PreDriveID: Pre-Trip Driver Identification from In-Vehicle Data

Gorkem Kar
WINLAB, Rutgers University
gkar87@winlab.rutgers.edu

Shubham Jain
WINLAB, Rutgers University
shubhamj@winlab.rutgers.edu

Marco Gruteser
WINLAB, Rutgers University
gruteser@winlab.rutgers.edu

Jinzhu Chen
General Motors Research
jinzhu.chen@gm.com

Fan Bai
General Motors Research
fan.bai@gm.com

Ramesh Govindan
University of Southern California
ramesh@usc.edu

ABSTRACT

This paper explores the minimal dataset necessary at vehicular edge nodes, to effectively differentiate drivers using data from existing in-vehicle sensors. This facilitates novel personalization, insurance, advertising, and security applications but can also help in understanding the privacy sensitivity of such data. Existing work on differentiating drivers largely relies on devices that drivers carry, or on the locations that drivers visit to distinguish drivers. Internally, however, the vehicle processes a much richer set of sensor information that is becoming increasingly available to external services. To explore how easily drivers can be distinguished from such data, we consider a system that interfaces to the vehicle bus and executes supervised or unsupervised driver differentiation techniques on this data. To facilitate this analysis and to evaluate the system, we collect in-vehicle data from 24 drivers on a controlled campus test route, as well as 480 trips over three weeks from five shared university mail vans. We also conduct studies between members of a family. The results show that driver differentiation does not require longer sequences of driving telemetry data but can be accomplished with 91% accuracy within 20s after the driver enters the vehicle, usually even before the vehicle starts moving.

CCS CONCEPTS

•**Information systems** →*Mobile information processing systems;* •**Computer systems organization** →*Real-time system architecture;*

KEYWORDS

Driving telemetry data, Vehicular sensing, On-board diagnostics

ACM Reference format:
Gorkem Kar, Shubham Jain, Marco Gruteser, Jinzhu Chen, Fan Bai, and Ramesh Govindan. 2017. PreDriveID: Pre-Trip Driver Identification from In-Vehicle Data. In *Proceedings of SEC '17, San Jose / Silicon Valley, CA, USA, October 12–14, 2017,* 13 pages.
DOI: 10.1145/3132211.3134462

1 INTRODUCTION

As vehicles are becoming programmable and connected, they are capable of supporting novel applications by computing the stream of data available on the vehicular platform. We expect that an increasing number of applications will enjoy access to internal vehicle data. Modern automobiles contain hundreds of sensors and actuators that exchange data on internal buses. A small part of this data has already been exposed in the OBD-II standard but the majority has, to date, been used only internally. Recently, car makers have been experimenting with opening more of this information to smartphone or in-car apps [16, 22]. Such data is also increasingly accessible through telematics services and could potentially be processed in the cloud.

Driver specificity of data. One relevant question in this context is how driver-specific the data is. How easily can different drivers of a vehicle be distinguished from such in-vehicle data—or, more precisely, what is the minimal amount of data necessary to effectively distinguish drivers? The answer to this question will help in understanding the feasibility of building driver-specific applications for the many vehicles that are used by multiple drivers. We focus on shared vehicles in a professional or commercial setting, and personal use setting, where applications include personalization of vehicle settings (e.g., automatically adjusting entertainment, preferred temperature, transmission, or suspension configurations to driver preferences), automated vehicle use logs, driver-dependent pay-as-you-drive insurance, or unauthorized vehicle use detection (where a vehicle might notify owners when it encounters an unexpected driver). The answer to this question will also contribute to an understanding of the privacy implications of such in-vehicle data. The more easily drivers can be distinguished, the less anonymous one can expect this data to be.

Existing work. Existing product solutions to distinguish drivers usually require a token, such as a smart key fob, that the driver carries. Such systems are robust only if every driver consistently uses a separate key fob. They are often limited to two keys due to cost and cannot distinguish drivers when keys are shared in commercial settings with more than two drivers. The academic literature has also explored how mobile devices in vehicles can determine whether they are used in the driver area of the vehicle, which would usually indicates that their owner is the driver [12, 23, 24]. Such techniques can also identify the driver, but depend on specific interfaces to the vehicle, keeping the phone close-by while driving, or the usage of more advanced wearable devices.

Existing work towards understanding the driver-specificity of vehicle data has been limited to a few parameters such as vehicle movement and steering inputs [5]. More work exists, of course, on location data which can also be obtained from cell phones inside a vehicle [7–9]. Such techniques require a complete trace of vehicle data from a longer trip. These results therefore tell us that drivers can be identified in longer sets of data but do not identify a minimal set of data for identification or convey a good sense about the ease of this identification.

A pre-trip profiling approach. In this paper, we address these questions by examining in-vehicle data streams and exploring a driver differentiation system that can rely on minimal time sequences of in-vehicle data. We define minimal time sequences in terms of the amount of time that has passed since approaching the vehicle for a new trip. This allows us to understand the driver-specificity of different types of in-vehicle data generated over the course of a trip, and it is also consistent with the personalization use case, wherein the vehicle needs to rapidly identify the driver to switch to the driver's preferences. Our system can leverage the sensing and computation modalities of the vehicular edge platform, thereby enabling support for driver identification faster than mainstream approaches, using unexplored in-vehicle sensor data.

We first examine sensor information shared on the vehicle bus for its driver specificity. This analysis includes data that was so far largely unavailable to external entities. We find that, in addition to the expected driving telemetry data generated while the vehicle is steered, the data streams contain a rich set of fields that reflect other driver actions, such as fastening seat belts, closing doors, or changing HVAC settings. A particularly revealing burst of this data occurs at the start of a trip, before the vehicle starts moving. Based on this insight, we explore a pre-trip data profiling approach and compare it with the use of more conventional driving telemetry data. Pre-trip data are due to driver actions taken in the first 20 seconds after entering the vehicle and include: the time at which the vehicle door was closed, the vehicle was started, the seatbelt was fastened, and the brake pedal was released. Driving telemetry data includes vehicle speed, acceleration/deceleration patterns, braking patterns at stop signs, or turn signal use. We consider a system that monitors these events on the vehicle bus, extract timing features, and distinguish different drivers of one vehicle using a classifier. We conduct controlled experiments with 24 volunteer drivers, who take a test vehicle along a pre-defined campus course. We also collect and analyze a real-world dataset of 480 trips from five shared university mail-vans spread over 12 weeks. Finally, we conduct a three-week study with households and present the results. The experiments reveal that the data when starting the car is actually more revealing, than the driving behavior on the roadway.

In summary, the salient contributions of this work are the following:

- accessing a rich set of in-vehicle sensor data through a custom CAN bus interface and examining its driver-specificity; this explicitly includes data that was so far inaccessible through interfaces such as OBD-II and OpenXC.
- designing classifier features and a system that allows distinguishing drivers based on a minimal set of in-vehicle sensor data, with no additional hardware cost.
- evaluating the system with data from 480 real-world trips collected over 3 weeks from five university mail vans, with 24 drivers in a controlled experiment, and 103 trips with four drivers across two households.
- finding that data from the vehicle start is particularly specific to individual drivers, allowing our system to achieve 91% of accuracy within 20s after the driver enters the vehicle in the real-world mail van experiment.

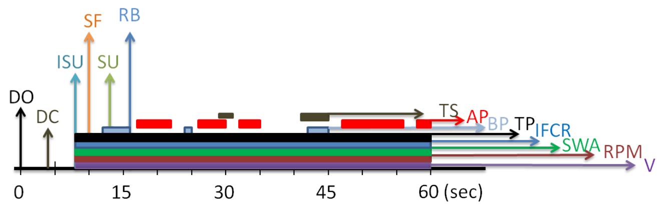


Figure 1: Timing events of the first minute of a trip.

2 BACKGROUND AND APPLICATIONS

Modern vehicles are equipped with many Electronic Control Units (ECUs) that control and monitor different vehicle modules, such as the engine, power windows, HVAC, power seats, or doors. Most ECUs are connected to the Controller Area Network (CAN) bus [4], which is a standardized vehicle data bus that allows those ECUs to communicate with each other. Many vehicle functions require (aggregated) sensor data from other ECUs. For this reason, many sensor data fields are broadcast on this bus.

