Naturalistic E-Scooter Maneuver Recognition with Federated Contrastive Rider Interaction Learning

MAHAN TABATABAIE, University of Connecticut, USA
SUINING HE, University of Connecticut, USA

Smart micromobility, particularly the electric (e)-scooters, has emerged as an important ubiquitous mobility option that has proliferated within and across many cities in North America and Europe. Due to the fast speed (say, ~15km/h) and ease of maneuvering, *understanding how the micromobility rider interacts with the scooter* becomes essential for the e-scooter manufacturers, e-scooter sharing operators, and rider communities in promoting riding safety and relevant policy or regulations.

In this paper, we propose FCRIL, a novel ***F***ederated maneuver identification and ***C***ontrastive e-scooter ***R***ider ***I***nteraction ***L***earning system. FCRIL aims at: *(i)* understanding, learning, and identifying the e-scooter rider interaction behaviors during *naturalistic riding* (NR) experience (without constraints on the data collection settings); and *(ii)* providing a novel federated maneuver learning model training and contrastive identification design for our proposed rider interaction learning (RIL). Towards the prototype and case studies of FCRIL, we have harvested an NR behavior dataset based on the inertial measurement units (IMUs), e.g., accelerometer and gyroscope, from the ubiquitous smartphones/embedded IoT devices attached to the e-scooters. Based on the harvested IMU sensor data, we have conducted extensive data analytics to derive the relevant rider maneuver patterns, including time series, spectrogram, and other statistical features, for the RIL model designs. We have designed a contrastive RIL network which takes in these maneuver features with class-to-class differentiation for comprehensive RIL and enhanced identification accuracy. Furthermore, to enhance the dynamic model training efficiency and coping with the emerging micromobility rider data privacy concerns, we have designed a novel asynchronous federated maneuver learning module, which asynchronously takes in multiple sets of model gradients (e.g., based on the IMU data from the riders' smartphones) for dynamic RIL model training and communication overhead reduction. We have conducted extensive experimental studies with different smartphone models and stand-alone IMU sensors on the e-scooters. Our experimental results have demonstrated the accuracy and effectiveness of FCRIL in learning and recognizing the e-scooter rider maneuvers.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing**.

Additional Key Words and Phrases: E-scooter rider interaction learning, supervised contrastive maneuver learning, asynchronous federated learning, e-scooter naturalistic riding behavior analysis

ACM Reference Format:
Mahan Tabatabaie and Suining He. 2022. Naturalistic E-Scooter Maneuver Recognition with Federated Contrastive Rider Interaction Learning. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 4, Article 205 (December 2022), 27 pages. https://doi.org/10.1145/3570345

Authors' addresses: Mahan Tabatabaie, Department of Computer Science and Engineering, University of Connecticut, Storrs, CT, USA, mahan.tabatabaie@uconn.edu; Suining He, Department of Computer Science and Engineering, University of Connecticut, Storrs, CT, USA, suining.he@uconn.edu.

2474-9567/2022/12-ART205 $15.00
https://doi.org/10.1145/3570345

1 INTRODUCTION

Thanks to the recent progress in the internet of things (IoT), electric motors, and batteries, as well as the growing needs for a clean mobility platform in response to climate changes [36], smart micromobility systems, represented by electric (e)-scooter, has become an increasingly popular means of the first-/last-mile commutes as well as recreational travel for a growing number of smart cities. According to the market survey, the global electric scooters' market size, estimated at USD $20.78 billion in 2021, is expected to expand at a compound annual growth rate (CAGR) of 7.8% from 2022 to 2030 [2]. In addition, there is a rising trend in promoting and expanding the dockless e-scooter sharing in many metropolitan cities in North America and Europe [4]. For example, the New York City (NYC) Department of Transportation in the U.S. will double their dockless e-scooter sharing fleet from 3,000 to 6,000 scooters in 2022 [5].

As a growing fleet of e-scooters (private-owned or shared) is expanding within and across cities, the accompanying concerns and pressing needs from the e-scooter manufacturers (such as Razor, Segway, and SuperPedestrian) as well as the dockless e-scooter sharing service operators (for instance, Bird and Lime) lie in *understanding*, *learning*, and *recognizing* the e-scooter riders' *maneuver behaviors* (e.g., left turn, right turn, and acceleration). We term this problem as the *rider interaction learning* (RIL) in this study. According to the U.S. Consumer Product Safety Commission (CSC) report [13], the number of accidents related to the e-scooters and other micromobility platforms has increased by 70% during 2017–2020, i.e., from 34,000 to 57,800 cases, due to various mechanical, electrical, and human factors. In the face of this challenge, accurate and effective RIL designs can benefit the above-mentioned major e-scooter manufacturers and sharing service operators in: *(i)* developing safety measures and improving interaction designs upon the manufactured and deployed scooters (e.g., handlebars, brake handles, and electric motor designs); *(ii)* reducing conflicts and collision risks in the urban roads (with motorized vehicles), sidewalks (with pedestrians), and bike/scooters lanes (with pedaled bikes and other scooters); and *(iii)* understanding the potentially aggressive maneuvers and promoting safe riding conducts or policies within and across the rider communities (as illustrated in Fig. 1).

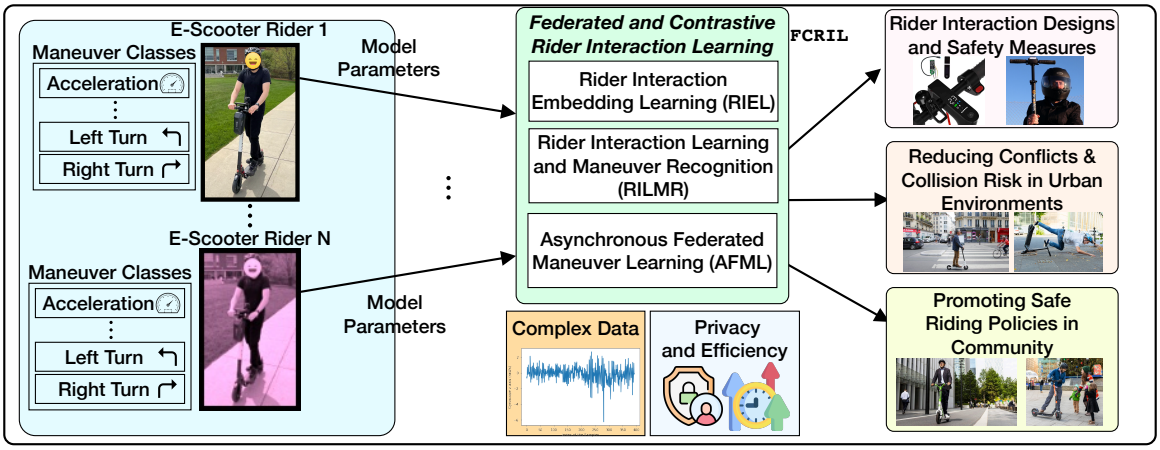

Fig. 1. Motivations of the e-scooter rider interaction learning (RIL) and its potential ubiquitous computing applications.

Motivated by the above needs, we will conduct a novel pilot study that focuses on the inertial measurement units (IMUs), e.g., accelerometers and gyroscopes, that are ubiquitously available in existing smartphones and embedded/IoT devices (such as an embedded/stand-alone IMU sensor, or wearable devices like smartwatches).

Compared with the Global Position System (GPS) [12] and camera-based approaches [41], our prototype studies focus on the IMU sensors that can provide more ubiquitous and less privacy-intrusive understandings of the scooter riding behaviors under complex urban environments (for instance, GPS signals may be blocked or weakened by the city buildings and other built urban infrastructures, and the camera-based approaches may be prone to low-light situations). Since many e-scooters are emerging as highly-integrated and smart *personal mobility systems* (PMSs) with a myriad of on-board sensors, communication, and computing capabilities available, our pilot studies here will pave new directions for the e-scooter (and other micromobility systems) manufacturers and sharing system operators to re-imagine the interaction designs of e-scooters and promote safer, more convenient, and more entertaining riding experience. In fact, many e-scooter manufacturers and sharing service operators have started to consider expanding the sensing, computing, and communication capabilities of the e-scooters for enhanced riding experience, promoted rider safety, and better fleet management [1, 3].

Towards a ubiquitous riding behavioral analysis and RIL system, this prototype study aims at understanding and addressing the following two important technical challenges:

(1) **Complex rider maneuvers during naturalistic riding experience and interactions with the neighboring environments:** Our studies focus on the settings of *naturalistic riding* (NR), i.e., deriving insights from the e-scooter rider behaviors during everyday trips by recording mobility patterns of the rider and the e-scooter without experimental control. The *goal* of such an NR design is to gain insights into how the e-scooter maneuver behaviors during commutes and recreational rides can be reflected by learning the IMU sensor time series. The collected NR data that are closely related to the real-world riding conditions will help shed light upon the e-scooter rider behaviors in response to the complex urban environments. However, based on our extensive data analytics, we have observed from the NR data that the complex rider maneuvers under various road conditions, spatio-temporal contexts, and sensor settings may lead to complex rider maneuver and behavior patterns, degrading the performance of conventional data processing and feature learning. How to enable a pervasive and adaptive RIL model design is essential for identifying the rider maneuvers during the NR studies.
(2) **Absence of decentralized and efficient RIL designs for emerging smart e-scooter designs:** While e-scooter rider NR data collection may meet the practical needs of effective RIL from the e-scooter manufacturers, sharing service operators, and rider communities, another concern arises regarding how we can provide data privacy-preserving designs, particularly for RIL. There remains an important industrial need as well as research gap on the *federated maneuver learning* for the emerging smart e-scooter rider data collection and RIL process, and how the inclusion of such a mechanism will impact and interact with the RIL remains largely unexplored. Realizing a federated maneuver learning will meet the demands of many micromobility manufacturers and shared service operators who would like to gain a more comprehensive understanding of the riders' behaviors. These stakeholders can further enhance the micromobility vehicle (i.e., e-scooters) safety designs and enforce potential precautionary methods (i.e., risk alarming) to the riders. However, the related sensitive interaction and mobility data should be kept locally on the client side. Furthermore, due to the highly dynamic maneuverability and mobility of the e-scooters and their riders in complex urban environments with dynamic wireless communication connectivities, how to realize the *scalable* and *efficient* RIL, particularly when the synchronized RIL model training may not be available, is worth further exploration.

To overcome the above challenges, we propose and pilot FCRIL, a novel ***F***ederated maneuver identification system with ***C***ontrastive e-scooter ***R***ider ***I***nteraction ***L***earning. To achieve FCRIL, we have made the following three important technical contributions:

(1) **E-Scooter Rider Maneuver Interaction Identification with Contrastive Learning:** Towards piloting this system, we have conducted extensive real-world NR data collection for rider maneuver records (such

as left/right turn, acceleration/deceleration) during commutes and recreational rides on our university campus, and derived the model designs for RIL studies. To handle the complex rider maneuvers, we have derived multiple important rider maneuver feature designs based on the harvested IMU sensor data, including the time series, spectrogram, and other statistical features, to comprehensively characterize the rider interactions when maneuvering the e-scooter. To further differentiate different rider maneuvers for enhanced identification accuracy, we have designed a rider interaction embedding learning (RIEL) module based on a supervised contrastive loss function. Specifically, based on supervised contrastive learning, for each rider maneuver class (e.g., left or right turns), the RIEL module forms the distinguishable rider interaction embeddings, i.e., the encoded features of the rider interactions and behaviors, by differentiating them against the other rider maneuver classes to derive the essential RIL features. We further feed the differentiated rider interaction embeddings to the rider interaction learning and maneuver recognition (RILMR) module, where we have designed the residual layers to yield accurate and generalizable identification results.

(2) **Asynchronous Federated Rider Maneuver Learning:** Towards efficient RIL model training with privacy-preserving implications, we have further designed a scalable asynchronous federated maneuver learning (AFML) mechanism for FCRIL. Specifically, our AFML mechanism will make use of the computing capability that is emerging on smart e-scooters (with smartphones or other on-board computing devices), by distributing the FCRIL training process across them [3]. Our AFML can provide an *asynchronous* and *decentralized* RIL model training mechanism. This way, the RIL training process does not need to be simultaneous among the smart e-scooters (say, in the e-scooter sharing service systems), enhancing the scalability of the FCRIL system (e.g., with about 38% faster convergence than the existing synchronized federated learning designs [34]). This way, our RIL model's performance will be more robust towards the varying mobility, computing constraints, and communication delay of the smart e-scooters in the complex urban environments.

(3) **Extensive E-Scooter Naturalistic Riding Data Analytics and Experimental Studies:** We have conducted extensive experimental studies with FCRIL based on a total of 2,800 rider maneuver records collected with the commercial off-the-shelf e-scooters, four different models of smartphones (iPhone 13 Pro, iPhone 13 Mini, iPhone 12 Mini, and iPhone 7 Plus), and stand-alone IMU sensors (installed on the e-scooters), with various real-world case studies (e.g., road conditions, sensor placement positions, model sensitivity studies, and convergence analysis) and performance evaluations. With the NR data collected from our campus (the college town environment at the University of Connecticut), our experimental studies have corroborated the effectiveness and accuracy (10.95% accuracy improvements on average compared with the baseline approaches) of FCRIL in identifying the rider maneuvers from the NR data, with important efficiency improvement and privacy-preserving implications.

Contributions to UbiComp: Our model designs, system prototyping, and experiment insights will benefit the *UbiComp* communities in the following two perspectives: *(1)* understanding the e-scooter rider interactions in the naturalistic riding (e.g., commutes and recreational rides) using ubiquitous IMU sensors; and *(2)* enhancing privacy-preserving mobility data collection designs for ubiquitous interaction learning. Our RIL framework can serve as an important *building block* to enable other important riding safety and experience enhancement measures for the micromobility vehicle designs (e.g., improving human-scooter interactions through redesign of handlebars and brake handles). Furthermore, FCRIL will help enable privacy-preserving model training for the individual riders and enhance the deployability of RIL in the real-world scenarios (such as the e-scooter sharing services).

