Towards a Real-Time Driver Identification Mechanism Based on Driving Sensing Data

Sasan Jafarnejad
Interdiscipl. Centre for Security, Reliability & Trust (SnT), Univ. of Luxembourg, Esch-sur-Alzette, Luxembourg
Email: sasan.jafarnejad@uni.lu

German Castignani
Interdiscipl. Centre for Security, Reliability & Trust (SnT), Univ. of Luxembourg, Esch-sur-Alzette, Luxembourg
Email: german.castignani@uni.lu

Thomas Engel
Interdiscipl. Centre for Security, Reliability & Trust (SnT), Univ. of Luxembourg, Esch-sur-Alzette, Luxembourg
Email: thomas.engel@uni.lu

***Abstract*—The growing penetration of telematics systems and connectivity in vehicles has enabled a large variety of possible value-added services for drivers and service providers. In particular telematics based real-time driver identification is in interest of entities such as insurance companies, car rentals and public transportation fleet managers. We propose a mechanism for driver identification based on driving dynamics signals currently available in production cars. The system collects and filters sensing data in a sliding window iteration, computes statistical and spectral features and finally provides driver identification for each window frame through a classification process. Finally, a decision function takes individual predictions and outputs a single prediction for the ongoing trip. We evaluate the impact of various elements of the process on identification accuracy, including sliding window size, classifier algorithms and feature sets. Results show that complementing gas pedal signal with steering wheel cepstral analysis improves identification accuracy by 22.4%. We also show that Boosting classifiers provide better predictions for our problem and the best results have been achieved using AdaBoost with $95, 89, 82$ percent accuracies for $5, 15, 35$ drivers respectively. In terms of real-time identification performance, the proposed system is able to correctly identify 75% of the drivers in less than 65 s in a 5 drivers scenario.**

***Index Terms*—driver identification, driver modeling, machine learning, driving behavior, real-time driver modeling.**

I. Introduction

The growing penetration rate of telematics systems and connectivity in vehicles has enabled a large variety of possible value-added services for drivers and service providers. In particular, being able to identify a driver in real-time based on telematics data is of interest. More specifically, insurance and car rental companies might be interested in fast detection of stolen cars or undeclared secondary drivers (which commonly are charged extra within the base pricing). Public fleet managers might also be interested in verifying that the attribution of drivers to different vehicles is respected as planned.

Very recently, research has been conducted by proposing different feature sets to characterize and identify a driver. In particular, those features are related to vehicle dynamics parameters collected through a telematics system. Such a telematics system data consists of vehicle parameters that might be available through an On-Board Diagnostics (OBD) connection that provides, for instance: speed, RPM, pedals pressure, fuel efficiency, following distance, among others. Moreover, for a driver identification mechanism, focus has to be put on defining features that are based on data that the driver can directly control so that the influence of the driving context (i.e., road topology, weather, visibility) is minor. As an example, some researchers [1] [2] have focused on the spectral analysis of accelerator and brake pedal signals, which has shown good identification performance for a limited set of drivers.

In this paper, we propose to extend the methodology of spectral analysis to other telematics signals and we propose to validate the model using a large driving dataset called UYANIK [3] consisting 105 people driving a single car along a pre-defined route. In terms of the features used in the proposed mechanism, we have exclusively focused on features that are computed from sensors that are already available in market vehicles in order to guarantee deployability. Since we want this mechanism to provide driver identification in real-time, we propose to run a classification algorithm in a time regular basis and to apply a decision function to define the driver's identity. In this paper we present an evaluation of the performance of the proposed mechanism in terms of accuracy of the decision making and detection speed since the beginning of the trip.

The remainder of the paper is structured as follows. In Section II we present the related work on driver identification. Then in Section III, we introduce and characterize the dataset used for the evaluation of the proposed mechanism. In Section IV, we present the proposed feature set, the classification algorithms and decision-making rules to identify drivers. Then, in Section V we present evaluation results and corresponding discussions. And finally we conclude and give directions for future work in Section VI.

II. Related Work

A number of studies on driver identification relevant to our work have been proposed so far. In one of the earliest works Wakita et al. propose to identify drivers using behavioral signals captured during car-following task [4]. They compare identification performance of Gaussian Mixture Model (GMM) against physical models. In terms of features, they use following distance (FD), vehicle speed (VS), brake and gas pedal pressures. They achieve an identification rate of

73% using GMM when considering thirty drivers. Miyajima et al. [1] proposed a follow up on Wakita's work using the same dataset for evaluation. The key idea of this study is the addition of features based on spectral analysis of gas and brake pedals. The authors showed an identification rate of 76.8% using GMM for a field test with 276 drivers. This study presents good results, but they were achieved using data from sensors that is not available on commercial vehicles.