2.1 Accessing In-Vehicle Data

The extent to which this data is accessible through the On-Board Diagnostics II (OBD II) port[1] varies. OBD II is a standard interface for vehicles to provide self-diagnostics and data reporting capabilities, and has been mandatory for vehicles sold in the United States since 1996. While this port is usually directly connected to the CAN bus, only a small subset of fields are mandatory, primarily fields relevant for government permission testing and basic troubleshooting. In modern vehicles, the majority of in-vehicle data uses proprietary encodings and is not directly accessible through the OBD II standard. In recent years, we have witnessed efforts to open more of this proprietary in-vehicle data to external entities, since there exist considerable opportunities to exploit this data for other applications. For example, OpenXC [22] provides access to select proprietary in-vehicle data fields through a special OBD-II adapter. Similar to some standard OBD-II adapters, this OBD-to-Bluetooth device relays CAN messages to Bluetooth equipped mobile devices (e.g., smartphones), but it decodes additional proprietary fields through a custom firmware. Similarly, the OnStar AppFramework provides access to select proprietary fields of General Motors vehicles with OEM-enabled APIs[2]. With the increasing availability of broadband (e.g., LTE) connectivity in vehicles, car makers could also remotely access and process vehicle data.

[1]www.obdii.com
[2]https://developer.gm.com

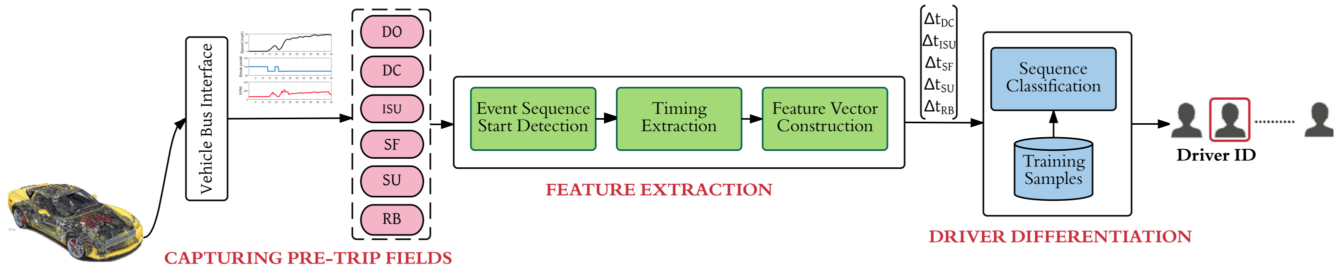


Figure 2: System Overview.

2.2 Applications

With increasing access to in-vehicle data, we foresee a broad vista of vehicle apps, many of which may benefit from implementing driver-specific functions. Let us consider the following examples.

In-Car Personalization. Modern vehicles are capable of personalizing settings according to our preferences. Some vehicles already automatically learn and switch to personal settings for radio station, temperature, dashboard display brightness, navigation view, among others. This is straightforward for vehicles driven by only one driver. Some vehicles provide two keys with different electronic IDs and encourage drivers to consistently use the same key. The vehicle then switches preferences based on the key. This remains inaccurate whenever drivers share keys, which is very common. Particularly in family settings, drivers may opt for the nearest or most easily obtainable key fob and leave, or often some may set one of the fobs aside as a backup in case of emergencies, while they continue to share the same key. One might argue that drivers can easily be distinguished by using their smartphone as an authenticator. Again, this has limitations when family members who share a car, ride together. With more than one driver's smartphone being present inside the car, driver distinction becomes difficult if that's the sole differentiation mechanism.

Pay-as-you-drive Insurance. With the rising popularity of usage-based insurance premiums, automotive insurance companies are now tracking driver behavior with incentives for safe driving. This is primarily done via estimation of parameters like rapid acceleration, hard braking, air bag deployment etc., by reading sensor data over the CAN bus. In addition to adjusting insurance rates based on driver behavior and actual vehicle usage, pay-as-you-drive insurance rates could also take into account who actually drove the vehicle.

Targeted Advertisement. Advertising revenues have had a significant impact on the Internet and mobile economy. With the trend towards programmable, and connected vehicles, they can also be expected to play a role in vehicle-related applications. Driver differentiation can help in constructing personal profiles and improve the targeting of advertisements.

Detecting Unauthorized Use of Vehicle. Vehicles are rapidly being connected to the Internet with broadband technologies. If the vehicle can differentiate drivers, it could also potentially identify unauthorized drivers. On detecting such an unauthorized driver, the vehicle could notify the owner or, with sufficient confidence, other authorities.

All these applications benefit from a low-cost system to distinguish different drivers of a shared vehicle. Driver differentiation is thus a fundamental technique for the future vehicle applications to provide customized experiences to the users.

3 DRIVER DIFFERENTIATION

We seek to identify a minimal time-sequence of in-vehicle data to distinguish drivers by identifying events that are closely tied to driver habits and behaviors but minimally affected by driving conditions and other traffic participants.

3.1 Selection of Pre-Trip Events

We analyzed the available in-vehicle data and identified 14 fields that are available in many vehicle models and whose value commonly changes early in a trip. These fields are illustrated in the timeline in Fig. 1, derived from one example trip with a mail van. Note that the vehicle generates data even before it starts moving and that many of these initial events directly correspond to driver actions such as *door opening (DO)*, *door closing (DC)*, *starting the ignition (ISU)*, *seatbelt fastening (SF)*, *shifting gear (SU)* and *releasing the brake pedal (RB)*. As the engine is turned on and the vehicle begins to move, additional driving and telemetry data streams indicating steering wheel angle (SWA), engine revolutions per minute (RPM), vehicle speed (V), and acceleration (AP) values become available.

We hypothesize that the pre-trip events generated before the vehicle moves are not only available early after entering the vehicle but are also particularly distinctive because they are largely dependent on habit and unaffected by the road configuration and actions of other traffic participants. While the type of events does not differ across drivers, the order and precise timing of these actions is mostly determined by habit. To what degree one turns the steering wheel while leaving a parking lot is often affected by the presence of other obstacles and vehicles at that particular location. The relative timing of ignition start and seatbelt fastening, in comparison should not depend significantly on these external factors. These steps, their specific order (sequence) and their timing interval should therefore, be helpful in creating a minimal driver profile.

To support the hypothesis, we conducted a preliminary experiment with eight drivers. The drivers were instructed to drive a Cadillac CTS and complete a loop in the parking lot. Each driver repeated the experiment 10 times. For this preliminary experiment, drivers were asked to consistently follow their regular habits, to reveal possible distinct patterns across drivers. We describe our in-the-wild experiments in the evaluation section. Figure 3 shows the pre-trip event timing collected from those drivers in a scatter plot with quartiles marked for each event type. Time zero is defined as the door open event, the first event related to this trip. The data shows that the relative timing is quite distinct across drivers, even when drivers started the vehicle in the same controlled test situation. We also observe that these pre-trip events occur within 20 seconds after opening the door in all cases.

3.2 System Overview

Based on the aforementioned insights, we consider a driver differentiation system that seeks to distinguish drivers using a minimal time sequence of in-vehicle data. The system primarily consists of three components, vehicle bus data capture, feature extraction, outlier rejection, and driver differentiation. It obtains in-vehicle data, particularly pre-trip events, through a CAN bus interface. The feature extraction module scans this data to identify the start of a new trip, extract event timings, and construct a feature vector. This vector is then examined for outliers before being processed by a classification algorithm, that matches the feature vector to profiles constructed from past trips of the same vehicle. In applications where labeled training data does not exist, the use of unsupervised classification techniques is also possible, and presented in Section 5.

There are multiple possible realizations of such a system in practice, and we illustrate the design space through the following examples. Potentially, all the above mentioned components could be directly embedded in the vehicles, perhaps as part of a driver personalization feature that is transparent to users. Such built-in components could directly access the CAN bus and acquire the necessary data from there. A second possibility is that the feature extraction and driver differentiation functions are executed in cloud-based car maker applications and receive access to the vehicle data stream over increasingly available wireless broadband data connections to vehicles. For commercial settings, where the same driver may end up driving different vehicles on different days, running the driver differentiation module on a remote server is more suitable. However, for drivers who share the same car, such as members of a family, the computation can be done locally on the vehicle itself. A third possibility is that the components are realized within a third-party application that acquires sensor data through a vehicle manufacturer developer API. Depending on the availability of suitable APIs and other considerations, such applications could reside either on an app platform in the vehicle or in the cloud. It is also possible that some components are located on a mobile device brought into the vehicle, which pairs with a vehicle interface that provides access to the vehicle bus. In our experiments, we focused on this last option, a smartphone interface that allows data capture, with feature extraction and classification can be performed either in the cloud or in the vehicle.