To the best of our knowledge, this is the *first* work on analyzing, learning, and understanding the e-scooter rider interactions and maneuver behaviors in the NR settings, with efficiency and data privacy preserving

implications. Despite our current focus on RIL with the IMU sensors, our model insights and system prototype can be beneficial for and extensible to other mobility system interaction designs (such as bike riders [26] and motorized vehicle drivers [43, 44]) and other sensing modalities (e.g., integrating with GPS and other wireless sensing techniques [26]) to advance understandings of the rider maneuvers and enable the advanced micromobility rider assistance systems.

The rest of the paper is organized as follows. We first review the related work in Sec. 2. Then, we overview our system framework, data analysis, and problem formulation in Sec. 3. Afterwards, we present the details of our core model designs towards contrastive rider interaction learning with maneuver recognition in Sec. 4. We present our experimental studies and provide deployment discussion in Secs. 5 and 6, respectively. We finally conclude in Sec. 7.

2 RELATED WORK

We briefly go through the related work in the following two categories.

• **Ubiquitous Computing for Micromobility:** Thanks to the advances of IoTs and ubiquitous computing, how to characterize and learn the *interactions* of the human riders with the smart micromobility vehicles (e.g., e-scooters) starts to gain more attention due to the riding safety concerns [10] and operation planning [19] needs. Towards understanding the *macroscopic* interactions, He et al. [20] investigated the e-scooter rider interactions with the urban environments, such as the points-of-interest. He et al. [19] studied the e-scooter rider mobility interactions with the dynamically reconfigured distributions. To further capture the spatio-temporal interaction patterns, Merlin et al. [35] proposed a segment-level origin-destination demand prediction for the e-scooter sharing service, and Xu et al. [52] introduced a framework for real-time prediction of e-scooter demands. Further *microscopic* interactions of the micromobility riders and the pedestrians have been studied in [6]. Jin et al. designed a smartphone-based assistive tool to enhance cyclist safety based on acoustic ranging [26]. Ling et al. studied the cyclist behaviors at the signalized intersections [32]. Ding et al. [14] studied geo-referenced egocentric video data harvested from the handlebar cameras of cyclists to learn and capture cyclists' behaviors.

Despite the aforementioned micromobility studies, how the riders interact with the scooters and how to identify the complex interactions from the rider maneuver behavior sensing data remains largely unexplored. Our pilot studies with the proposed FCRIL fill the important gap, and provide a novel rider interaction learning (RIL) system design with the novel contrastive recognition [47] and scalable federated maneuver learning mechanisms. We note that our work focuses on microscopic rider interaction learning and maneuver identification, which is orthogonal to the transportation mode detection approaches that leverage conventional machine learning [25, 55] and deep learning [24, 51] techniques to detect the metro/bus/car/bike use. However, with the increasing popularity of e-scooter in urban mobility networks, our work can be further integrated with the transportation mode detection frameworks [24, 31, 46, 51, 55], expanding the sensing capability of the mobile apps for recommending mobility options.

• **Federated Learning for Human Activity Recognition:** With the proliferation of mobile sensors and IoTs, federated learning [29, 38, 53] has emerged as an important machine learning training mechanism that enables model training while keeping the sensor data close to each of the individual users, thus ensuring that the data for training and inference does not leave the client's (user's) own device (e.g., their smartphones). This design benefits privacy preservation and helps reduce communication overhead when training and applying the human activity recognition model [42]. Ouyang et al. [38] studied a similarity-aware federated learning system that can provide high model accuracy and low communication overhead for human activity recognition applications. Tu et al. [48] proposed a dynamic layer-sharing scheme that learns the behavioral similarity among clients' model weights to form the sharing structure and merges their models accordingly in an iterative, bottom-up, and layer-wise manner.

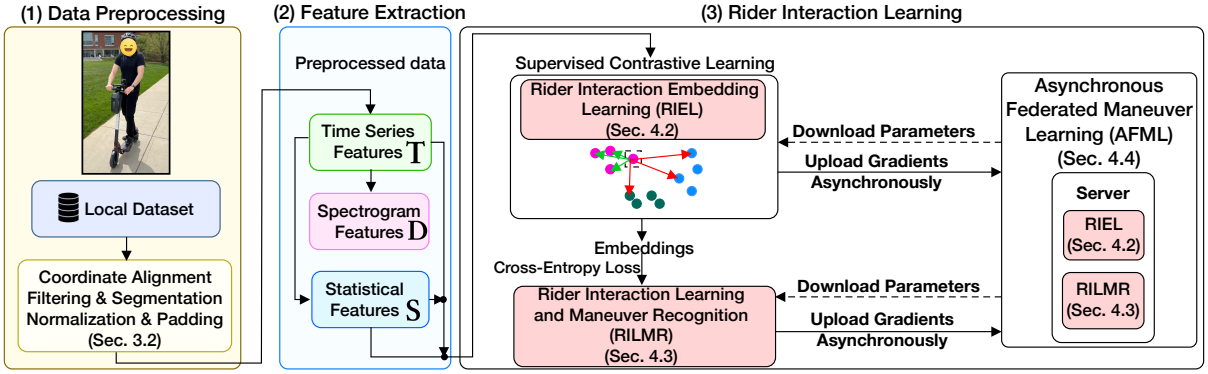

Fig. 2. Overview of FCRIL framework: (1) *Data Processing*; (2) *Feature Extraction*; and (3) *Rider Interaction Learning*.

In addition to expanding the human activity recognition, how to further enhance the scalability and efficiency of federated learning is attracting much attention recently. Lee et al. [28] improved the ring-based federated learning method to reduce its high variance and increase its scalability. Reisizadeh et al. [40] introduced a federated averaging method that only periodically averages on the server side to improve the scalability of their algorithm.

While various federated learning aggregation algorithms have been studied in the context of human activity recognition, how the federated learning mechanism impacts or scalably interacts with the e-scooter rider interaction learning (RIL), particularly under highly mobile, dynamic, and urban naturalistic riding settings, remains largely unexplored. Our studies here on RIL fill these essential research gaps, and will benefit the smart e-scooter manufacturers and sharing service operators in designing their future NR data collection and RIL model training mechanisms.

3 SYSTEM OVERVIEW, DATA ANALYSIS, AND PROBLEM FORMULATION

We first overview the system framework in Sec. 3.1. Then we show the dataset preparation and problem definition of RIL in Sec. 3.2.

3.1 System Overview

Fig. 2 provides the overview of the information flow of our FCRIL framework, which consists of the following three major phases:

(1) Data Preprocessing: As discussed in Sec. 1, our FCRIL study focuses on the rider behavior data generated from the IMUs to perform RIL, i.e., accelerometer (denoted by $\mathbf{a}$) and gyroscope (denoted by $\mathbf{g}$) in our prototype studies. In this prototype study, we focus on two typical IMU data collection cases: *(a)* using the IMU of the rider's smartphone attached to the e-scooter handlebar, and *(b)* using a stand-alone IMU sensor node that can be attached or installed on the e-scooter (e.g., emulating the rider maneuver analysis with the smart PMSs).

We note that the data generated by the accelerometer and gyroscope are time series or sequence values along the $\dot{x}$, $\dot{y}$, and $\dot{z}$ axes. We respectively denote them by $\mathbf{a}_{\dot{x}}$, $\mathbf{a}_{\dot{y}}$, $\mathbf{a}_{\dot{z}}$, $\mathbf{g}_{\dot{x}}$, $\mathbf{g}_{\dot{y}}$, and $\mathbf{g}_{\dot{z}}$, where $\{\dot{x}, \dot{y}, \dot{z}\}$ represent the sensing device's coordinate system. To ensure consistency in the model training, we align the coordinate system of the smartphone or the stand-alone IMU sensor, $\{\dot{x}, \dot{y}, \dot{z}\}$, to that of the e-scooter, denoted as $\{x, y, z\}$. We perform the coordinate alignment by calculating the rotation matrices using the magnetometer readings of the smartphone or the stand-alone IMU sensor.

Furthermore, since the generated data by the sensors can be very noisy while the rider maneuvers the e-scooter, we filter the IMU time series data based on the moving average method. Then, we segment the data using the sliding window (e.g., 10s in our study) and label each segment with the corresponding rider maneuver class based on the ground-truth information provided in the recorded video frames and GPS coordinates (for the purpose of our prototype studies only). We note that these more privacy-sensitive videos/GPS trajectories will not be collected (or will be regulated [50]) in the real-world e-scooter deployment, and our prototype core model designs do not take them into account. Finally, we normalize the values of each segment and pad them with zero values to generate equal-length segments for later feature extraction.

(2) Feature Extraction: After the data preprocessing stage, we take into account and extract a total of three sets of features as follows to serve as the inputs to our model (detailed in Sec. 3.2), i.e., *time series features*, denoted as **T**, of the maneuvers, the *spectrogram features*, denoted as **D**, based on the discrete wavelet transform (DWT) function [21] on **T**, and the other *statistical features*, denoted as **S**, that are retrieved from **T**.

(3) E-Scooter Rider Interaction Learning and Maneuver Recognition: Given the extracted features from (2), the inputs to our FCRIL, i.e., {**T**, **D**, **S**}, are first processed by the *Rider Interaction Embedding Learning* (RIEL) module in Sec. 4.2 based on a novel supervised contrastive learning loss function [27] to generate more distinguishable maneuver embeddings. Then, FCRIL feeds the resulting embeddings to the *Rider Interaction Learning and Maneuver Recognition* (RILMR) module in Sec. 4.3 to further generate the probability scores of each rider maneuver class for RIL. Furthermore, we have designed a scalable Asynchronous Federated Maneuver Learning (AFML) algorithm to handle dynamic rider mobility data collection in the complex urban environments (Sec. 4.4).

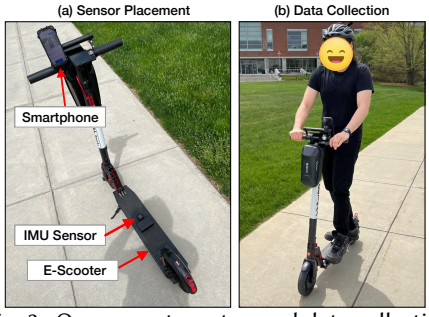

Fig. 3. Our e-scooter setup and data collection for FCRIL prototype development.

3.2 RIL Data Preparation and Problem Definition

Data Collection: We illustrate our data collection setup in Fig. 3(a) and (b). In this prototype study, we mounted and tested four different kinds of smartphones, i.e., iPhone 13 mini, iPhone 13 Pro, iPhone 12 mini, and iPhone 7 Plus, on the e-scooter's handles to harvest the smartphone IMU data (sampling rate: 40Hz) during daily commutes/recreational rides (to record the naturalistic riding experience) on our university campus. We also attach and install a stand-alone or embedded TDK MPU9250 IMU sensor on the e-scooter to harvest the accelerometer and gyroscope readings, as illustrated in Fig. 3(a). We have also harvested the video frames using the smartphone rear-facing cameras as well as smartphone GPS traces for our NR status reviews and rider maneuver ground-truth labeling purposes only (see Figs. 4(a) and (d)). We have followed the local regulations and riding ethics (e.g., wearing a scooter helmet, avoiding rush hours with pedestrian crowds) during the e-scooter riding and the data collection.

Given the harvested IMU and other sensor data, we have labeled a total of 7 different rider maneuvers: *short left turn* (SLT), *short right turn* (SRT), *long left turn* (LLT), *long right turn* (LRT), *straight* (S) *ride*, *acceleration* (AC), and *deceleration* (DC). We have performed the data augmentation and obtained a total of 2,800 records of e-scooter rider maneuvers (400 maneuver records per class), out of which 2,380 are from the smartphones' IMUs, and 420 are from the stand-alone IMU sensor installed on the e-scooters.

Data Preprocessing: For ease of our experimental studies and model designs, we align the coordinate system of the smartphones/sensor devices to that of the e-scooter. Let $\{\dot{x}, \dot{y}, \dot{z}\}$ and $\{x, y, z\}$ be the coordinate system of the IMU device (e.g., the smartphone) and the e-scooter, respectively. Furthermore, we let $\beta_{\dot{x}}$, $\beta_{\dot{y}}$, and $\beta_{\dot{z}}$ be the rotation angles of the smartphone/IMU device along each of x, y, and z axes of the e-scooter coordinate system.

We first calculate the rotation matrices $\mathbf{E}_{\dot{x}}$, $\mathbf{E}_{\dot{y}}$, and $\mathbf{E}_{\dot{z}}$ as shown in Eq. (2). Then, to perform coordinate alignment, we multiply each of the IMU sensor readings with the rotation matrix that is given by

$$\mathbf{E} = \mathbf{E}_{\dot{z}} \cdot \mathbf{E}_{\dot{y}} \cdot \mathbf{E}_{\dot{x}}, \tag{1}$$

where

$$\mathbf{E}_{\dot{x}} = \begin{bmatrix} \cos(\beta_{\dot{x}}) & -\sin(\beta_{\dot{x}}) & 0 \\ \sin(\beta_{\dot{x}}) & \cos(\beta_{\dot{x}}) & 0) \\ 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{E}_{\dot{y}} = \begin{bmatrix} \cos(\beta_{\dot{y}}) & 0 & \sin(\beta_{\dot{y}}) \\ 0 & 1 & 0 \\ -\sin(\beta_{\dot{y}}) & 0 & \cos(\beta_{\dot{y}}) \end{bmatrix}, \text{ and } \quad \mathbf{E}_{\dot{z}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\beta_{\dot{z}}) & -\sin(\beta_{\dot{z}}) \\ 0 & \sin(\beta_{\dot{z}}) & \cos(\beta_{\dot{z}}) \end{bmatrix}, \tag{2}$$

to convert them into the readings in the e-scooter coordinate system.

