal, "You are how you drive: Peer and temporal-aware representation learning for driving behavior analysis," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 2457–2466.
[56] Q. Wen et al., "Time series data augmentation for deep learning: A survey," in *Proc. Int. Joint Conf. Artif. Intell.*, 2021, pp. pp. 4653–4660.
[57] T.-F. Wu, C.-J. Lin, and R. C. Weng, "Probability estimates for multi-class classification by pairwise coupling," *J. Mach. Learn. Res.*, vol. 5, no. Aug, pp. 975–1005, 2004.
[58] F. Xia, J. Wang, X. Kong, Z. Wang, J. Li, and C. Liu, "Exploring human mobility patterns in urban scenarios: A trajectory data perspective," *IEEE Commun. Mag.*, vol. 56, no. 3, pp. 142–149, Mar. 2018.
[59] X. Xu, J. Yu, Y. Chen, Y. Zhu, S. Qian, and M. Li, "Leveraging audio signals for early recognition of inattentive driving with smartphones," *IEEE Trans. Mobile Comput.*, vol. 17, no. 7, pp. 1553–1567, Jul. 2018.
[60] S. Yang, C. Wang, H. Zhu, and C. Jiang, "APP: Augmented proactive perception for driving hazards with sparse GPS trace," in *Proc. ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, 2019, pp. 21–30.
[61] J. Yu, Z. Chen, Y. Zhu, Y. Chen, L. Kong, and M. Li, "Fine-grained abnormal driving behaviors detection and identification with smartphones," *IEEE Trans. Mobile Comput.*, vol. 16, no. 8, pp. 2198–2212, Aug. 2017.
[62] Y. Yu, H. Tang, F. Wang, L. Wu, T. Qian, T. Sun, and Y. Xu, "TULSN: Siamese network for trajectory-user linking," in *Int. Joint Conf. on Neural Netw.*, 2020, pp. 1–8.
[63] C. Zhang, J. Zhu, W. Wang, and J. Xi, "Spatiotemporal learning of multi-vehicle interaction patterns in lane-change scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6446–6459, Jul. 2022.
[64] K. Zhao et al., "Discovering subsequence patterns for next POI recommendation," in *Proc. Conf. Assoc. Advance. Artif. Intell.*, 2020, pp. 3216–3222.
[65] Y. Zheng, "Trajectory data mining: An overview," *ACM Trans. Intell. Syst. Technol.*, vol. 6, no. 3, pp. 1–41, 2015.
[66] F. Zhou, Q. Gao, G. Trajcevski, K. Zhang, T. Zhong, and F. Zhang, "Trajectory-user linking via variational autoencoder," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 3212–3218.
[67] F. Zhou, X. Liu, T. Zhong, and G. Trajcevski, "MetaMove: On improving human mobility classification and prediction via metal-earning," *IEEE Trans. Cybern.*, vol. 52, no. 8, pp. 8128–8141, Aug. 2022.

Zhidan Liu received the PhD degree in computer science and technology from Zhejiang University, Hangzhou, China, in 2014. After that, he worked as a Research Fellow in Nanyang Technological University, Singapore. He is currently an associate professor with College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His research interests include mobile computing, big data analytics, Internet of Things, and urban computing. He is a member of IEEE, ACM, and CCF.



Junhong Zheng received the BS degree in software engineering from the Zhuhai College of Science and Technology, Zhuhai, China, in 2019. He is currently a third-year master student at the College of Computer Science and Software Engineering of Shenzhen University, under the supervision of Dr. Zhidan Liu. His research interests are in the areas of trajectory data analysis, traffic modeling, and urban computing.



Jinye Lin received the BE degree in Internet of Things engineering from Shenzhen University, Shenzhen, China, in 2021. She is currently a first-year master student at the College of Computer Science and Software Engineering of Shenzhen University, under the supervision of Dr. Zhidan Liu. Her research interests are in the areas of urban computing, traffic modeling, and deep learning.

Liang Wang received the PhD degree from the Shenyang Institute of Automation (SIA), Chinese Academy of Sciences, Shenyang, China, in 2014. He is currently an associate professor with Northwestern Polytechnical University, Xi'an, China. His research interests include ubiquitous computing, mobile crowdsensing, and Internet of Things.

Kaishun Wu received his PhD degree in computer science and engineering from The Hong Kong University of Science and Technology (HKUST), Hong Kong, China, in 2011. After that, he worked as a Research assistant professor at HKUST. In 2013, he joined Shenzhen University as a Distinguish Professor. Currently, he is a Professor of the DSA & IoT Thrust Area under the Information Hub at Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China. He has co-authored 2 books and published over 100 high quality research papers in international leading journals and primer conferences, like IEEE TMC, IEEE TPDS, ACM MobiCom, IEEE INFOCOM. He is the inventor of 6 US and over 90 Chinese pending patents. He received 2012 Hong Kong Young Scientist Award, 2014 Hong Kong ICT awards: Best Innovation, and 2014 IEEE ComSoc Asia-Pacific Outstanding Young Researcher Award. He is an IET Fellow.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/csdl.

Authorized licensed use limited to: SHENZHEN UNIVERSITY. Downloaded on November 12,2023 at 04:35:38 UTC from IEEE Xplore. Restrictions apply.