3.3 Driver Profiling

We create a robust driver profiling approach, that extracts features resilient to precise driving style but derived from driving habits.

3.3.1 Feature Selection. Our features emphasize pre-trip data that represent seemingly innocuous habitual patterns of every driver, which are crucial in distinguishing them from others. For every pre-trip event k, the corresponding feature is defined as Δt_k, which is the time difference between the occurrence of event k and $k-1$. Specifically, our pre-trip feature vector f_{pt}, is defined as follows.

$$f_{pt} = [\Delta t_{DC},\ \Delta t_{ISU},\ \Delta t_{SU},\ \Delta t_{SF},\ \Delta t_{RB}]$$

Here, Δt_{DC} represents the time difference between the occurrence of the (driver) Door Close (DC) event and the occurrence of a reference starting event, which marks the beginning of the vehicle data stream for the new trip. Unless otherwise mentioned, we use the (driver) Door Open (DO) event as the reference event, since this was the first observable event on the vehicles that we experimented with. Similarly, Δt_{ISU}, Δt_{SU}, Δt_{SF}, and Δt_{RB} represent the time difference of the Ignition Switch Usage (ISU), Shift Usage (SU), Seatbelt Fastened (SF), and Release of the Brake pedal (RB) events to events DC, ISU, SU, and SF, respectively. The feature vector is always constructed in the same order irrespective of the actual order of events. In considering time intervals between specific events, we capture their relative occurrences. The DO and DC events are usually triggered when the driver enters the car. The ISU event represents starting the engine. The Seatbelt Fastened event occurs when the driver fastens their seatbelt. Shift Usage refers to changing the setting on an automatic transmission from park mode to another mode, often drive or reverse. Lastly, the Release Brake event marks the instance when the brake pedal was released to start driving.

In addition to pre-trip fields, we also extract features from the driving fields for comparison. Our driving feature matrix f_d, is defined as follows:

$$f_d = \begin{bmatrix} bp_1 & ap_1 & rpm_1 & tp_1 & ts_1 & v_1 & swa_1 \\ bp_2 & ap_2 & rpm_2 & tp_2 & ts_2 & v_2 & swa_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ bp_N & ap_N & rpm_N & tp_N & ts_N & v_N & swa_N \end{bmatrix}$$

Each row in f_d is a feature vector at some time i. Every feature is the value of the sensor as read from the vehicle bus. bp_i and ap_i represent how far the Brake Pedal and Accelerator Pedal are pressed at time i. Similarly, rpm_i denotes the engine Revolutions Per Minute, and tp_i represents the Throttle Position. Turn Signal is signified by ts_i. v_i and swa_i stand for the values obtained from the vehicle Velocity and Steering Wheel Angle sensors. The frequency and range of these sensors is shown in Table 1.

3.3.2 Adaptive Outlier Rejection. The normalized feature vector is used as an input to a learning algorithm. Using all the features in f_{pt} can help us differentiate drivers, but is easily affected by slight variations in event timings. To account for day to day driving behavior, we devise an adaptive outlier rejection technique. We note that although pre-trip sequences are peculiar to each driver, they are not always exactly comparable. Circumstances may arise when drivers break out of their pre-trip routine, causing a larger than usual delay in one of the events, as a result of which the subsequent events might be delayed as well. A common example is when drivers are interrupted by a phone call. Consequently, the corresponding feature is an outlier compared to past values

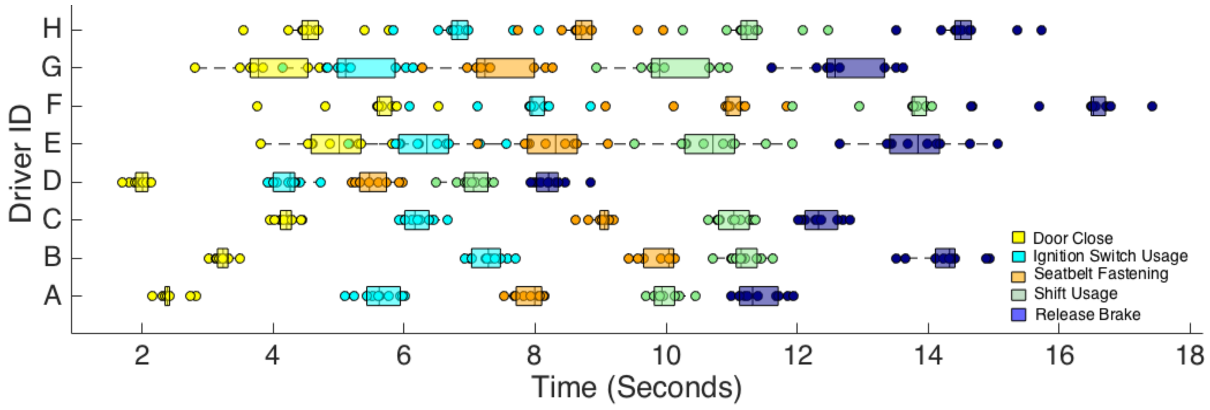


Figure 3: Pre-trip fields timeline - Controlled Experiment.

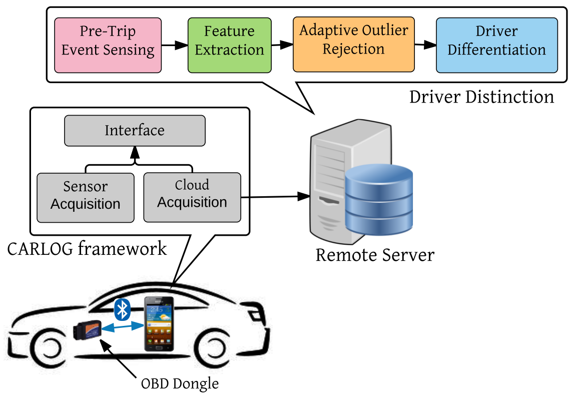


Figure 4: Our system implementation.

of that feature for the same driver. In such cases, we do not want the outlying time interval to affect our classification. The adaptive outlier rejection technique is designed to identify such outliers at test time, and adapt to them. We examine our pre-trip feature vector for outliers by comparing each feature value to the distribution of that feature. We deem a particular feature as an outlier if it lies one standard deviation or more above the mean for that feature. In this case, we dispose of that feature, but retain the subsequent time intervals (features). Note that we only remove at most one feature from f_{pt}. In case of multiple outliers, we discard the feature that is the furthest. The feature vector is now reduced to *n-1* values.

3.3.3 Two-step Driver Validation. For driver validation at run time, we perform outlier rejection and model selection. We use supervised learning for differentiating between drivers. During the training phase, we train a classifier using all the features (*n=5*), called the complete model. In addition, we fit separate models on different combinations of all but one feature in the training data. We achieve this by removing one column (feature) at a time. We refer to these models as partial learners. Partial learners learn from $n - 1$ features. Note that we train partial learners on the entire test set and not just those with outliers. Removing at most one feature at a time limits the total number of models to $n + 1$. If an incoming pre-trip sequence has no outliers, i.e. it conforms to within one standard deviation around the mean of the feature, we use all the features in f_{pt} and test it against the complete model. When we observe an outlier in incoming trip data, say a long delay for event k, we dispose of the Δt_k, but retain the subsequent time intervals. Our algorithm classifies the driver by testing this feature vector (with n-1 features), against the corresponding partial learner.

We use Support Vector Machine (SVM) with a linear function as our learning algorithm, with 5-fold cross validation. We observed this simple learner to give the best performances for our data as compared to other kernels, such as cubic and gaussian, and different learning algorithms like k-means clustering and decision trees. Additionally, this simple approach is computationally lightweight and suitable for real-time driver differentiation on COTS mobile devices. In an automated vehicle use logging, the set of drivers is typically known, and labeled training data may be available from earlier manual logs. In other applications, such as vehicle personalization, the number of drivers is far less. To collect training samples, the data logging application prompts the driver to mark the ground truth with a simple screen touch at the end of each trip.