Afterwards, we filter the data using the moving average method to remove the noise caused by the imprecision of the IMU sensors and those due to the complex road conditions. Next, we apply a sliding window of size W to divide each trip into multiple segments. Then, we use the ground-truth maneuver information (e.g., the recorded video frames) to label each trip segment with a rider maneuver class (e.g., right turn). Finally, we perform min-max normalization to convert the sensor values into [-1,1] range, and pad each segment with zero values to ensure equal-length segments for ease of `RIL` model learning. We note that the padded zeros do not affect the prediction results (similar to other machine learning tasks), and the IMU sensors generally do not generate long sequences of zeros in practice.

Feature Extraction & Visualization: After the preprocessing (including the coordinate alignment) stage, we further derive the following three different `RIL` features to handle the maneuver recognition.

(1) Time Series Features: We first input the i-th segment in the dataset as the time series features for a rider maneuver i. We denote the total set of the time series features as $\mathbf{T} = \{\mathbf{t}_1, \ldots, \mathbf{t}_N\}$, $\mathbf{t}_i \in \mathbb{R}^{W \times 6}$, where W is the length of the time series, N is the total number of the rider maneuvers, and we have a total of six axes from the accelerometer and gyroscope sensors, i.e., $\mathbf{a}_x$, $\mathbf{a}_y$, $\mathbf{a}_z$, $\mathbf{g}_x$, $\mathbf{g}_y$, and $\mathbf{g}_z \in \mathbb{R}^W$. We illustrate examples of the time series features of different maneuver classes for the gyroscope sensor along the z-axis, $\mathbf{g}_z$, as illustrated in Fig. 4(b). Furthermore, Figs. 4(a) and (d) show the GPS traces of the maneuvers and frames of the recorded ground-truth videos, which we use solely for the rider maneuver labeling process.

(2) Spectrogram Features: Time series features directly derived from the IMU sensors do not necessarily provide satisfactory patterns for differentiating and recognizing e-scooter maneuvers. Therefore, we further derive the spectrogram features of the i-th maneuver, $\mathbf{d}_i$, by applying the *discrete wavelet transform* (DWT) based on the Symlet wavelet function [16] upon the harvested time series features of the i-th maneuver, $\mathbf{t}_i$. Specifically, we apply the *Symlet wavelet* function [16] on each of the 6 axes of the accelerometer and gyroscope sensors separately in the time series features, $\mathbf{t}_i$, and stack the results to get the spectrogram features $\mathbf{d}_i \in \mathbb{R}^{W \times 6}$.

From the illustration of Fig. 4(c), we can observe more complex patterns within the rider maneuvers have been captured. For instance, we can observe that the spectrogram features of the long left turn (LLT) and short left turn (SLT) demonstrate more dissimilarities from accelerometer and gyroscope aspects, and therefore help to differentiate the two rider maneuver classes for enhanced identification accuracy. We note that the advantage of using DWT over other wavelet function transforms lies in its *joint* consideration of both frequency and time domain information, thus yielding accurate `RIL` results for `FCRIL`. We denote the total set of the spectrogram features by $\mathbf{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_N\}$, $\mathbf{d}_i \in \mathbb{R}^{W \times 6}$.

(3) Statistical Features: To generate the statistical features of the i-th maneuver, $\mathbf{s}_i$, we apply r statistical functions on each of the 6 axes of the accelerometer and gyroscope sensors to produce $r \times 6$ features. We denote the set of the statistical features by $\mathbf{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_N\}$, $\mathbf{s}_i \in \mathbb{R}^{r \times 6}$. In our prototype studies, we take into account $r = 3$ statistical functions, i.e., *minimum*, *maximum*, and *average* of each of the 6 sensor axes in time series features, $\mathbf{t}_i$. Therefore, we obtain the statistical features of i-th rider maneuver as a $3 \times 6 = 18$ dimensional vector, i.e., $\mathbf{s}_i \in \mathbb{R}^{18}$.

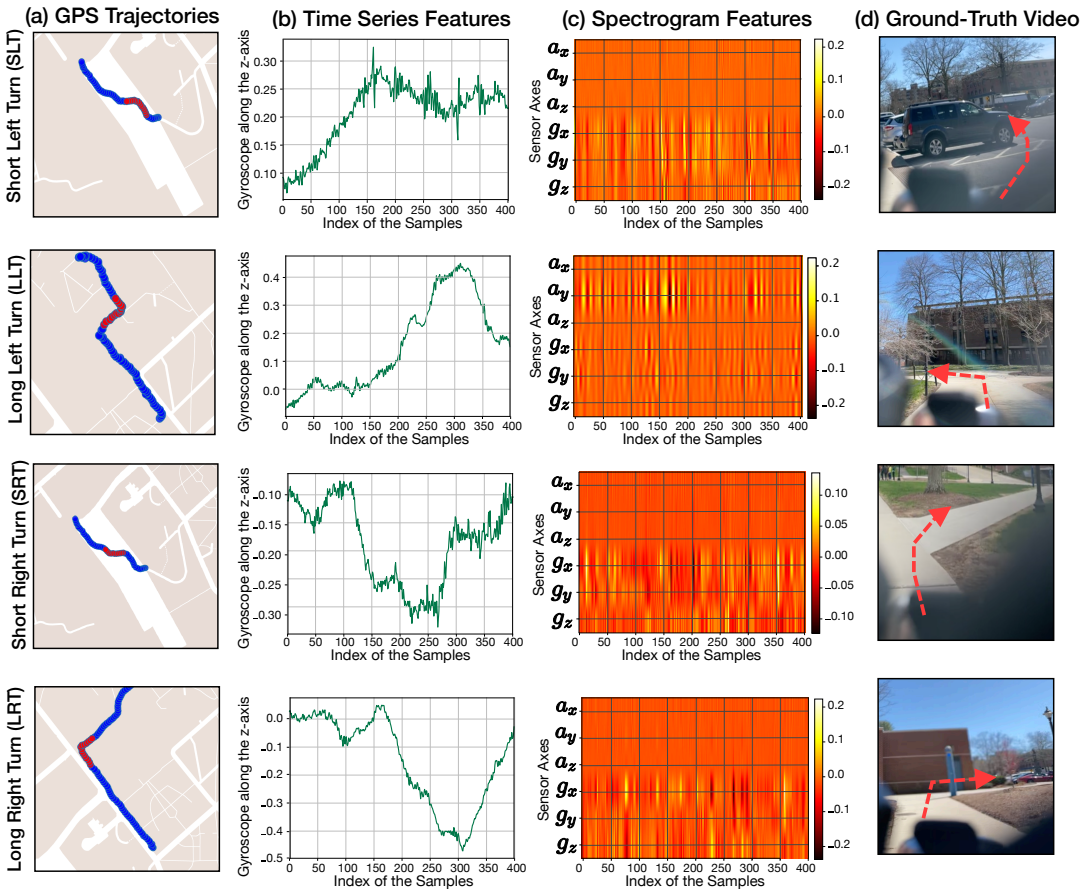

Fig. 4. Visualizations of the rider's *(a)* GPS trajectories; *(b)* time series; *(c)* spectrogram; and *(d)* ground-truth videos.

Problem Definition of FCRIL**:** Given the time series features, denoted as **T**, spectrogram features, denoted as **D**, and statistical features, denoted as **S**, of the rider maneuver interactions, the problem of FCRIL is to learn a function that maps the input rider maneuver features towards the corresponding rider maneuver class.

4 CONTRASTIVE RIDER INTERACTION LEARNING WITH MANEUVER RECOGNITION

We first overview the core model designs in Sec. 4.1. Then we detail the module designs of the rider interaction embedding learning in Sec. 4.2, and the rider interaction learning and maneuver recognition in Sec. 4.3. We further present the asynchronous federated learning designs in Sec. 4.4.

4.1 Core Model Overview

We overview the core model architecture of FCRIL in Fig. 5, which consists of the following three important modules.

(i) Rider Interaction Embedding Learning (RIEL): We first train the Rider Interaction Embedding Learning (RIEL in Sec. 4.2) module based on a novel supervised contrastive loss function design. Specifically, given the time series features $\mathbf{t}_i$, spectrogram features $\mathbf{d}_i$, and statistical features, $\mathbf{s}_i$ of the i-th e-scooter rider maneuver

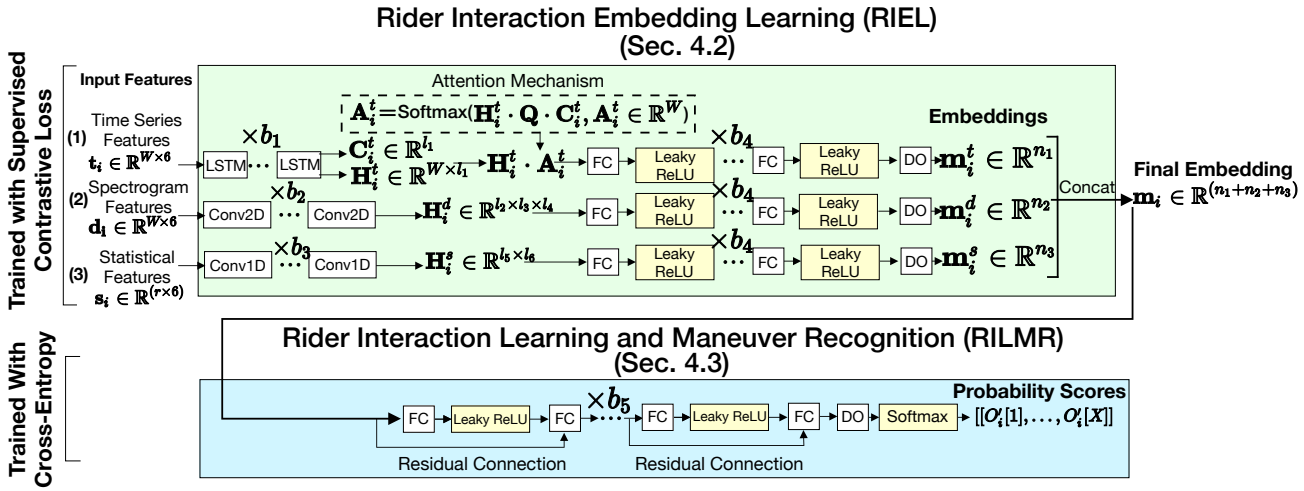

Fig. 5. Overall network structure of FCRIL for our RIL, which consists of RIEL and RILMR.

record, RIEL first extracts temporal features from $\mathbf{t}_i$ with an attention-based long short-term memory (LSTM) mechanism [33], and produces the embeddings $\mathbf{m}_i^t$. RIEL also extracts more complex patterns from $\mathbf{d}_i$ based on the 2D convolutional neural networks (Conv2Ds) and generates the embeddings $\mathbf{m}_i^d$. In the meantime, RIEL processes the statistical features based on the 1D convolutional neural networks (Conv1Ds) and outputs the embeddings $\mathbf{m}_i^s$. We consider the concatenation of all the embeddings $\mathbf{m}_i = \{\mathbf{m}_i^t, \mathbf{m}_i^d, \mathbf{m}_i^s\}$ as the final embedding of the i-th e-scooter rider maneuver record.

(ii) Rider Interaction Learning and Maneuver Recognition (RILMR): Given the differentiated embeddings from the RIEL module, we further classify the rider maneuvers based on the Rider Interaction Learning and Maneuver Recognition (RILMR in Sec. 4.3) module. To ensure the modules of RIEL and RILMR do not interfere with each other due to their different optimization settings during the training process, we train RIEL and RILMR in a sequential manner. In other words, we first train RIEL, and then freeze its weight parameters to further refine RILMR. RILMR further processes the output embeddings of the RIEL and generates the class probabilities for each rider maneuver class with the Softmax function based on the cross-entropy loss function.

(iii) Asynchronous Federated Maneuver Learning (AFML): To take advantage of the computation power of the edge on-board devices, we train FCRIL based on our proposed Asynchronous Federated Maneuver Learning (AFML in Sec. 4.4) algorithm, where each rider's mobile device (e.g., a smartphone or an on-board computer attached with an e-scooter) is not required to perform the training in a synchronized procedure along with the other e-scooter riders, thus increasing the overall scalability of the FCRIL framework in complex urban environments.

4.2 Rider Interaction Embedding Learning (RIEL)

Design Motivations: The goal of the RIEL module is to encode, generate, and associate the *rider interaction embeddings*, i.e., the encoded features of rider maneuvers, for different rider maneuver classes (e.g., the left/right turns) in order to enhance differentiation and identification accuracy. Specifically, based on the input features of the rider behavior data (e.g., time series, spectrogram, and statistical features in our studies), we have designed within RIEL a supervised contrastive learning loss function to produce the maneuver embeddings and encode the features. In RIEL, our supervised contrastive learning ensures that the resulting embeddings of the same rider maneuver class are *close* together within the rider interaction embedding space, while they are *distant* from those

of other classes. This optimization property of our supervised contrastive loss designs is crucial for realizing accurate RIL as traditional embedding learning methods [17, 27] may not effectively capture and generate the embeddings to maintain accurate identification results given the complex and noisy NR data.

Contrastive Learning Feature Processing Designs: Given the time series, spectrogram, and statistical features of the i-th e-scooter rider maneuver record, i.e., $\mathbf{t}_i$, $\mathbf{d}_i$, and $\mathbf{s}_i$, RIEL generates the embedding of the i-th e-scooter rider maneuver, $\mathbf{m}_i = \{\mathbf{m}_i^t, \mathbf{m}_i^d, \mathbf{m}_i^s\}$, via the following three steps.