4 IMPLEMENTATION

We have implemented the entire system using a custom OBD-II scan tool (dongle), a smartphone and a remote server. We place a smartphone in the vehicle that can communicate with the dongle over Bluetooth, as shown in Figure 4. Data can be requested through the dongle, by using Parameter IDs (PIDs). The dongle has been specifically designed for research purposes to make available a large set of internal vehicle bus fields that are not yet available through OBD-II or other interfaces such as OpenXC. The dongle sends a PID over the vehicle bus to request data.

We use the CARLOG [11] framework on the smartphone, which is a programming framework, for accessing sensor data from the vehicle. It houses a query optimizer that eases the task of querying, capturing and parsing low level sensor information from vehicles, and provides an interface for applications to access this information. We use a battery conscious smartphone application, shown in Fig 5 to record the sensor readings in our testbed of vehicles. The app

Pre-trip Fields	Frequency (Hz)	Range	Driving Fields	Frequency (Hz)	Range
Door status (DO & DC)	10	Boolean	Brake pedal (BP)	10	0-100
Ignition switch status (ISU)	10	Boolean	Accelerator pedal (AP)	50	0-100
Seatbelt status (SF)	10	Boolean	Revolutions per minute (RPM)	10	0-16000
Shifter position (SU)	40	Integer(1-6,13,14,15)	Throttle position (TP)	10	0-100
Parking brake active	100	Boolean	Turn signals (TS)	Event	Boolean
			Vehicle velocity (V)	10	0-255 kmh
			Steering wheel angle (SWA)	100	0-1340°

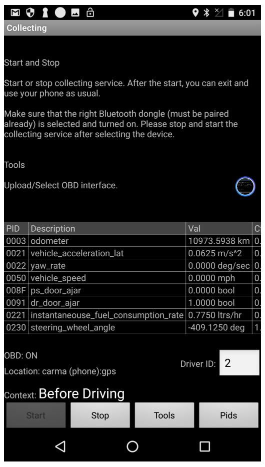
Table 1: Fields captured from the CAN bus.

Pre-trip fields	Mid-size sedan		Luxury vehicle		Van	
	Before ISU	After ISU	Before ISU	After ISU	Before ISU	After ISU
Door status (DO & DC)	χ	✓	✓	✓	χ	✓
Seatbelt status (SF)	χ	✓	χ	✓	χ	✓
Shift status (SU)	χ	✓	χ	✓	χ	✓
Release break (RB)	✓	✓	✓	✓	✓	✓

Table 2: Availability of pre-trip fields for different vehicle models.

continues logging until the connection to the vehicle is lost or the app receives no new sensor readings within 1 minute. In this timed-out state, the app closes the connection with the vehicle data port for 30 seconds; after this period, it reconnects to the vehicle, which restarts the cycle. The app sends specific predefined PIDs to the OBD dongle, which then requests it to the CAN bus.

The application initializes a subscription request to receive data from the sensors required for our pre-trip sensing. Among other things, it lets us define the sensors and their associated frequency, in samples per second. It captures data received from the dongle over extended period of time, handles all the incoming data, parses it and records it to local storage. The timestamp for this data is applied in the application when the event reaches the Carlog framework. It does not always accurately reflect the event time, but we expect the error to be small compared to event time differences that we use for our algorithm. In addition, the phone is connected to the internet and updates all this data to a remote server, via WiFi or cellular service. This ensures simultaneous sensing and uploading of the required data fields. For our proposed scheme, we access the on-board sensors on vehicles. Table 1 lists some of the sensors whose values are acquired by a smartphone with a bluetooth dongle, through the On-Board Diagnostics (OBD-II) port. We also implemented a real-time driver differentiation application for Android smartphones. The classification model is trained offline and loaded on to the smartphone for real-time classification as the driver approaches the car and following pre-trip events.



Start and Stop

Start or stop collecting service. After the start, you can exit and use your phone as usual.

Make sure that the right Bluetooth dongle (must be paired already) is selected and turned on. Please stop and start the collecting service after selecting the device.

Tools

Upload/Select OBD interface.

PID	Description	Val	C
0003	odometer	10973.5938 km	0.
0021	vehicle_acceleration_lat	0.0625 m/s^2	0.
0022	yaw_rate	0.0000 deg/sec	0.
0050	vehicle_speed	0.0000 mph	0.
008F	ps_door_ajar	0.0000 bool	0.
0091	dr_door_ajar	1.0000 bool	0.
0221	instantaneouse_fuel_consumption_rate	0.7750 ltrs/hr	0.
0230	steering_wheel_angle	-409.1250 deg	1.

OBD: ON

Location: carma (phone):gps

Driver ID: 2

Context: Before Driving

Figure 5: Logging Application for In-vehicle sensors.

5 PERFORMANCE EVALUATION

For evaluating our system, we aim to answer the following questions:

- What is the the minimal amount of data required to accurately identify a driver?
- How is accuracy affected with increasing number of drivers?
- Which vehicle data fields are most useful for prompt and accurate driver differentiation?
- How does training size impact system performance?
- How does unsupervised driver classification compare to supervised classification?
- How does driver differentiation adapt to drivers within the same household?

To gain an understanding of driver behavior, we carried out the following experiment.

5.1 Experimental Setup

Hardware and Signals. We used a 2008 Cadillac CTS vehicle (test vehicle), an LG Nexus 5 phone and a custom OBD scanner (dongle) to extract data from the vehicle. This custom dongle provides access to a richer set of vehicle data streams, compared to standard dongles. During the experiments, the smartphone is located inside the vehicle and the dongle is plugged into the OBD-II port. We conducted an IRB approved study, and used coded data for drivers instead of their actual identities. All participants are 18 years old or older and have a valid driver's license in United States.

Metrics. In our work, we evaluate the performance of driver detection algorithms in terms of accuracy and data length. Accuracy

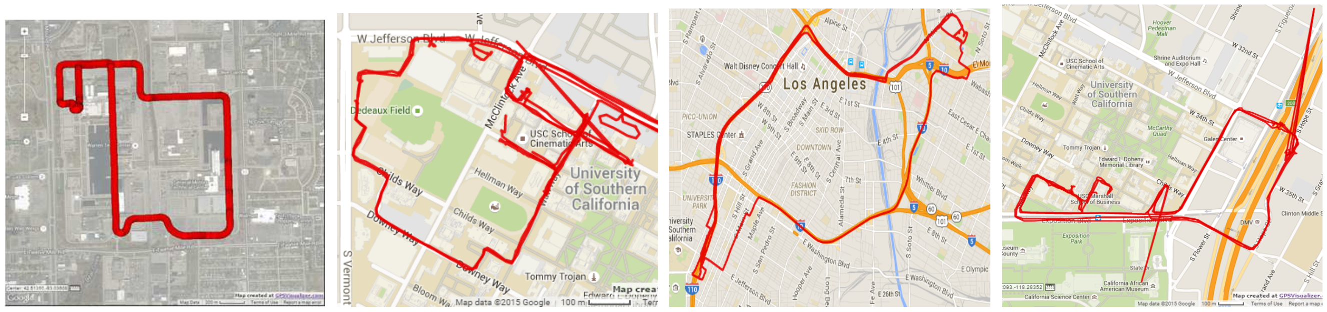


Figure 6: (a) Controlled test environment, (b) Mailvan first test environment, (c) Mailvan second test environment, (d) Mailvan third test environment

is the ratio of correctly identified number of drivers to the total number of drivers. Data length represents how much portion of the data was used for the identification process.

5.1.1 Controlled Experiment. To evaluate the performance of our system, we first use a dataset that was collected by 16 volunteers who drove the test vehicle on a 3 mile long road. The drivers were requested to drive as they normally would, and were not provided any instructions. They were allowed to make any adjustments they wanted, for example mirrors and seats. The only instruction provided to the drivers was to drive as usual along the given route. Most volunteers were not aware of the vehicle data that we were collecting other than it being related to driver's behavior. During this experiment, we monitored the activity of about 20 sensors, that communicate different types of vehicle state information. These sensors describe the state of electronic vehicular subsystems such as engine start events, cabin climate control and trip events such as speeding, braking, throttle positions and many others. Each driver is asked to drive a pre-defined path 10 times during the same day to create the database that consists of 160 traces. The test path for controlled experiment is shown in Figure 6 (a).

5.1.2 Mailvan Experiment. In the mail van driver differentiation tests, we compare the results of our classifier to anonymized versions of the USC Mailing Services Department driver records. These documents contain mandatory vehicle access logs that specify the times when drivers acquire and release a van. Drivers sign-out a single van before leaving the warehouse, and sign-in that van only after returning to the warehouse. During a day, between 1 and 5 drivers will use a single van. Each day, the drivers initially sign-in vans between 6 am and 8 am, and sign them out sometime around 5 pm. The vans are utilized for most of the working business hours, with delays between driver changes ranging from instantaneous to 60 minutes.

The USC Mail Service employees use the vans to traverse specific routes for delivering incoming and outgoing university postage. Each van is associated with a distinct set of routes, which are typically related by location. The three most common routes are shown in Figure 6 (b)-(d). The drivers use one van for example, to travel North East from the warehouse towards the USPS office in the morning and a nearby satellite campus in the afternoon. While traversing these routes the driver may make several stops to service multiple buildings on the campuses. These stops typically last between 10 and 60 minutes, and require the drivers to turn off and exit the vehicle. We consider each segment a different trip if the vehicle was turned off at the end of a segment. The mailvan dataset is the result of measuring the electronic subsystems of 7 USC Mailing Service fleet vehicles, over a period of 6 months. Out of the 6 months of vehicle data, we obtained 3 weeks of mail vehicle access logs for each USC Mail Service employee. Digitizing and anonymizing these driver logs requires manual effort, which limited our access to only 15 days worth of records. In an effort to maximize the number of driver changes included in our study, we selected the 3 weeks of measurements from our dataset having the most number of active vans and total number of sensor readings. These 3 weeks worth of data detail the actions of at least 5 vans each day, during the months of September, October, and November. This dataset includes 480 trip start trails that we are mainly using in our algorithm.

5.2 Driver Differentiation Evaluation

We evaluate the accuracy of the proposed driver differentiation algorithm with respect to duration since entering the vehicle, various classification techniques, training size, dataset size and importance of each pre-trip field. Unless otherwise specified, for supervised learning we use the 2-step driver validation with a 5-fold cross validation.

5.2.1 Identifying minimal data. With longer driving traces a driver can almost certainly be identified by modeling the vehicle speed, analyzing the destinations visited and the routes taken. This analysis, however, could take several minutes, or may be even hours. Prompt identification of drivers is pivotal for many personalization applications. We compare pre-trip fields and driving fields. Pre-trip fields are a result of fixed actions any driver undertakes before starting a drive. It is a finite set, the duration of which lasts only about 20 seconds or less. Driving fields on the other hand, could be collected for minutes or hours to accurately recognize a driver. We wish to explore the smallest time series of events that can provide reasonable driver differentiation accuracy. From Figure 7, it is evident that using only the pre-trip events, we can differentiate drivers with a higher than 90% accuracy, under 20 seconds. Note

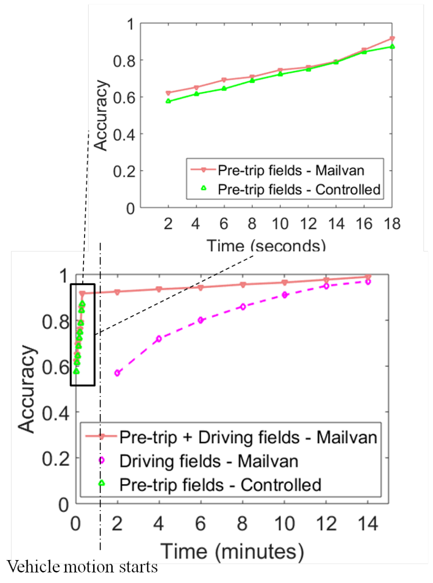


Figure 7: Accuracy over trip length. Using only pre-trip fields our system can achieve 90% accuracy in less than 20 secs.

the zoomed in version that plots the accuracy achieved by using the pre-trip events with respect to data length in seconds. This differentiation occurs before the driver actually starts driving, and well in time to support personalization applications and modifying vehicle settings based on preferences. Note that a baseline approach using only driving data requires approximately 10 min to reach the same accuracy that can be obtained from pre-trip data in 20s. On the other hand, driving data is useful to further improve the accuracy. Processing driving data in addition to the pre-trip data drives the accuracy to 98% within 10 mins. This strengthens our belief that pre-trip events are significant in determining driver behavior, and can be used effectively to perform such behavior based distinction in one-tenth the time proposed by prevalent driver recognition techniques. *The minimal duration of data required for 90% driver distinction accuracy, would thus be less than 20 seconds.*

5.2.2 *Effect of increasing number of drivers.* We take a step further in understanding these pre-trip fields, and their sequence. We asked 16 drivers to carry out the start-up process 10 times, and to drive on 3 mile route. This route was the same for all drivers, and the vehicle was in the same situation at the start of the drive. Since all the drivers were driving on the same route under almost similar traffic situations, and driving the same test vehicle, it might seem hard to accurately differentiate drivers with driving data alone.

It is noticeable from Figure 8, that most drivers do not have a clearly separable pattern for pre-trip fields and the time spacing between them. Even for the same driver, these events possess significant variance from the time the driver enters the car. With a large number of drivers, distinction becomes challenging due to high variance for each driver and apparent similarities in the pre-trip event timelines. This is the dataset we used to measure the performance of our algorithm. This emphasizes that even with minimalistic pre-trip fields, driver distinction cannot be performed by basic thresholding, and justifies the need for a learning component for understanding driver specificity. For a multi-class classification problem, such as ours, a confusion matrix is commonly employed to demonstrate the classifier performance. Figure 9(a) shows the confusion matrix for the mailvan dataset, and Figure 9(b) shows the confusion matrix for the controlled set. It can be seen from the figure that our classifier attains high accuracy levels, and does not misclassify drivers.

Next, we seek to observe the effect of number of drivers in the dataset, on the classifier performance. In the mailvan dataset, we have traces for five drivers and in the controlled experiment, we have traces for 16 drivers. Figures 10 (a) and 10 (b) show how accuracy improves when we focus on only a subset of drivers, and it reduces slightly as the number of drivers increases. We observe that, we can get up to 96% accuracy when differentiating four drivers in controlled experiment, and 98% when differentiating two drivers in the mailvan experiment. Our system accuracy declines by about 4% when the number of total drivers is increased from 4 to 16 in the controlled experiment, and by about 7% when the number of drivers increases from two to 4, in a real-world mailvan experiment. *Based on these results, we infer that system accuracy declines as the number of unique drivers in the dataset increases.*

5.2.3 *Analyzing individual field importance.* The next question we want to investigate is which data fields are most influential in rapidly differentiating drivers. In doing so, we first focus on the Door Open event as the origin event because it is the first event to occur, i.e at time=0. We compute the time difference between each individual event and the origin event. In addition to the aforementioned pre-trip fields, we add another field ACC, which represents the time taken by the driver to go from a speed of 0 mph to 5 mph. Figure 11 (a) shows the accuracy obtained using only one event at a time from each trace, for both the datasets. For the subset of 16 drivers and for the mail van experiment, we notice that Ignition Switch Usage and Shift Usage are one of the first events captured by the OBD device. This is primarily because accessing information about Door Open and Door Close may not be possible in all vehicles before the ignition is turned on. Considering this limitation of some vehicles, we also evaluate our system using Ignition Switch Usage (ISU) as the origin event. Figure 11 (b) shows the accuracy for all traces with origin event set as ISU. *We observe that of all pre-trip fields, Release Brake is the most important field for rapid driver differentiation.*

We also choose Analysis of variance (ANOVA) method to investigate the important fields. In this method, we calculate the mean for each field (field mean). We then calculate the mean for all fields combined, the overall mean, followed by the standard deviation within a field, for each field. Finally, we calculate the standard deviation of each field mean from overall mean. Figure 12(a) shows the statistics for each field that we get using ANOVA method. From this Figure, we observe that the highest between-group variations come from SF and ISU events.