(1) Time Series Features **T**. RIEL processes the time series features $\mathbf{t}_i \in \mathbb{R}^{W\times 6}$ with b_1 consecutive long short-term memory (LSTM) layers with l_1 units to capture the temporal dependencies, i.e.,

$$\mathbf{H}_i^t, \mathbf{C}_i^t = \mathtt{LSTM}_{b_1}\left(\mathtt{LSTM}_{b_1-1}\left(\ldots\left(\mathtt{LSTM}_1(\mathbf{t}_i)\right)\right)\right), \tag{3}$$

and produces the hidden states, denoted as $\mathbf{H}_i^t \in \mathbb{R}^{W\times l_1}$, as well as the cell states, denoted as $\mathbf{C}_i^t \in \mathbb{R}^{l_1}$, that encode the long-term memory information, and W is the length of the input time series. Then, with a trainable weight matrix $\mathbf{Q} \in \mathbb{R}^{l_1\times l_1}$, RIEL learns the similarity of the hidden states, $\mathbf{H}_i^t$, and the cell states, $\mathbf{C}_i^t$, as the attention scores. Let $A_i[j]$ be the attention score for the j-th element in the i-th time series, $\mathbf{t}_i$. Then, we form the attention scores for the time series features, denoted as $\mathbf{A}_i^t$, i.e.,

$$\mathbf{A}_i^t = [A_i[1], A_i[2], \ldots, A_i[W]] \triangleq \mathbf{H}_i^t \cdot \mathbf{Q} \cdot \mathbf{C}_\mathbf{i}^\mathbf{t}, \tag{4}$$

where each attention score $A_i[j]$ will be normalized and processed by a Softmax function, i.e.,

$$A_i[j] = \mathtt{Softmax}\left(A_i[j]\right) \triangleq \frac{\exp\left(A_i[j]\right)}{\sum_{k=1}^{W}\exp\left(A_i[k]\right)}. \tag{5}$$

We note that, unlike the prior studies [8, 45], our RIEL module does not leverage the non-linear activation functions to find the attention scores (e.g., Tanh or Sigmoid). This way, we can mitigate the potentially noisy activation that may impact the learning quality of rider interaction embeddings. Next, we multiply the output of the LSTM layers, $\mathbf{H}_i^t$, by the attention scores, $\mathbf{A}_i^t$, and we obtain the updated hidden states $\mathbf{H}_i'^t \in \mathbb{R}^W$, i.e.,

$$\mathbf{H}_i'^t = \mathbf{H}_i^t \odot \mathbf{A}_i^t. \tag{6}$$

This way, the RIEL model learns to identify and focus on the most important parts of the input time series. Next, $\mathbf{H}_i'^t$ is further processed by b_4 consecutive fully connected layers (denoted as FC) with n_1 neurons and the leaky ReLU (denoted as LR) activation function, followed by a Dropout (denoted as DO) layer [15] for regularization. Then FCRIL produces the embedding of the time series features for the i-th e-scooter rider maneuver, $\mathbf{m}_i^t \in \mathbb{R}^{n_1}$, i.e.,

$$\mathbf{m}_i^t = \mathtt{DO}(\mathtt{LR}_{b_4}(\mathtt{FC}_{b_4}(\ldots(\mathtt{LR}_1(\mathtt{FC}_1(\mathbf{H}_i'^t)))))). \tag{7}$$

(2) Spectrogram Features **D**. Next, given the spectrogram features of the i-th rider maneuver, $\mathbf{d}_i \in \mathbb{R}^{W\times 6}$, RIEL processes them with b_2 consecutive 2D convolutional layers (denoted as Conv2D) with l_4 filters to get $\mathbf{H}_i^d \in \mathbb{R}^{l_2\times l_3\times l_4}$, where l_2 and l_3 are the height and width of the features after the convolutional layers, i.e.,

$$\mathbf{H}_i^d = \mathtt{Conv2D}_{b_2}\left(\mathtt{Conv2D}_{b_2-1}\left(\ldots\left(\mathtt{Conv2D}_1(\mathbf{d}_i)\right)\right)\right). \tag{8}$$

Similarly, RIEL processes $\mathbf{H}_i^d$ using b_4 consecutive FC layers with b_2 neurons and the LR activation function followed by a DO layer to produce the spectrogram features' embedding of the i-th rider maneuver record, denoted as $\mathbf{m}_i^d \in \mathbb{R}^{n_2}$, i.e.,

$$\mathbf{m}_i^d = \mathtt{DO}(\mathtt{LR}_{b_4}(\mathtt{FC}_{b_4}(\ldots(\mathtt{LR}_1(\mathtt{FC}_1(\mathbf{H}_i^d)))))). \tag{9}$$

(3) Statistical Features **S**. RIEL further processes the statistical features of the i-th rider maneuver, $\mathbf{s}_i$, with b_3 consecutive 1-D convolutional layers (denoted as Conv1D) with l_6 filters and obtains $\mathbf{H}_i^s \in \mathbb{R}^{l_5\times l_6}$, i.e.,

$$\mathbf{H}_i^s = \mathtt{Conv1D}_{b_3}(\ldots(\mathtt{Conv1D}_1(\mathbf{s}_i))), \tag{10}$$

where l_5 is the size of the features after the convolutional layers. The output of the 1-D convolutional layers, $\mathbf{H}_i^s$, is further processed by the b_4 consecutive FC layers with n_3 neurons to get the embedding of the statistical features of the i-th e-scooter rider maneuver, $\mathbf{m}_i^s \in \mathbb{R}^{n_3}$, i.e.,

$$\mathbf{m}_i^s = \mathtt{DO}\left(\mathtt{LR}_{b_4}\left(\mathtt{FC}_{b_4}\left(\mathtt{LR}_{b_4-1}\left(\mathtt{FC}_{b_4-1}\left(\ldots\left(\mathtt{LR}_1(\mathtt{FC}_1(\mathbf{H}_i^s))\right)\right)\right)\right)\right)\right). \tag{11}$$

Finally, to produce the final embedding of the i-th rider maneuver, RIEL concatenates (the Concat operation) the three embeddings to get $\mathbf{m}_i \in \mathbb{R}^{(n_1+n_2+n_3)}$, i.e.,

$$\mathbf{m}_i = \mathtt{Concat}(\mathbf{m}_i^t, \mathbf{m}_i^d, \mathbf{m}_i^s). \tag{12}$$

Contrastive Learning Loss Function Design: In order to train RIEL with enhanced differentiation across rider maneuver classes, we have designed a supervised contrastive loss function for FCRIL. We illustrate the embedding learning process of RIEL in Fig. 6.

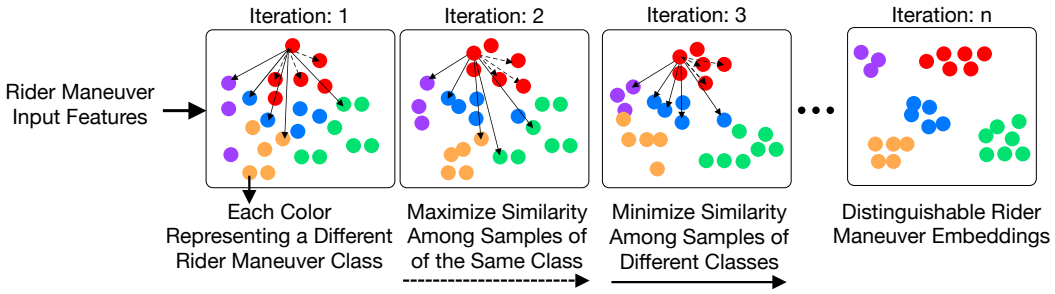

Fig. 6. Illustration of our supervised contrastive learning. With the iterations of learning, our RIL aims at: *(i)* differentiating maneuver embeddings across different classes; *(ii)* ensuring similar embeddings are generated within each class.

In particular, our supervised contrastive loss function leverages the provided ground-truth maneuver labels (e.g., left turns, right turns) to *iteratively* maximize the similarity of the current generated embeddings of the rider maneuvers that belong to the same class (e.g., two left turns). Taking the rider maneuvers in red in Fig. 6 as examples, the goal of RIEL is to maximize the similarity of the generated embeddings within this maneuver class. In the meantime, the contrastive loss function also aims to minimize the similarity of the embeddings for rider maneuvers that belong to different classes (e.g., to make the rider maneuver embeddings in red differ from the blue or green ones in Fig. 6). Furthermore, during the optimization, the RIL model parameters may be "penalized" given dissimilar embeddings within the same rider maneuver class. This way, we fill the essential gap of the existing cross-entropy loss function for the classification and improve the model adaptivity when handling complex rider interactions.

In particular, let $\mathbf{I}$ be the set of all the rider maneuvers. Given the embeddings $\mathbf{m}_i$ of the i-th maneuver, let $\mathbf{U}(i) = \mathbf{I} \setminus \{i\}$, and let $\mathbf{P}(i)$ be the set of all rider maneuvers with the same label as the i-th maneuver (i.e., the positives). Then, the supervised contrastive loss $\mathcal{L}^{\mathrm{SC}}$ is given by:

$$\mathcal{L}^{\mathrm{SC}} = \sum_{i\in\mathbf{I}} \mathcal{L}_i^{\mathrm{SC}} \triangleq \sum_{i\in\mathbf{I}} \frac{-1}{|\mathbf{P}(i)|} \sum_{p\in\mathbf{P}(i)} \log \frac{\exp\left(\mathbf{m}_i \cdot \mathbf{m}_p/\tau\right)}{\sum_{a\in\mathbf{U}(i)} \exp\left(\mathbf{m}_i \cdot \mathbf{m}_a/\tau\right)}, \tag{13}$$

where τ represents the penalty parameter. We note that lower values of τ can encourage model training and optimization towards the negative pairs of maneuvers that are hard to differentiate. Specifically, the denominator in Eq. (13) calculates the similarity of each rider maneuver to the others within the same rider maneuver class based on the dot product of their corresponding embeddings. On the other hand, the nominator computes

the similarity between each rider maneuver and the others regardless of their class. This way, minimizing the loss function in Eq. (13) can help produce the maneuver embeddings with our desired properties for RIL, i.e., embeddings of different classes that can be differentiated from each other.

4.3 Rider Interaction Learning and Maneuver Recognition (RILMR)

Design Motivations: The role of the rider interaction learning and maneuver recognition (RILMR) module is to further train a classifier given the embeddings learned by the previous RIEL and yield the recognition results of the rider maneuvers. Inside the RILMR module, we further design the residual layers in order to overcome the model over-fitting issues and further allow the model to determine the required complexity for this module. As discussed in Sec. 4.1, we train the RIEL and RILMR modules in a sequential manner to ensure the optimization processes of RIEL and RILMR do not interfere with each other.

Detailed Designs: The weights of RIEL are frozen (i.e., not updated anymore) after the training, and its output is then connected to RILMR module for further training with the cross-entropy loss for the rider maneuver identification. In particular, let $\mathbf{m}_i \in \mathbb{R}^{(n_1+n_2+n_3)}$ be the output embedding of RIEL. RILMR will first process $\mathbf{m}_i$ with the b_5 consecutive FC layers with n_4 neurons and the LR activation function to obtain $\mathbf{O}_i \in \mathbb{R}^{n_4}$, i.e.,

$$\mathbf{O}_i = \mathtt{DO}(\mathtt{LR}_{b_5}(\mathtt{FC}_{b_5}(\ldots(\mathtt{LR}_1(\mathtt{FC}_1(\mathbf{m}_i)))))), \tag{14}$$

where we adopt residual connections [56] between every two consecutive FC layers to enhance the gradient flows in model training. Then, the RILMR module further processes $\mathbf{O}_i$ with an additional FC layer with X neurons (corresponding to the number of the rider maneuver classes; $X = 7$ in our studies). Let $\mathbf{O}'_i$ be the output of the FC layer that further processes $\mathbf{O}_i$, and $O'_i[j]$ be its j-th element that corresponds to the j-th rider maneuver class ($j \in \{1, \ldots, X\}$), i.e.,

$$\mathbf{O}'_i = \left[O'_i[1], \ldots, O'_i[X]\right] = \mathtt{FC}\left(\mathbf{O}_i\right). \tag{15}$$

Then, we have the Softmax function on each element $O'_i[j]$ of the output, i.e.,

$$O'_i[j] = \mathtt{Softmax}\left(O'_i[j]\right) \triangleq \frac{\exp\left(O'_i[j]\right)}{\sum_{k=1}^{X} \exp\left(O'_i[k]\right)}, \quad j \in \{1, \ldots, X\}. \tag{16}$$

Let $\mathbf{p}_i = [p_i[1], \ldots, p_i[X]]$ be the one-hot encoding of the true label for the i-th rider maneuver record (for instance, we let $[1, 0, \ldots, 0] \in \mathbb{R}^X$ be a maneuver of short left turn), and N be the total number of the rider maneuvers. Then, the cross-entropy loss function, denoted as C_{loss}, of our RILMR module is given by

$$C_{\text{loss}} \triangleq -\sum_{i=1}^{N}\sum_{j=1}^{X} p_i[j] \cdot \log\left(O'_i[j]\right), \tag{17}$$

which is used to train and optimize the RILMR module.

4.4 Asynchronous Federated Maneuver Learning (AFML) Design

Design Motivations: Towards the ubiquitous RIL, one may consider uploading the sensor data (e.g., IMU data in our case) harvested by the e-scooter riders (e.g., through smartphones or on-board wireless connectivities) to the cloud server for large-scale and centralized RIL model training, which, however, may have rider data privacy concerns (e.g., regarding the trustworthiness of the RIL system), discouraging the riders from participating in the RIL that may potentially benefit their riding safety or enhance their riding experience (for instance, for the e-scooter sharing service).