Next we explored the importance of the driving fields in our mailvan dataset. The driving fields used are BP, AP, RPM, TP, TS, V and SWA. Figure 12(b) shows the accuracy achieved using each

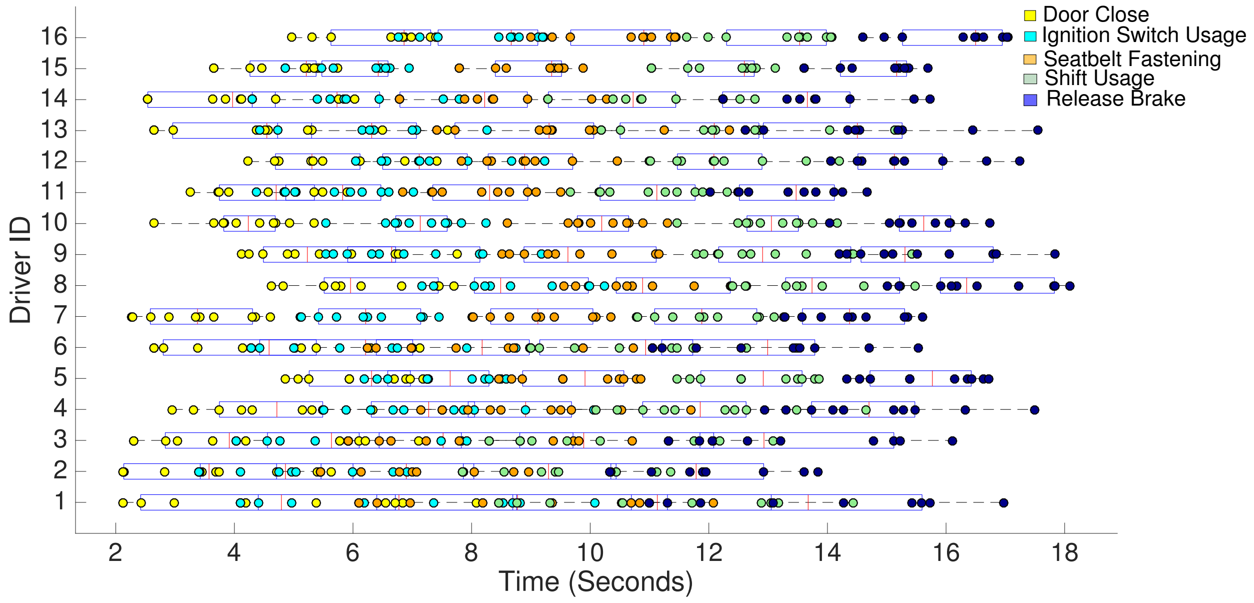
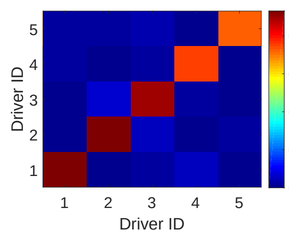
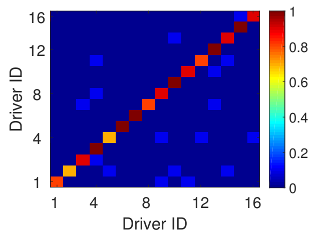


Figure 8: Timeline for pre-trip fields.

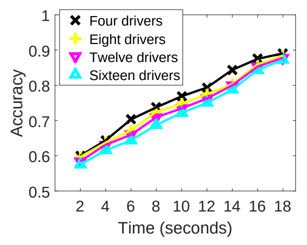


(a) Mailvan experiment



(b) Controlled experiment

Figure 9: Confusion matrix for driver differentiation.



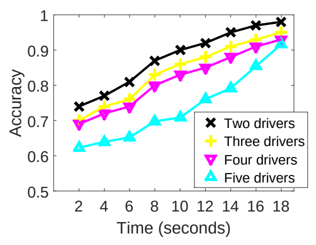
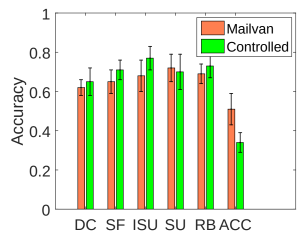


Figure 10: Varying number of drivers (a) Controlled experiment, (b) Mailvan experiment

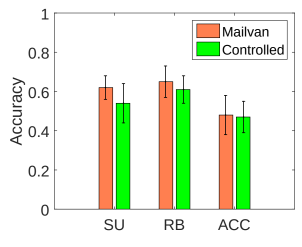
driving field alone for driver distinction. It must be noted that these events occur long after a person has started driving, and hence may not be as useful for applications that can use this data early on before the start of the trip. *We observe that BP and SWA are the most important driving fields in distinguishing drivers, and provide the highest accuracy when used alone.*

5.2.4 Effect of training size. For supervised learning using 2-step driver validation, choosing the portion of data to be used as a training set is critical since the remaining data is used for testing the algorithm. If the training set is too small, it may not be enough to cover all characteristics of dataset and performance results may not be good as expected. Or if that set is too big, there may not be enough data to be tested and performance results could be lower than expected. In our experiments, we choose to train our classifier on different number of traces. We select, as our training size, 50%, 60%, 70%, 80% and 90% of the data. The remaining data is then used as the test set. Figure 13 shows the accuracy of our classifier for different training set sizes. *As the training size increases, our algorithm provides better results as expected, and by using 90% of dataset as training set, we can get 91% and 89% accuracy for the mailvan and controlled experiments, respectively.*

5.2.5 Unsupervised driver differentiation. While our system performs well under supervised learning, we are also interested in quantifying its performance when prior driver data is not available for learning, i.e. unsupervised learning. We use K-means clustering as our first example of an unsupervised learning algorithm, where we only provide the number of expected drivers, but no advance information about driver behavior or learning traces. This algorithm then assigns each trace to a cluster, based on its distance from the cluster. Another unsupervised learning algorithm we investigate is hierarchical clustering algorithm. This algorithm does not require the number of drivers (clusters) or the event trace. It builds clusters iteratively with each input trace. We compare these two

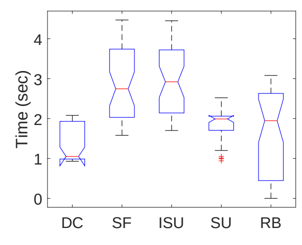


(a) Origin at DO event

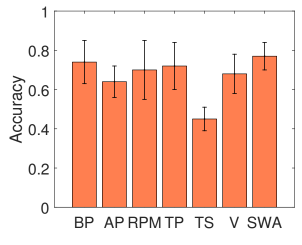


(b) Origin at ISU event

Figure 11: Performance of individual pre-trip fields.



(a) ANOVA comparison



(b) Accuracy comparison

Figure 12: Individual field importance.

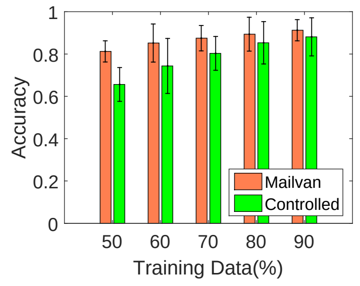


Figure 13: Accuracy over training data size.

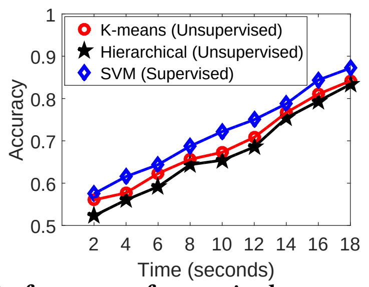


Figure 14: Performance of supervised vs unsupervised learning algorithms.

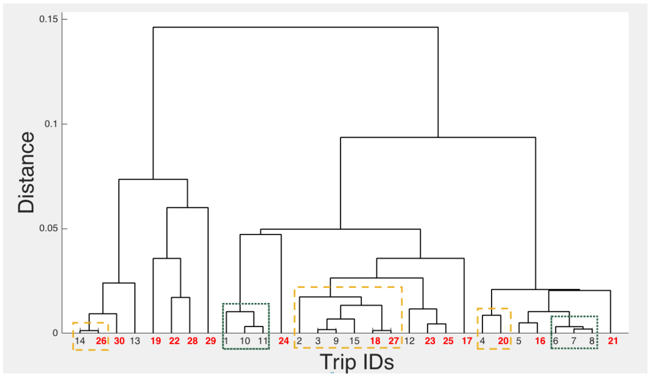


Figure 15: Similarity measure for 2 different drivers in a household. Trips IDs for husband: 1 to 15, Trip IDs for wife: 16 to 30

clustering approaches with the SVM algorithm. Figure 14 shows the comparison between supervised and unsupervised approaches for driver classification. *It is evident that at an accuracy of* 89% *in less than 20 seconds, supervised learning performs only slightly better than the unsupervised learning approach, with* 84% *accuracy for the same trace length.*

5.2.6 Personal vehicle use setting. While sharing of a single vehicle is most common in commercial settings, members of a household may also share a car. In some cases the sharing is somewhat uniform, where the drivers spend similar amounts of time driving the car. In other scenarios, each car may have a primary driver and an occasional secondary driver. We hypothesize that pre-trip sequences within members of a household may undergo higher variation than professional drivers. They may encounter more distractions from the time they approach the car, to the time they start driving. Moreover, one might think behavioral similarities are higher among family members, specially when teenagers learn driving from their parents. To this end, we conducted a three-week user study with two different households, where members share a car for personal use.

Experiment setting. Household 1 is a married couple, and Household 2 is a mother-daughter pair. Members of household 1 drove a Chevrolet Impala 2017, while the sensor data was being logged on a Nexus 4. About 3 days per week the wife drops the husband at work and uses the car for her daily chores. Most of her drives are non-routine and differ from day to day. Her stops are approximately one hour long. At the end of the day, she picks up the husband from work, and they go home together. On the other days, the husband drives himself to work and uses the car for lunch etc. They usually share driving tasks over the weekend. We collected 66 trips over a period of 3 weeks and logged all the pre-trips events and driving data on the smartphone. Each segment is considered a new trip if the engine was turned off. In the mother-daughter scenario, the mother, who is the primary driver, drives the car to work every day and back home. She drives a Chevrolet Equinox 2016, and a Nexus 5X was used for logging data. On some days she even drives it out for lunch, and other daily chores. The daughter drives the car far less, and thus is the secondary driver, often only over the weekends. We use this scenario for new driver detection, wherein we want to identify when the driver is not the primary driver of the vehicle. This entails what is known as one-class learning.

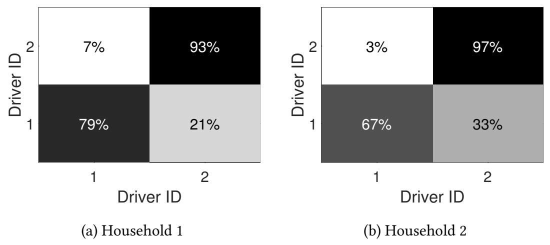


Figure 16: Confusion matrix for household experiments.

Evaluation. Fig 16(a) shows the confusion matrix for Household 1, where we use our adaptive outlier rejection algorithm with 5-fold cross validation, for distinguishing between husband and wife. The detection accuracy of this classifier was 85.6%. The data collected for the couple was significantly different from that observed during the commercial setting of the mailvan experiments. The start up sequences were longer by a factor of 10. During the exit interview, the couple informed that several times they made calls after entering the car, before starting to drive. Additionally, the order of pre-trip events was observed to be more irregular for personal use scenario, as compared to the professional use case.

Figure 15 depicts a dendrogram to visualize the similarity between drivers in Household 1. The vertical axis indicates the average distance between clusters, using correlation as the distance metric. Thus, lower distance implies higher correlation. The height of a node represents the distance of the two clusters that the node joins. We randomly select 15 trips from each driver. Trips IDs 1-15 are obtained from the husband, and Trip IDs 16-30 are obtained from the wife. The yellow dashed boxes mark trips from different drivers with very high correlation. Green boxes mark trips from the same driver.

For Household 2, where one of the driver is driving the car far less, we use clustering for one-class learning. We collected 34 trips from the primary driver and 3 trips from the secondary driver. The cluster is created using 31 random trips from the primary driver, and the remaining trips are used to calculate the distance from the centroid of the cluster. If the distance is above an empirical pre-calculated threshold, then that trip is registered as a new driver trip. Since each trip from the primary driver could have some outliers, we tried to cover all cases by using randomly selected trips to create the cluster. Averaging over 10 iterations, we obtain 73.3% accuracy in classifying test trips, as shown in Fig 16(b). *We observe that the correlation between drivers from a household who use a vehicle is higher compared to professional drivers, and the day to day variation much higher.*

6 RELATED WORK

In the realm of driver distinction, Choi et. al [1] show that driver behavior can be modeled using steering angle, brake status, acceleration status and vehicle's speed that are collected from vehicle's CAN bus. The driver identification part resembles our methodology however since they are only using longitudinal data collection, the accuracy of driver identification is less than 35% with their system. In another driver distinction work, Enev et. al [5] could manage to identify 15 drivers with 100% accuracy using their longitudinal behaviors. Their system needs at least 15 minutes of training dataset for each user, that includes braking patterns, vehicle speed, acceleration, throttle position etc.

In another work, Miyajama et. al [14] show that up to 276 drivers could be identified with 76.8% accuracy using gas/brake pedal usage, engine speed, steering wheel angle and car following distances. Their system works reasonably well for 276 drivers using longitudinal behaviors but their system also needs 5 minutes of training dataset for each user. With our proposed design, we can attain up to 95% accuracy in under 30 seconds.

Driver identification is also investigated by Riener et. al [18]. The authors used sitting postures to distinguish drivers using a pressure pad. However, privacy of the collected data is of utmost importance. And while a fair amount of related work proposes to use speed and location coordinates for privacy preservation, Krumm et.al [13] have proved that location traces can be used to successfully identify individuals. Gao et. al [6] have also demonstrated that speed is not a privacy preserving parameter and is enough to track a driver.

Researchers have been exploring ways to maintain driver privacy and anonymity, by masking identifying data [10]. Hoh et. al [8] have proposed using virtual trip lines to maintain driver privacy. In another work, Zan et.al [25] guarantee a high degree of anonymity by using a zone-aware path cloaking scheme. Another approach has been investigated by researchers in [2], [3], [19] and [20] with the usage of a Trusted Third Party to perform cryptographic operations.

Modeling and predicting human behavior has been investigated in another work by Pentland et.al [17]. The authors achieved 95% accuracy at predicting automobile driver's actions from their initial preparatory movements. However, the algorithm can only determine when a car will be passing another, turning or following the previous cars in next couple of seconds. There has been much work on systems for traffic monitoring, rather than driver monitoring both in commercial companies and research facilities. Many of them leverage GPS units on cars(OnStar[16] system) to track the vehicle's movements and analysis can be done at the server. The Nericell[15] project concentrates on the road topology and shows that potholes, bumps and braking can be detected by using accelerometer and GPS sensors of the mobile phones. In another work [21], the authors show how driver's behavior under critical circumstances(sudden breaks, extreme steering angle rotations) varies compared to regular times, using smartphones. In contrast to smartphone sensor based techniques, we use vehicle sensors for more reliable measurements.

7 DISCUSSION

We have presented the design, implementation and evaluation of a driver differentiation technique using only pre-trip events. Unlike previous work, that focuses on parameters collected during driving, such as speed, most visited locations, etc., our work aims at sensing pre-trip events, that occur before the driver starts driving. Parameters monitored during the drive are a reflection of the driving style, but are greatly influenced by external driving conditions, such as traffic. Moreover, collection of values such as speed and most visited locations is detrimental to user privacy, as has been shown in previous work [6, 13]. Our proposed system focuses on sensing events that happen before a person starts driving. All these events are common acts that any driver conducts before a drive, such as closing the door, fastening the seatbelt, turning the ignition on etc. It is, however, the sequence and interval of these simple events that presents some inherent habits that differ widely from person to person.

While this study has focused primarily on professional drivers, it is also an interesting question whether the results hold in family settings with multiple drivers. It is possible that random events of daily life lead to more variability in startup routines, which would make identification more challenging. It is also possible, however, that startup routines are more diverse across drivers within a family, because of greater differences in driving experience and trip purpose than in our professional drivers who all drove to deliver mail. The latter could lead to improved differentiation results.

In sensing these events, we employ the innocuous sensors that are already present on most vehicles these days. One might claim that adjusting mirrors, seat settings can also be used to differentiate drivers. Unfortunately, in our test vehicles, mirror positions are adjusted manually and seat positions could not be retrieved from the dongle. The dongle is customized and can work with most of the GM brand vehicles. Therefore, these settings cannot be used to distinguish the drivers. With the grant of accessing more sensors by vehicle companies, a better performance at distinguishing drivers could be achieved.