Furthermore, despite the conventional federated learning mechanisms for less sensitive model parameters (e.g., gradients) [34], it is crucial for emerging smart e-scooters in complex urban environments to reduce the communication overhead and delays, while the spatial distributions and communication connectivities of the

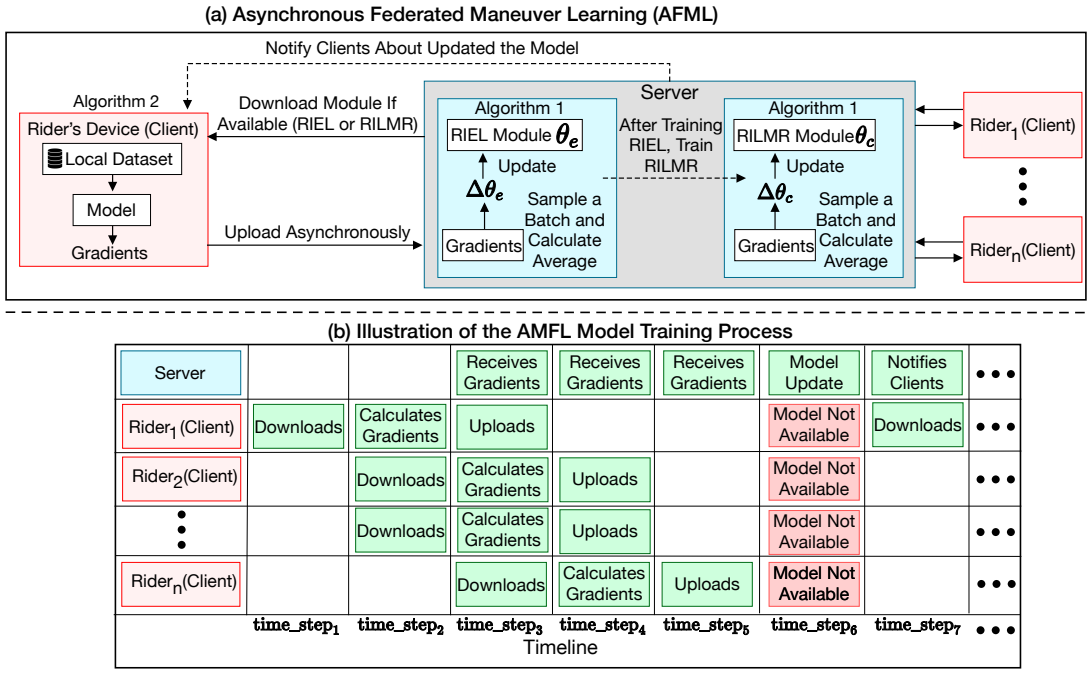

Fig. 7. Our AFML designs: (a) overview of AFML; and (b) illustration of the training process.

smart e-scooters are highly dynamic, making the synchronized model training across multiple devices impractical. It is highly challenging to achieve fast convergence of the global RIL model, and conventional federated learning methods might be unscalable for large-scale ubiquitous and urban computing scenarios.

Therefore, to meet these practical needs of e-scooter manufacturers and sharing service operators, we design within our FCRIL, a novel Asynchronous Federated Maneuver Learning (AFML) to train the core RIL model. We illustrate the overall workflow of our proposed AFML algorithm in Fig. 7(a). In particular, in the early stages of the algorithm, the client side (e.g., the rider's smartphone or the smart e-scooter with computing capability), first downloads the latest global parameters of RIEL module. Then, it calculates the gradients of its parameters based on its local maneuver dataset and uploads the gradients to the RIL server asynchronously.

On the other hand, on the server side, the RIL server is asynchronously waiting for enough gradients to be accumulated. Once enough gradients are available, the RIL server samples batches of gradients and averages each batch to update the RIEL parameters. We note that, while the RIL server is training the FCRIL model, the clients do not download the FCRIL model. We also note that in order to ensure both modules do not interfere with each other, after the training procedure is completed for the RIEL module, we cease its updates and freeze its weights, and start the training procedure of the RILMR module, which has similar steps as stated above.

Detailed Designs: We present the AFML procedures as follows.

(1) RIEL *Module Instance Training at the* RIL *Server Side*: We first create an instance of RIEL module parameterized by θ_e. Then, we execute the Algorithm 1 with the following parameters as its input to train RIEL module for module parameters $\theta = \theta_e$, i.e., number of the communication rounds, $i_{\text{round}} = i_e$; learning rate $\lambda = \lambda_e$; and gradients batch size, $G = G_e$. As illustrated in Algorithm 1, the RIL server maintains a list of *gradients*, denoted as `total_gradients`, which are gradients *asynchronously* transmitted from the clients (e.g., a smartphone or an

embedded IMU sensor). AFML first continuously checks the available gradients (Line 8). If the harvested gradients are enough for RIEL training, the RIL server samples the batches of gradients with a batch size of G_e from the total_gradients list, and updates the parameters of RIEL, θ_e, using the average of the sampled gradients (Lines 11–15 in Algorithm 1). The algorithm updates the parameters for i_{inner} iterations by sampling batches of the gradients, for which λ_e is used as the learning rate.

We note that when performing parameter updates, the RIL server freezes the RIEL module parameters, θ_e and prevents devices from downloading them (Line 11 in Algorithm 1) to ensure that the mobile devices (on the smart e-scooters or with the rider) would use the latest parameters for the local training on the client side. Besides, to prevent the algorithm from updating the modules using outdated gradients, total_gradients is cleared after each parameter update (Line 18 in Algorithm 1). Additionally, to avoid unnecessary computations, before updating the module, the clients that are currently performing module training are notified to stop their training as their module is an outdated version of the global parameters (Line 9 in Algorithm 1). The entire procedure above is repeated for i_e communication rounds to complete the training of RIEL.

Algorithm 1: AFML on the server side (for RIEL or RILMR).

```
1  Input: Number of the communication rounds i_round; learning rate λ; gradients batch size, G.
2  Output: Learned global parameters θ (either RIEL or RILMR).
3  current_devices ← list of clients (e.g., a smartphone) currently performing model training, which is maintained
     asynchronously.
4  θ ← initialize module parameters uniformly;
5  module_available ← true;
   // List of gradients received from different clients that is updated asynchronously.
6  total_gradients ← [];
7  while i ≤ i_round do
       // Perform an update only if enough gradients are received from different clients.
8      if enough gradients are available in total_gradients then
9          Notify current_devices to stop training;
10         current_devices ← [];
11         module_available ← False;
12         for j ← 1 to i_inner do
13             grads ← sample G records of gradients from total_gradients;
14             Δθ ← calculate average of grads;
15             θ ← θ − λ × Δθ;
16         i ← i + 1;
17         module_available ← True;
18         total_gradients ← [];
19 return θ;
```

(2) RILMR *Module Instance Training at the* RIL *Server Side*: Recall in Sec. 4.1, once the parameters of RIEL are updated and trained (say, for i_e communication rounds), we freeze the RIEL parameters and continue to train RILMR based on a similar procedure in Algorithm 1. Specifically, we create an instance of the RILMR module parameterized by $\theta = \theta_c$. We set the number of the communication rounds $i_{\text{round}} = i_c$; learning rate $\lambda = \lambda_c$; and gradients batch size $G = G_c$.

(3) Local Training on the Client Side: As illustrated in Algorithm 2, AFML first determines if the parameters of the FCRIL model are available for the download from the RIL server (Line 2). Then, AFML calculates the gradients

of the parameters of either the RIEL module, denoted as θ_e (Lines 4–6), according to the supervised contrastive loss in Eq. (13), or the RILMR module, denoted as θ_c (Lines 12–13), based on the cross-entropy loss in Eq. (17), respectively. Afterwards, the calculated gradients are uploaded to the RIL server for later aggregation. We note that if the algorithm is training the RILMR module, it also downloads the latest parameters of the RIEL module, θ_e (Line 9), freezes them (Line 10), and connects the output layer of the RIEL module (Eq. (12)) to the input layer (Eq. (14)) of the RILMR (i.e., feeding the embeddings of the local dataset to the RILMR module in Line 11).

Algorithm 2: AFML on the client side for the local FCRIL model training.

```
1  while True do
2      if module_available then
           // Check with the server if the RIL model (either RIEL or RILMR) is available to
              download.
3          if training_RIEL then
               // If yes, we train the RIEL module, and the RILMR module otherwise.
4              θ_e ← Download latest RIEL module parameters;
5              Δθ_e, L_1 ← Supervised contrastive loss according to Eq. (13) using θ_e and the local data;
6              Upload Δθ_e to the server (to be added to the total_gradients in Algorithm 1);
7          else
8              θ_c ← Download the latest RILMR module parameters;
9              θ_e ← Download the latest RIEL module parameters;
10             Freeze RIEL parameters θ_e;
11             Connect the output layer of RIEL in Eq. (12) (with θ_e) to the input layer of RILMR in Eq. (14) (with θ_c);
12             Δθ_c, L_2 ← Cross-entropy loss according to Eq. (17) using θ_c and the local data;
13             Upload Δθ_c to the server (to be added to the total_gradients in Algorithm 1);
```

We further illustrate an example of the workflow of AFML in Fig. 7(b). Specifically, the clients asynchronously download the module parameters (either RIEL or RILMR), calculate the gradients, and upload them to the RIL server. In the meantime, the RIL server is receiving the gradients asynchronously. Next, at the time_step$_5$, the RIL server has received enough gradients and therefore at time_step$_6$, it starts updating the RIL model, during which the the clients do not have access to the FCRIL model parameters.

5 EXPERIMENTAL EVALUATIONS

We first present the experimental settings in Sec. 5.1, and then show the experimental results in Sec. 5.2.

5.1 Experimental Settings

• **Baselines:** We compare our FCRIL model framework with the following baseline approaches and models. We note that these baseline approaches considered are capable of processing the time series or image features (i.e., the spectrogram features), and their comparison with FCRIL helps validate the effectiveness and accuracy of our proposed designs.

(1) – (3) LSTM, GRU, and RNN: which respectively use stacked long short-term memory (LSTM) [30, 51], gated recurrent unit (GRU), and recurrent neural network (RNN) to learn the time series features **T** of the rider maneuvers.

(4) BiLSTM: which leverages the bidirectional LSTM (BiLSTM) layers [46] to process the time series features **T** of rider maneuvers.

(5) ATTLSTM: which leverages the attention-based LSTM [8] to process the time series features **T**.
(6) Conv2D-**T**: which leverages 2D convolutional neural networks (Conv2D) [31] to process time series features **T** of the rider maneuvers.
(7) Conv2D-**D**: which uses the Conv2D to process the spectrogram features **D** of the rider maneuvers.
(8) Conv1D-**T**: which uses the 1D convolutional neural networks (Conv1D) and takes in only the time series features **T** of the rider maneuvers.
(9) ResNet-**T**: which uses the residual neural network (ResNet) [18, 22] to take in the time series features **T** of the rider maneuvers.
(10) ResNet-**D**: which implements the ResNet architecture [18, 22] that processes the spectrogram features **D** of the rider maneuvers.
(11) DenseNet-**D**: which leverages the dense connections between layers, i.e., the DenseNet [23, 57], to process the spectrogram features of the rider maneuvers.
(12) DLSTM-Conv: which implements a denoising LSTM-based auto-encoder to remove the input noise. The filtered data are further processed by Conv2D layers with wide kernels along the time axis [45].
(13) TST: which implements a time series feature classification framework [54] based on the encoder designs adapted from the transformer architectures [49].
(14) SVM: which applies the support vector machine (SVM), a traditional machine learning algorithm, to process the statistical features **S** of the rider maneuvers.
(15) GBDT: which leverages the gradient boosting decision tree (GBDT) as another statistical machine learning to process the statistical features **S** of the rider maneuvers.

Parameter and Evaluation Settings: In our *data preprocessing* stage, we leverage a sliding window of size 10 seconds with an overlap of 50%. We note that before the sliding window is applied, the time series data from all the sensors are re-sampled to have a sampling frequency of 40 Hz, leading to records with 10 seconds $\times$ 40 Hz = 400 samples. Furthermore, we have used the moving average method (i.e., using the 150 values preceding the current one) to filter the noise.

To train our RIEL and RILMR modules, we set the penalty parameter τ in Eq. (13) to 0.025 to ensure that the resultant rider maneuver embeddings across different classes are differentiated. We use one layer of LSTM, Conv1D, and Conv2D in RIEL ($b_1 = b_2 = b_3 = 1$ in Eqs. (3), (8), and (10)). RIEL leverages $b_4 = 3$ (in Eqs. (7), (9)), and (11)) consecutive FC layers with LR activation function to produce the embeddings. Also, RILMR uses $b_5 = 6$ FC layers with residual connections and $n_4 = 64$ neurons to further process the embeddings of the rider maneuvers (Eq. (14)). We set the output embedding sizes for the time series, spectrogram, and statistical features to $n_1 = n_2 = n_3 = 64$ (Eqs. (7), (11)), (9)). Furthermore, for RIEL module, we use $l_1 = 16$ (Eq. (3)) units for its LSTM layers, and $l_4 = l_6 = 32$ filters for the convolutional layers with (3,3) and 2 as the kernel sizes for the 2D and 1D convolutional layers, respectively (Eqs. (8) and (10)). We use a dropout rate of 20% and 10% for all the Dropout (DO) layers in the RIEL and RILMR modules, respectively.

For the AFML studies (Algorithms 1 and 2), we respectively train RIEL and RILMR with $i_e = 35$ and $i_c = 10$ communication rounds, and therefore we have a total of 45 communications rounds. Furthermore, we empirically set $G_c = G_e = 5$ to sample from the gradients list to perform $i_{\texttt{inner}} = 5$ module parameter updates. We use the learning rates, $\lambda_e = 0.01$ and $\lambda_c = 0.0005$, to train RIEL and RILMR, respectively. To simulate the federated learning settings, we divide the NR data collected with each smartphone or IMU sensor into morning and evening and consider them as different clients.