Depending on the application scenario, having an accuracy less than 100% may not be enough. But it must be noted that this 90% accuracy is achieved by using pre-trip data. For higher accuracy, we could always include additional driving and telemetric data such as vehicle speed, engine's RPM and steering wheel angle.

One might argue that the described system could be implemented as an app executing on the phone. Vehicular platforms, with their rising capabilities as cyber-physical systems, and large amounts of real-time data are uniquely positioned to measure, process, configure, and manage in-vehicle sensor data for applications such as driver differentiation. Vehicles are not as resource-constrained as smartphones, that are also projected as mobile edge nodes. With more and more connected vehicles, in-vehicle sensor data can be collected anonymously over a diverse range of drivers, and processed in remote cloud servers to generate large-scale analytics on driving behaviors and practices.

There are several potential privacy implications of this result. In one scenario, the result suggests that pre-trip data is potentially useful in re-identifying drivers in anonymous in-vehicle datasets. Storing pre-trip data together with anonymous streams of privacy sensitive data is therefore undesirable. It is worth noting, however, that re-identifying a driver in an anonymous dataset would require pre-trip profiles labeled with driver names, which are not always straightfoward to obtain. In another scenario, pre-trip data such as the timing of door closing events will likely be considered less privacy sensitive than driving data, which can be linked to visited locations, driving speeds, or aggressive driving styles. If the choice is between collecting driving data or pre-trip data for driver differentiation, pre-trip data therefore presents an opportunity to differentiate drivers using a less sensitive source of data.

8 CONCLUSION

We have shown that pre-trip in-vehicle data are particularly distinctive and represent a minimal set of in-vehicle data for driver differentiation. Driver differentiation based on pre-trip data requires only 20 seconds of data, while driving telemetry data approaches require about 10 minutes of data to reach a comparable accuracy. Specifically, we have shown that drivers can be distinguished using only pre-trip vehicle sensor data from the CAN bus with an accuracy of 91% in a real-world dataset of 480 trips collected over five mail vans with five drivers. In a controlled experiment, where all drivers steer the same vehicle along the same route, we have also shown that up to 16 drivers can be distinguished with similar accuracy. This accuracy can be further increased by combining it with driving telemetry data. It is also worth noting that the real-world mail van experiment was performed with mass-market vehicles that are close to 10 years old. With a larger number of sensors and electronic control systems in newer vehicles and luxury vehicles, one can expect even higher accuracy. We also validate our results for members of a household. The study has been focused on distinguishing drivers from the same vehicle. Whether a driver profile also holds across multiple vehicle models remains an open question.

ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under Grant No CNS-1329939.

REFERENCES

[1] SangJo Choi, JeongHee Kim, DongGu Kwak, Pongtep Angkititrakul, and John HL Hansen. Analysis and classification of driver behavior using in-vehicle can-bus information. In *Biennial Workshop on DSP for In-Vehicle and Mobile Systems*, pages 17–19, 2007.

[2] Tim Churches and Peter Christen. Blind data linkage using n-gram similarity comparisons. In *Advances in Knowledge Discovery and Data Mining*, pages 121–126. Springer, 2004.

[3] Tim Churches and Peter Christen. Some methods for blindfolded record linkage. *BMC Medical Informatics and Decision Making*, 4(1):9, 2004.

[4] Steve Corrigan. Introduction to the controller area network (can). *Application Report*, 2008.

[5] Miro Enev, Alex Takakuwa, Karl Koscher, and Tadayoshi Kohno. Automobile driver fingerprinting. *Proceedings on Privacy Enhancing Technologies*, 2016(1):34–50, 2016.

[6] Xianyi Gao, Bernhard Firner, Shridatt Sugrim, Victor Kaiser-Pendergrast, Yulong Yang, and Janne Lindqvist. Elastic pathing: Your speed is enough to track you. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '14, pages 975–986, New York, NY, USA, 2014. ACM.

[7] Philippe Golle and Kurt Partridge. On the anonymity of home/work location pairs. In *Pervasive computing*, pages 390–397. Springer, 2009.

[8] B. Hoh, T. Iwuchukwu, Q. Jacobson, D. Work, A.M. Bayen, R. Herring, J.-C. Herrera, M. Gruteser, M. Annavaram, and J. Ban. Enhancing privacy and accuracy in probe vehicle-based traffic monitoring via virtual trip lines. *Mobile Computing, IEEE Transactions on*, 11(5):849–864, May 2012.

[9] Baik Hoh, Marco Gruteser, Hui Xiong, and Ansaf Alrabady. Preserving privacy in gps traces via density-aware path cloaking. *Proceedings of CCSfi07*, 2007.

[10] Shubham Jain and Janne Lindqvist. Should I protect you? Understanding developers' behavior to privacy-preserving APIs. In *Workshop on Usable Security 2014*, 2014.

[11] Yurong Jiang, Hang Qiu, Matthew McCartney, William G. J. Halfond, Fan Bai, Donald Grimm, and Ramesh Govindan. Carlog: A platform for flexible and efficient automotive sensing. In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, pages 221–235, New York, NY, USA, 2014. ACM.

[12] Cagdas Karatas, Luyang Liu, Hongyu Li, James Liu, Yan Wang, J Yang, Yingying Chen, Marco Gruteser, and Rich Martin. Leveraging wearables for steering and driver tracking,. In *IEEE International Conference on Computer Communications (Infocom) 2016.* ACM, 2016.

[13] John Krumm. Inference attacks on location tracks. In *Proceedings of the 5th International Conference on Pervasive Computing*, PERVASIVE'07, pages 127–143, Berlin, Heidelberg, 2007. Springer-Verlag.

[14] Chiyomi Miyajima, Yoshihiro Nishiwaki, Koji Ozawa, Toshihiro Wakita, Katsunobu Itou, Kazuya Takeda, and Fumitada Itakura. Driver modeling based on driving behavior and its evaluation in driver identification. *Proceedings of the IEEE*, 95(2):427–437, 2007.

[15] Prashanth Mohan, Venkata N. Padmanabhan, and Ramachandran Ramjee. Nericell: Rich monitoring of road and traffic conditions using mobile smartphones. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, SenSys '08, pages 323–336, New York, NY, USA, 2008. ACM.

[16] OnStar. OnStar by GM. http://openxcplatform.com/, 2015. Online; accessed 2015-12-5.

[17] Alex Pentland and Andrew Lin. Modeling and prediction of human behavior. *Neural Computation*, 11:229–242, 1995.

[18] Andreas Riener and Alois Ferscha. Supporting implicit human-to-vehicle interaction: Driver identification from sitting postures. In *The First Annual International Symposium on Vehicular Computing Systems (ISVCS 2008)*, page 10, 2008.

[19] Monica Scannapieco, Ilya Figotin, Elisa Bertino, and Ahmed K Elmagarmid. Privacy preserving schema and data matching. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 653–664. ACM, 2007.

[20] Rainer Schnell, Tobias Bachteler, and Jörg Reiher. Privacy-preserving record linkage using bloom filters. *BMC medical informatics and decision making*, 9(1):41, 2009.

[21] Heikki Summala. Automatization, automation, and modeling of driver's behavior. In *Recherche - Transports - Scurit*, pages 35–45. Elsevier, 2000.

[22] The OpenXC. The OpenXC Platform. http://openxcplatform.com/, 2015. Online; accessed 2015-12-5.

[23] Yan Wang, Jie Yang, Hongbo Liu, Yingying Chen, Marco Gruteser, and Richard P Martin. Sensing vehicle dynamics for determining driver phone use. In *Proceeding of the 11th annual international conference on Mobile systems, applications, and services*, pages 41–54. ACM, 2013.

[24] Jie Yang, Simon Sidhom, Gayathri Chandrasekaran, Tam Vu, Hongbo Liu, Nicolae Cecan, Yingying Chen, Marco Gruteser, and Richard P Martin. Detecting driver phone use leveraging car speakers. In *Proceedings of the 17th annual international conference on Mobile computing and networking*, pages 97–108. ACM, 2011.

[25] Bin Zan, Peng Hao, M. Gruteser, and Xuegang Ban. Vtl zone-aware path cloaking algorithm. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 1525–1530, Oct 2011.