The parameter settings of our baselines are as follows. For (1)–(5), we use two recurrent layers with 64 neurons. For (6)–(8), we leverage three convolutional layers, and for (9) and (10), we implement the first two stages of the ResNet architecture [18]. For (11), we adopt DenseNet with a growth rate of 5, 32 filters, and 3 blocks of Dense-Transition blocks, each with 3 layers. In (12), we use 6 LSTM and 4 Conv2D layers. For (13), we leverage a

transformer encoder block with 4 attention heads, each with 64 units. For (14) we consider SVM with the radial basis function (RBF) kernel, and for (15) we consider GBDT with a total of 100 estimators and a maximum depth of 128.

To train our proposed FCRIL and other baseline approaches, we adopt the Adam [15] optimizer, and we leverage 70% of the data for training and 30% for evaluation. We train our models on an HPC deep learning server equipped with an AMD Ryzen Threadripper 3960X 24-Core Processor, 2× Quadro RTX 8000 48GB GDDR5 GPUs, 128GB RAM, and Linux Ubuntu 18.04.6 LTS. For our current prototype with the above settings, the average computation times (per rider maneuver record) for data preprocessing and feature extraction, model training, and prediction are 18.61ms, 342.85ms, and 0.1435ms, respectively. We have collected a total of 2,800 rider maneuver records, i.e., 630 from iPhone 13 Pro, 700 from iPhone 13 Mini, 770 from iPhone 12 Mini, 280 from iPhone 7 Plus, and 420 from our stand-alone IMU sensors. We focus on the smartphone IMU data for our model ablation studies, sensitivity studies, AFML convergence analysis, and rider generalization studies. We use *Accuracy* to measure the performance of our FCRIL and the baselines, i.e., the ratio of the correctly predicted maneuvers over the total number of the maneuvers.

5.2 Experimental Results

• **Overall Performance:** We first demonstrate the overall performance in RIL based on the rider maneuver classification accuracy. Here we train our FCRIL as well as the baselines based on a centralized learning method (i.e., considering the availability of all the data on the server) and the same loss function designs in order to study the learnability and identifiability improvements through our core designs. We also note that a centralized learning study will help us compare FCRIL with the conventional machine learning approaches (SVM and GBDT in our studies) that may be hard to be adapted for a federated learning setting. For each approach, we run the experimental studies 5 times and report the average accuracy as well as the standard deviation.

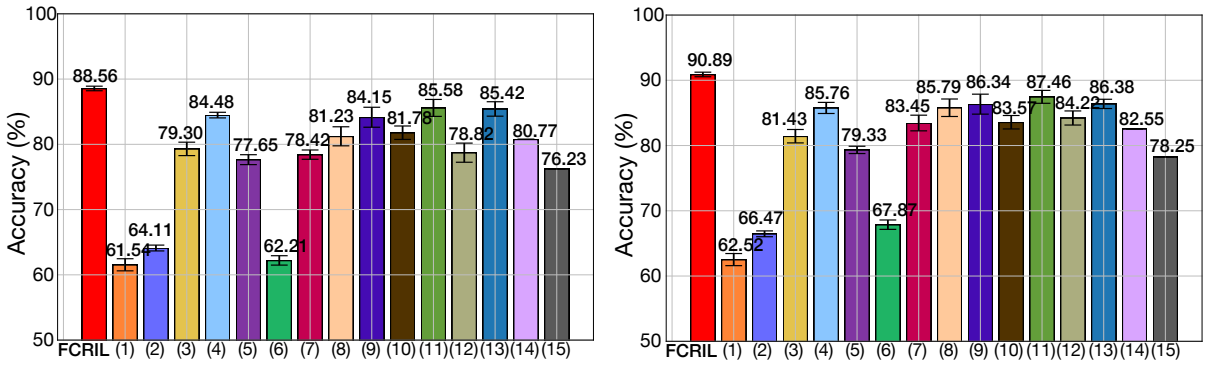

Fig. 8. Overall accuracy (%) of FCRIL and the baseline approaches for RIL based on (a) the stand-alone IMU data; and (b) the smartphone IMU data (centralized learning settings).

Specifically, we illustrate the results of the RIL accuracy in Figs. 8(a) and (b), where the RIL results are based on the stand-alone IMU sensor and smartphone IMU measurements, respectively. We can first see that the RNN (scheme (3)) outperforms the baselines based on LSTM and GRU (schemes (1) and (2)), indicating that only using the gated mechanism of the LSTM and GRU may not necessarily capture the temporal dependencies in the complex rider maneuvers. The bidirectional time series processing of BiLSTM (scheme (4)) and the attention mechanism in ATTLSTM (scheme (5)) focus on the essential temporal information of the input features and hence enhance their performance from the LSTM and GRU. However, we also observe that BiLSTM and ATTLSTM do not achieve

satisfactory RIL accuracy since they only capture the temporal dependencies across the time series features of the rider maneuvers.

In our studies, Conv2D-**D**, ResNet-**D**, and DenseNet-**D** (schemes (7), (10), and (11)) are shown to achieve overall better performance than the above-mentioned BiLSTM, ATTLSTM, LSTM, GRU, and RNN. This is mainly because these image-based approaches extract more comprehensive patterns from the spectrogram features and are less prone to the noisy rider maneuver data thanks to the convolutional and pooling operations. However, learning the spectrogram features through Conv2D-**D**, ResNet-**D**, and DenseNet-**D** may not necessarily encode the correct positions of important rider maneuver patterns within the spectrogram features, yielding degraded performance. DLSTM-Conv (scheme (12)) leverages the auto-encoders to filter the sensor data, but this approach cannot comprehensively capture the patterns within rider maneuvers due to its main use of the temporal information solely for denoising. The transformer-based encoder within TST (scheme (13)) performs better than DLSTM-Conv. However, TST only focuses on short-term dependencies and hence might not achieve satisfactory results like FCRIL. We can also see that the traditional machine learning algorithms like SVM and GBDT in our studies cannot fully capture the complex rider maneuver features due to their shallow learning structures, and hence achieve low RIL accuracy.

Compared with the aforementioned approaches, our FCRIL achieves a higher RIL accuracy due to our novel designs of rider interaction embedding learning (RIEL) that differentiates the feature embeddings generated from the time series, spectrogram, and statistical features. Furthermore, the rider interaction learning and maneuver recognition (RILMR) module further overcomes the model over-fitting issue through the inclusion of residual layers, yielding better accuracy than the other baseline approaches. Additionally, our FCRIL achieves comparable performance for the stand-alone IMU sensor and the smartphone IMU datasets, demonstrating the generalizability and ubiquitousness of our model designs. This will also help the smart micromobility manufacturers and sharing service operators in the micromobility sensing designs, such as using the low-cost IMU sensor installed on e-scooters, or prompting the riders to use their own smartphones for behavior analysis.

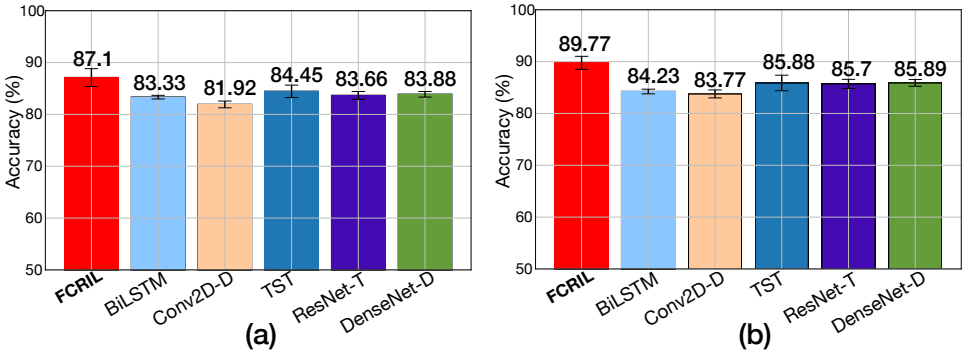

Fig. 9. Overall accuracy (%) of FCRIL and the selected baseline approaches for RIL based on (a) the stand-alone IMU data; and (b) the smartphone IMU data (federated learning settings).

With a federated learning setting in Fig. 9, we further compare FCRIL with five different selected deep learning baselines (with the best performance in Fig. 8). We can observe FCRIL achieves better accuracy compared with the other baseline approaches when integrated with federated learning settings of AFML, demonstrating the effectiveness of the entire framework designs of FCRIL. Furthermore, we can observe from both Figs. 9 and 8, FCRIL achieves comparable performance given both centralized learning (CL) and federated learning (FL) settings, demonstrating the effectiveness of the AFML module in FCRIL. We can also observe a slight increase of standard

deviation in terms of RIL classification accuracy compared with the centralized learning setting due to more stochastic processes of sampling batches of gradients (i.e., nondeterministic steps) in AFML. Despite the negligible performance degradation (say, up to 1.46% for FCRIL), the AFML module within FCRIL will bring the benefits in terms of model adaptivity and learning flexibility, as well as data privacy implications for RIL. In the real-world RIL scenarios, when the e-scooter riders' data may be opportunistically fed to the RIL server, our proposed AFML design can achieve an overall more consistent and robust model training performance than the centralized learning settings.

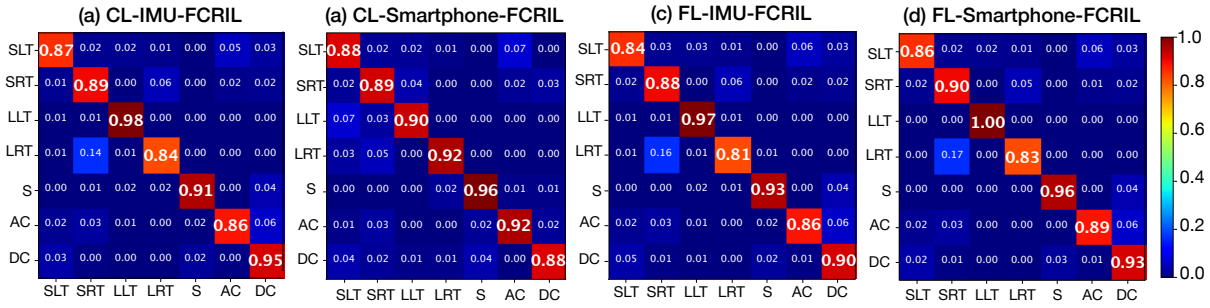

Fig. 10. (a)–(b) Confusion matrices of FCRIL based on the proposed federated learning algorithm (AFML); (c)–(d) FCRIL trained based on the centralized learning (CL) setting.

To show the more detailed classification results of our FCRIL, we have included the confusion matrices of RIL results of FCRIL in a centralized learning setting (Figs. 10(a) and (b)) and a federated learning setting with AFML (Figs. 10(c) and (d)). We can see that thanks to the comprehensive fusion of multiple features and the differentiated maneuver embeddings by our RIEL module, our FCRIL achieves high and consistent accuracy in classifying all the rider maneuver classes. In real-world RIL scenarios, such an accurate and consistent classification across all maneuver classes will help FCRIL achieve more robust and generalizable rider interaction understandings and enable more ubiquitous RIL applications in the NR scenarios.

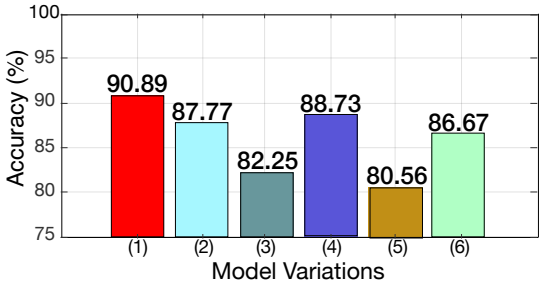

Fig. 11. Ablation studies performance results of FCRIL.

• **Model Ablation Studies:** Taking the smartphone data as an example, we further conduct the model ablation studies to demonstrate the relative importance of different core components of our FCRIL framework. Specifically, we have considered the following variations of our model: *(1)* the complete model, *(2)* without the time series features, **T**, *(3)* without the statistical features, **S**, *(4)* without the spectrogram features, **D**, *(5)* without using the

supervised contrastive loss function (i.e., using a cross-entropy loss function instead), and *(6)* without the residual layers in the RILMR module. We illustrate the results of model ablation studies in Fig. 11, from which we can see that the most significant performance degradation results from the removal of the supervised contrastive learning *(5)*, which demonstrates the importance of the supervised contrastive loss function compared with the traditional cross-entropy loss function in differentiating the rider maneuver embeddings. In addition, we can observe the considerable performance degradation due to the removal of each feature in *(2)–(4)*, indicating the importance of fusing comprehensive features for more accurate RIL. From the performance degradation of *(6)*, we can observe the importance of using the residual layers in the RILMR module (Sec. 4.3) in overcoming the model over-fitting.

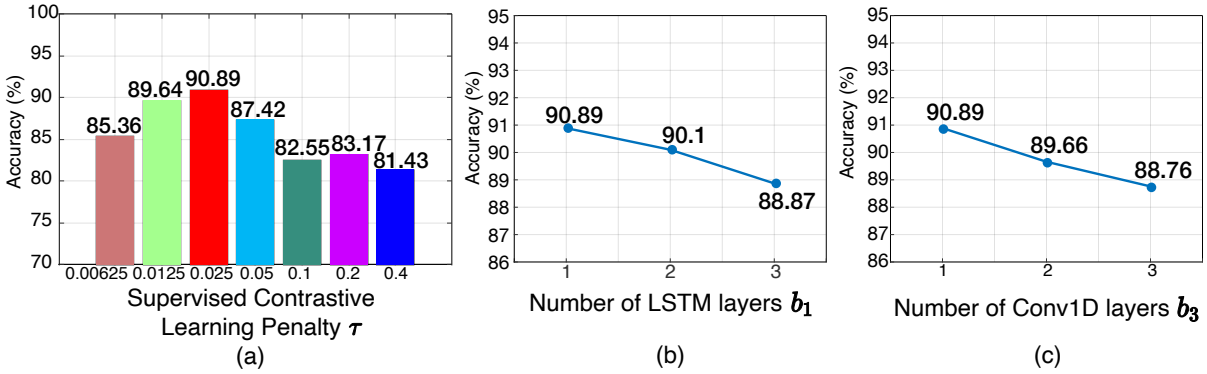

Fig. 12. Sensitivity studies of FCRIL: (a) penalty parameter τ; (b) number of LSTM layers b_1; and (c) number of Conv1D layers.

• **Model Sensitivity and Case Studies:** We have further evaluated the impacts of *(A)* important model parameters, *(B)* smartphone device types, *(C)* road types and conditions, and *(D)* IMU sensor placement positions. We perform the *(A)* and *(B)* studies based on the smartphone IMU data, and the *(C)* and *(D)* using the stand-alone IMU data.

(A) Model Parameters: We perform sensitivity studies regarding the following important model parameters: *(a)* the penalty parameter, τ, (Eq. (13)) for supervised contrastive learning; *(b)* the number of the LSTM layers in RIEL, denoted as b_1 (Eq. (3)); and *(c)* the number of the 1D convolutional layers in RIEL, denoted as b_3 (Eq. (10)). We demonstrate the result of the sensitivity studies in Fig. 12(a)–(c). From Fig. 12(a), we can see that the penalty parameter, τ, is important for the model performance. When τ is large (e.g., 0.4), the resulting rider maneuver embeddings may not be differentiated for accurate RIL as the RIEL module is not sensitive to the embedding differences across different rider maneuver classes. However, if τ is very small (e.g., 0.00625), RIEL may generate the rider interaction embeddings that are too distant from each other, and hence make it difficult to capture the common patterns within the embeddings. Furthermore, according to Figs. 12(b) and (c), we can see that increasing the number of the LSTM or Conv1D layers may complicate the RIL model architecture, leading to model over-fitting and performance degradation.

(B) Smartphone Devices: To study the impacts of different smartphones on the NR data collection, we further test FCRIL model (trained based on the data using iPhone 13 Pro) on the data harvested from *(a)* iPhone 13 Pro, *(b)* iPhone 13 Mini, *(c)* iPhone 12 Mini, and *(d)* iPhone 7 Plus. We can see from Fig. 13 that despite some performance differences due to the potential heterogeneity in the smartphone IMU chipsets, our FCRIL framework demonstrates robust performance and overall consistent accuracy across different smartphones, which corroborates the robustness and generalizability of our proposed model designs.

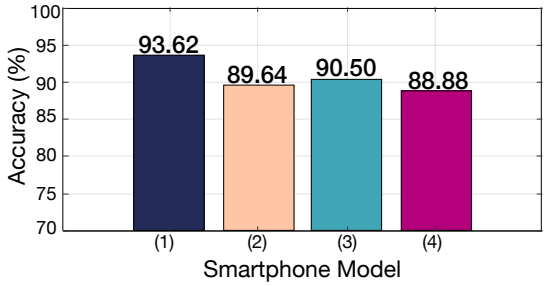

Fig. 13. Performance of FCRIL for different smartphones, i.e., (a) iPhone 13 Pro; (b) iPhone 13 Mini; (c) iPhone 12 Mini; and (d) iPhone 7 Plus.

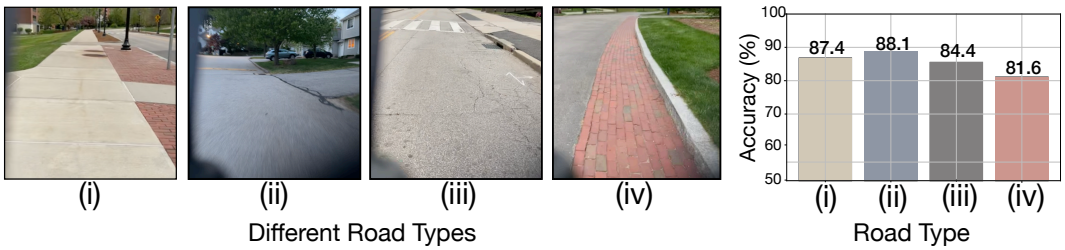

Fig. 14. *Left*: Different road types and conditions: *(i)* sidewalks; (ii) asphalt roads; (iii) concrete roads; and *(iv)* red bricks; *Right*: FCRIL's performance results for different road types.

(C) Road Types and Conditions: We collect the NR data from the road types of: *(i)* sidewalks, *(ii)* asphalt, *(iii)* concrete roads, and *(iv)* red bricks, as shown in Fig. 14 (left) with the stand-alone IMU sensor, and also illustrate the FCRIL's performance on the collected data in Fig. 14 (right). Since the uneven surfaces of the roads with red bricks may lead to noisier IMU readings than the others, one may observe a slight decrease of RIL accuracy. Despite the above, we can observe that FCRIL achieves overall high accuracy across different road types and conditions, demonstrating our model's adaptivity, ubiquitousness, and robustness.

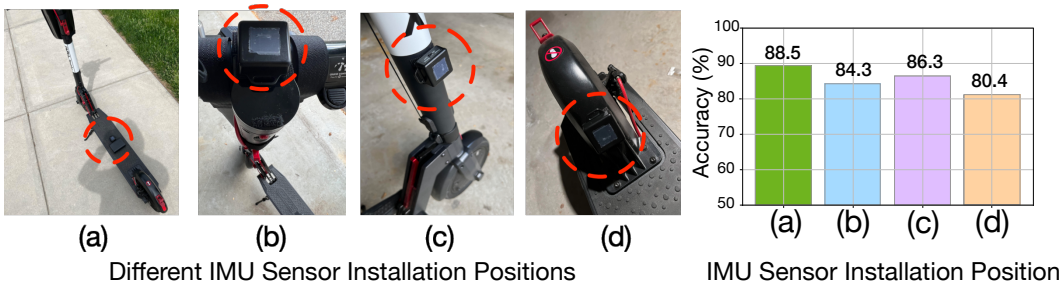

Fig. 15. *Left*: Different stand-alone IMU sensor installation positions, i.e., (a) the e-scooter board (our default setup), (b) on the top of the e-scooter's steering handle, (c) in the middle of the scooter bar, and (d) above the e-scooter rear end; *Right*: the resulting RIL performance.

(D) IMU Sensor Placement Position: As illustrated in Fig. 15(a)–(d) (left), we attach the stand-alone IMU sensor at different positions of the e-scooter, including: *(a)* the e-scooter board (our default setup); *(b)* on the top of the e-scooter's steering handle; *(c)* in the middle of the scooter bar; and *(d)* above the e-scooter rear end. We further

show in Fig. 15 (right) that the performance of our FCRIL model is overall robust with various stand-alone IMU sensor installation positions, while installation close to the rear end of the e-scooter (more vibrations) may not be recommended due to the noisier IMU measurements.

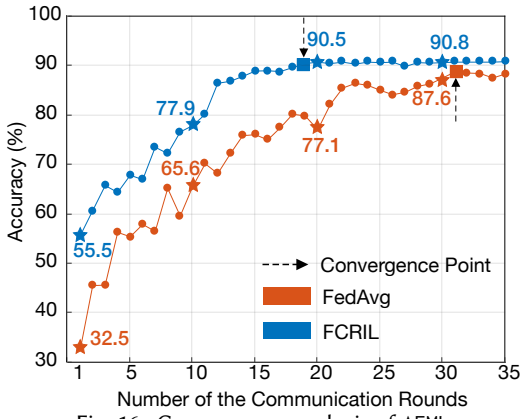

Fig. 16. Convergence analysis of AFML.

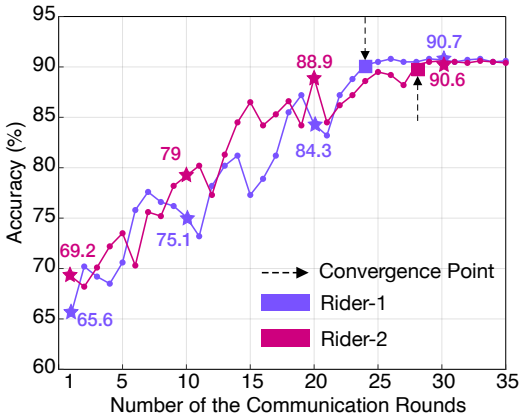

Fig. 17. AFML performance for two e-scooter riders.

• **Federated Learning Convergence Studies:** Furthermore, to investigate the performance of our proposed AFML algorithm, we have evaluated the number of communication rounds, denoted as i_c, needed (i.e., between clients and the server) to train RILMR after the RIEL is trained with the supervised contrastive loss (Eq. (13)). We can see in Fig. 16 that our model quickly converges (say, no further accuracy improvement higher than 0.5%) after about 19 communications rounds in our experimental studies, validating the effectiveness of our proposed AFML designs. We also compare AFML with FedAvg [34], which is a traditional *synchronized* federated learning algorithm that takes the average of the gradients (or model parameters) received from the clients to update the model. By showing the faster convergence (about 38% faster) than FedAvg, we also corroborate the efficiency and deployment potentials of our FCRIL framework in the complex urban environments for the e-scooter riders.

We also show the learning curve of our model for two e-scooter riders in our studies during the AFML process in Fig. 17. Since the riders may have different riding styles and use different devices to collect the NR data, we can see that their learning curves slightly oscillate after each round as the FCRIL is adapting to both riders. Despite this, after about 20 communication rounds, the learning curves stabilized and finally converged. We can also infer the similar RIL accuracy and the generalizability of FCRIL across different riders.

• **Result Visualization:** We visualize the embeddings learned by our FCRIL in Fig. 18(a) and when the supervised contrastive loss (Eq. (13)) is used to train RIEL, and we leverage principal component analysis (PCA) for dimensionality reduction and visualization. Fig. 18(a) illustrates the overall distinguishable rider maneuver classes, i.e., the resulting rider maneuvers embeddings belonging to different classes are distant from each other while those belonging to the same class are close. Furthermore, we can observe that similar maneuver classes such as long and short left or right turns are also differentiated from each other. As a comparison, as shown in Fig. 18(b), the rider interaction embeddings of different classes based on the cross-entropy loss function are not well distinguishable from each other, which validates the need to incorporate supervised contrastive learning within our FCRIL. We also note that the sequential learning of RIEL and RILMR modules help further differentiate resulting rider interaction embeddings as shown in Fig. 18(a), yielding accurate RIL results.

6 DEPLOYMENT DISCUSSION

We briefly discuss the following deployment considerations of our current FCRIL prototype.

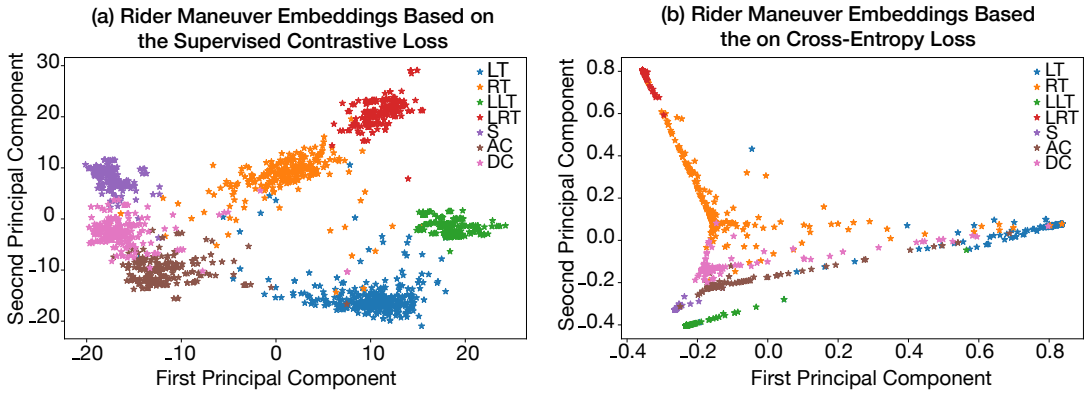

Fig. 18. Comparison of the resulting rider interaction embeddings: (a) w/ supervised contrastive loss; and (b) w/ conventional cross-entropy loss.

• **Societal Implications:** Our proposed RIL framework will benefit the micromobility manufacturers, sharing service operators, city planners, and other stakeholders in: *(i)* developing safety measures and improving interaction designs upon the manufactured and deployed e-scooters (e.g., handlebars, brake handles, and electric motor designs) based on the analyzed maneuver behaviors and their dynamics; *(ii)* reducing conflicts and collision risks with motorized vehicles, pedestrians, and pedaled bikes and other scooters by understanding the rider behaviors in various spatio-temporal contexts; *(iii)* promoting safe riding conducts and policies within and across the rider communities. For instance, city planners can receive more insights from our FCRIL framework's output to determine safe and suitable road types or conditions for e-scooters in complex city environments.

• **Expanding Rider Analysis and System Studies:** Our focus here is to design the core RIL algorithm and develop a prototype system framework for understanding and recognizing the e-scooter rider maneuvers. Due to the severe impacts of the COVID-19 pandemic, we were not able to conduct large-scale rider studies from the e-scooter rider communities for the RIL studies. However, our model and system prototype designs, including the federated maneuver learning mechanism, are generalizable and scalable, and can be easily extended to more rider studies (e.g., through crowdsourcing [11, 39]). We have also conducted various case studies to further corroborate the adaptivity and generalizability of our approach. Towards more in-depth RIL and behavioral analytics, we will take into account factors such as more rider studies, diverse weather conditions, and other mobile/wearable Android devices in our future studies.

• **Further Data Privacy Designs:** Our current prototype studies focus on understanding the rider maneuvers as well as the impacts of integration with a federated learning setting for privacy-preserving implications. Further data security and privacy enhancement designs (in response to various privacy attacks [9, 29, 37, 38]), including differential location privacy, have been studied from other application domains [7]. Since they are outside the major scope of the proposed studies, we will further investigate in our future studies.

7 CONCLUSION

We have proposed FCRIL, a novel e-scooter rider interaction learning framework based on contrastive federated maneuver recognition. Specifically, we have designed the rider interaction embedding learning network (RIEL) to learn and differentiate the e-scooter rider maneuver embeddings based on comprehensive RIL feature extraction as well as a novel supervised contrastive loss function. Furthermore, we have designed the rider interaction and maneuver recognition (RILMR) network to further perform RIL and rider maneuver recognition given the learned

rider interaction embeddings. We have integrated a scalable federated learning algorithm that asynchronously receives the gradients from the edge e-scooter (and its on-board mobile devices) and updates a global instance of our FCRIL model towards accurate RIL. We have performed extensive and real-world experimental studies based on our harvested e-scooter rider maneuver dataset, and have demonstrated the high accuracy and deployment potentials of our proposed FCRIL framework for ubiquitous, scalable, and privacy-preserving e-scooter rider behavior analysis.

ACKNOWLEDGMENTS

This project is supported, in part, by the 2021 Google Research Scholar Program Award.

REFERENCES

[1] 2018. New Tech Is Helping Lime Keep Bikes And Electric Scooters Upright. https://www.li.me/blog/new-tech-helping-lime-keep-bikes-electric-scooters-upright.
[2] 2020. Electric Scooters Market Size, Share & Trends Analysis Report By Product (Retro, Standing/Self-Balancing, Folding), By Battery (Sealed Lead Acid, NiMH, Li-ion), By Voltage, And Segment Forecasts, 2022 - 2030. https://www.grandviewresearch.com/industry-analysis/electric-scooters-market.
[3] 2021. E-scooter brain with smart sensors to be trialled in four cities. https://futureiot.tech/e-scooter-brain-with-smart-sensors-to-be-trialled-in-four-cities/.
[4] 2022. *DOT to Expand E-Scooter Pilot in the Bronx, Doubling Footprint and Fleet Size.* https://www1.nyc.gov/html/dot/html/pr2022/expand-e-scooter-pilot-in-the-bronx.shtml
[5] 2022. More Rental E-Scooters Are Coming to NYC in Expanded Pilot Program. https://www.thrillist.com/news/new-york/find-rental-e-scooters-nyc-lime-bird-veo.
[6] Rushdi Alsaleh, Mohamed Hussein, and Tarek Sayed. 2020. Microscopic behavioural analysis of cyclist and pedestrian interactions in shared spaces. *Canadian Journal of Civil Engineering* 47, 1 (2020), 50–62.
[7] Philip Asuquo, Haitham Cruickshank, Jeremy Morley, Chibueze P Anyigor Ogah, Ao Lei, Waleed Hathal, Shihan Bao, and Zhili Sun. 2018. Security and privacy in location-based services for vehicular and mobile communications: an overview, challenges, and countermeasures. *IEEE IoT-J* 5, 6 (2018), 4778–4802.
[8] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).
[9] Christopher Briggs, Zhong Fan, and Peter Andras. 2021. A review of privacy-preserving federated learning for the Internet-of-Things. *Federated Learning Systems* (2021), 21–50.
[10] Giuseppe Cantisani, Laura Moretti, and Yessica De Andrade Barbosa. 2019. Safety problems in urban cycling mobility: A quantitative risk analysis at urban intersections. *Safety* 5, 1 (2019), 6.
[11] Georgios Chatzimilioudis, Andreas Konstantinidis, Christos Laoudias, and Demetrios Zeinalipour-Yazti. 2012. Crowdsourcing with smartphones. *IEEE Internet Computing* 16, 5 (2012), 36–44.
[12] Chao Chen, Qiang Liu, Xingchen Wang, Chengwu Liao, and Daqing Zhang. 2021. semi-Traj2Graph: Identifying fine-grained driving style with GPS trajectory data via multi-task learning. *IEEE Transactions on Big Data* (2021).
[13] United States Consumer Product Safety Commission. 2021. *Injuries Using E-Scooters, E-Bikes and Hoverboards Jump 70% During the Past Four Years.* https://www.cpsc.gov/Newsroom/News-Releases/2021/Injuries-Using-E-Scooters-E-Bikes-and-Hoverboards-Jump-70-During-the-Past-Four-Years
[14] Yichen Ding, Xun Zhou, Han Bao, Yanhua Li, Cara Hamann, Steven Spears, and Zhuoning Yuan. 2020. Cycling-Net: A Deep Learning Approach to Predicting Cyclist Behaviors from Geo-Referenced Egocentric Video Data. In *Proc. ACM SIGSPATIAL*. 337–346.
[15] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep Learning*. Vol. 1. MIT Press Cambridge.
[16] Amara Graps. 1995. An introduction to wavelets. *IEEE Computational Science and Engineering* 2, 2 (1995), 50–61.
[17] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *Proc. IEEE CVPR*, Vol. 2. 1735–1742.
[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proc. IEEE CVPR*. 770–778.
[19] Suining He and Kang G. Shin. 2020. Dynamic Flow Distribution Prediction for Urban Dockless E-Scooter Sharing Reconfiguration. In *Proc. WWW*. 133–143.
[20] Suining He and Kang G. Shin. 2022. Socially-Equitable Interactive Graph Information Fusion-Based Prediction for Urban Dockless E-Scooter Sharing. In *Proc. WWW*. 3269–3279.

[21] Christopher E Heil and David F Walnut. 1989. Continuous and discrete wavelet transforms. *SIAM Rev.* 31, 4 (1989), 628–666.
[22] Sara Hernández Sánchez, Rubén Fernández Pozo, and Luis Alfonso Hernández Gómez. 2019. Deep neural networks for driver identification using accelerometer signals from smartphones. In *Proc. International Conference on Business Information Systems*. Springer, 206–220.
[23] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. 2017. Densely connected convolutional networks. In *Proc. IEEE CVPR*. 4700–4708.
[24] Jemshit Iskanderov and M Amac Guvensan. 2020. Breaking the limits of transportation mode detection: Applying deep learning approach with knowledge-based features. *IEEE Sensors Journal* 20, 21 (2020), 12871–12884.
[25] Arash Jahangiri and Hesham A Rakha. 2015. Applying machine learning techniques to transportation mode recognition using mobile phone sensor data. *IEEE T-ITS* 16, 5 (2015), 2406–2417.
[26] Wenqiang Jin, Srinivasan Murali, Youngtak Cho, Huadi Zhu, Tianhao Li, Rachael Thompson Panik, Anika Rimu, Shuchisnigdha Deb, Kari Watkins, Xu Yuan, and Ming Li. 2022. CycleGuard: A Smartphone-Based Assistive Tool for Cyclist Safety Using Acoustic Ranging. *Proc. ACM IMWUT* 5, 4, Article 163 (Dec 2022), 30 pages.
[27] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Proc. NeurIPS* 33 (2020), 18661–18673.
[28] Jin-woo Lee, Jaehoon Oh, Sungsu Lim, Se-Young Yun, and Jae-Gil Lee. 2020. Tornadoaggregate: Accurate and scalable federated learning via the ring-based architecture. *arXiv preprint arXiv:2012.03214* (2020).
[29] Chenglin Li, Di Niu, Bei Jiang, Xiao Zuo, and Jianming Yang. 2021. Meta-HAR: Federated representation learning for human activity recognition. In *Proc. WWW*. 912–922.
[30] Pei Li, Mohamed Abdel-Aty, Qing Cai, and Zubayer Islam. 2020. A deep learning approach to detect real-time vehicle maneuvers based on smartphone sensors. *IEEE T-ITS* (2020).
[31] Xiaoyuan Liang, Yuchuan Zhang, Guiling Wang, and Songhua Xu. 2019. A deep learning model for transportation mode detection based on smartphone sensing data. *IEEE T-ITS* 21, 12 (2019), 5223–5235.
[32] Huang Ling and Jianping Wu. 2004. A study on cyclist behavior at signalized intersections. *IEEE T-ITS* 5, 4 (2004), 293–299.
[33] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* (2015).
[34] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. 2017. Communication-efficient learning of deep networks from decentralized data. In *Proc. International Conference on Artificial Intelligence and Statistics*. PMLR, 1273–1282.
[35] Louis A Merlin, Xiang Yan, Yiming Xu, and Xilei Zhao. 2021. A segment-level model of shared, electric scooter origins and destinations. *Transportation Research Part D: Transport and Environment* 92 (2021), 102709.
[36] Hélie Moreau, Loïc de Jamblinne de Meux, Vanessa Zeller, Pierre D'Ans, Coline Ruwet, and Wouter MJ Achten. 2020. Dockless e-scooter: A green solution for mobility? Comparative case study between dockless e-scooters, displaced transport, and personal e-scooters. *Sustainability* 12, 5 (2020), 1803.
[37] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. 2021. A survey on security and privacy of federated learning. *Future Generation Computer Systems* 115 (2021), 619–640.
[38] Xiaomin Ouyang, Zhiyuan Xie, Jiayu Zhou, Jianwei Huang, and Guoliang Xing. 2021. ClusterFL: A Similarity-Aware Federated Learning System for Human Activity Recognition. In *Proc. ACM MobiSys*. 54–66.
[39] Jurairat Phuttharak and Seng W Loke. 2018. A review of mobile crowdsourcing architectures and challenges: Toward crowd-empowered internet-of-things. *IEEE Access* 7 (2018), 304–324.
[40] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. 2020. FedPAQ: A communication-efficient federated learning method with periodic averaging and quantization. In *Proc. International Conference on Artificial Intelligence and Statistics*. PMLR, 2021–2031.
[41] Felix Wilhelm Siebert, Madlen Ringhand, Felix Englert, Michael Hoffknecht, Timothy Edwards, and Matthias Rötting. 2021. Braking bad–Ergonomic design and implications for the safe use of shared E-scooters. *Safety Science* 140 (2021), 105294.
[42] Konstantin Sozinov, Vladimir Vlassov, and Sarunas Girdzijauskas. 2018. Human Activity Recognition Using Federated Learning. In *Proc. IEEE Intl Conf on Parallel Distributed Processing with Applications, Ubiquitous Computing Communications, Big Data Cloud Computing, Social Computing Networking, Sustainable Computing Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*. 1103–1111.
[43] Mahan Tabatabaie, Suining He, and Xi Yang. 2021. Reinforced Feature Extraction and Multi-Resolution Learning for Driver Mobility Fingerprint Identification. In *Proc. ACM SIGSPATIAL*. 69–80.
[44] Mahan Tabatabaie, Suining He, and Xi Yang. 2022. Driver Maneuver Identification with Multi-Representation Learning and Meta Model Update Designs. *Proc. ACM IMWUT* 6, 2, Article 74 (jul 2022), 23 pages.
[45] Nima Taherifard, Murat Simsek, Charles Lascelles, and Burak Kantarci. 2020. Attention-based event characterization for scarce vehicular sensing data. *IEEE Open Journal of Vehicular Technology* 1 (2020), 317–330.
[46] Qinrui Tang, Kanwal Jahan, and Michael Roth. 2022. Deep CNN-BiLSTM Model for Transportation Mode Detection Using Smartphone Accelerometer and Magnetometer. In *Proc. IEEE Intelligent Vehicles Symposium*. IEEE, 772–778.

[47] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. 2020. What makes for good views for contrastive learning? *Proc. NeurIPS* 33 (2020), 6827–6839.
[48] Linlin Tu, Xiaomin Ouyang, Jiayu Zhou, Yuze He, and Guoliang Xing. 2021. FedDL: Federated Learning via Dynamic Layer Sharing for Human Activity Recognition. In *Proc. ACM MobiSys*. 15–28.
[49] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Proc. NeurIPS* 30 (2017).
[50] Paul Voigt and Axel Von dem Bussche. 2017. The EU general data protection regulation (GDPR). *A Practical Guide, 1st Ed., Cham: Springer International Publishing* 10, 3152676 (2017), 10–5555.
[51] Chenxing Wang, Haiyong Luo, Fang Zhao, and Yanjun Qin. 2020. Combining residual and LSTM recurrent networks for transportation mode detection using multimodal sensors integrated in smartphones. *IEEE T-ITS* 22, 9 (2020), 5473–5485.
[52] Yiming Xu, Mudit Paliwal, and Xilei Zhao. 2021. Real-time Forecasting of Dockless Scooter-Sharing Demand: A Context-Aware Spatio-Temporal Multi-Graph Convolutional Network Approach. *arXiv preprint arXiv:2111.01355* (2021).
[53] Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. 2019. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 13, 3 (2019), 1–207.
[54] George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. 2021. A transformer-based framework for multivariate time series representation learning. In *Proc. ACM SIGKDD*. 2114–2124.
[55] Yi Zhao, Xu Wang, Jianbo Li, Desheng Zhang, and Zheng Yang. 2019. CellTrans: Private Car or Public Transportation? Infer Users' Main Transportation Modes at Urban Scale with Cellular Data. *Proc. ACM IMWUT* 3, 3 (2019), 1–26.
[56] Zilong Zhong, Jonathan Li, Zhiming Luo, and Michael Chapman. 2017. Spectral–spatial residual network for hyperspectral image classification: A 3-D deep learning framework. *IEEE Transactions on Geoscience and Remote Sensing* 56, 2 (2017), 847–858.
[57] Yida Zhu, Haiyong Luo, Runze Chen, Fang Zhao, and Song Guo. 2021. MSCPT: Toward Cross-Place Transportation Mode Recognition Based on Multi-Sensor Neural Network Model. *IEEE T-ITS* (2021).