There are also a number of contributions in the literature that make use of the same dataset that we use in this paper (UYANIK). The first of them is by E. Özturk et al. [5], which follows the same approach as the work presented above. However the highest accuracy achieved is 57.39% for 23 drivers, which is far lower than the reported accuracy by Miyajima et al. (76.8%). This large gap in accuracy maybe explained by the low sampling rate of sensor signals in the UYANIK dataset. Del Campo et al. [6] conduct a similar study as Özturk et al., but using a methodology based on multilayer perceptrons and focusing on real-time identification. For a group of three drivers they achieve an accuracy of 84%. In a follow-up work, Martìnez et al. [2] use extreme learning machine as their classifier for driver identification. In compare to other works mentioned above here features are extracted from larger set of signals from CAN-Bus and IMU (Inertial Measurement Unit). Then a feature selection stage selects the more relevant features for identification. This results in 90% accuracy for three drivers.

In a recent work, Enev et al. [7] present a comprehensive work on driver fingerprinting. Although their focus is on the security and privacy implications of driver identification, they also achieve good identification results. For fifteen drivers they achieved 100% accuracy. Since they extract statistical and spectral features from 15 driving signals, their methodology is computationally expensive. Other important findings of this work is the very high importance of sensors such as brake pedal and engine torque for driver identification.

TABLE I
SENSOR DATA AVAILABLE IN UYANIK

Channel	Source	Details
Video facing the driver	Retrofitted	15 fps 480x640
Video facing the road	Retrofitted	15 fps 480x640
Driver close-talking microphone	Retrofitted	16 KHz 16-bit
Rear-view microphone	Retrofitted	16 KHz 16-bit
Cellular phone microphone	Retrofitted	16 KHz 16-bit
Steering wheel angle (SWA)	CAN-Bus	32 Hz degrees
Steering wheel relative speed	CAN-Bus	32 Hz degrees/second
Vehicle speed (VS)	CAN-Bus	32 Hz km/h
Individual wheel speeds	CAN-Bus	32 Hz km/h
Engine RPM (ERPM)	CAN-Bus	32 Hz, rpm
Yaw rate (YR)	CAN-Bus	32 Hz
Clutch state	CAN-Bus	32 Hz, 0/1 state
Reverse gear	CAN-Bus	32 Hz, 0/1 state
Brake state	CAN-Bus	32 Hz, 0/1 state
Clutch	CAN-Bus	32 Hz, 0/1 state
Brake pedal pressure sensor	Retrofitted	Kg-force per cm^2
Gas pedal pressure sensor	Retrofitted	Kg-force per cm^2
XYZ directional accelerations	IMU	10 Hz
Laser rage-finder	Retrofitted	1-2 Hz, 181°, cm

III. DATASET

We consider the UYANIK dataset, which has been collected under the shared framework of Drive-Safe Consortium (Turkey) and NEDO (Japan) International Collaborative Research [3][8]. Its main application focus is driving behavior signal processing, more specifically, applications ranging from driver identification to driving environment personalization and driver assistance among others. For UYANIK, a Renault Megane was equipped with an adequate set of sensors to collect driving data. These sensors include three microphones to capture ambient sound in the cabin and cameras to capture video facing both the driver and the road. There is a laser range-finder placed in front of the vehicle to scan the distance to obstacles. An IMU measures the car's dynamics, and a GPS receiver keeps track of vehicle's location. Furthermore pressure sensors are retrofitted under gas and brake pedals to accurately record pedal actuations. Further, CAN-Bus data are recorded as well, which contains data from several vehicle sensors, ranging from engine speed to steering wheel angle. The complete list of available sensor signals is presented in Table I. Data collection is done in Turkey, and consists of a 25 km long path which includes a short ride inside university campus, a city traffic driving, motorway traffic driving, a dense city traffic driving and, finally, the way back to the point of departure. A typical trip lasts about 45 minutes.

The UYANIK dataset is invaluable to researchers but it has some shortcomings. For example, not all the recordings include brake and gas pedal sensors and the ones that do are frequently very noisy therefore unusable. In order to ensure more realistic conditions, we discard data from external pressure sensors (gas and brake pedals) as well as the laser range-finder and the IMU. We have performed a pre-processing step to ensure that driving sessions containing a high proportion of corrupt data are discarded. Then in order to have more balanced dataset, we discard recordings that are too short or too long. Such anomalies caused either due to technical difficulties or external factors such as severe traffic jams. Table II shows a summary of dataset before and after pre-processing and filtering.

IV. IDENTIFICATION MODEL

A. Methodology

The goal of driver identification is to associate a driving trace x to its corresponding driver y. We limit the target drivers to a finite set and approach driver identification as a supervised classification problem. In this work we only consider the case that we have information on all drivers, for example when a car is shared among family members. Suppose

TABLE II
SUMMARY STATISTICS OF DATASET

Dataset	Trip Length (Minutes)			Participants		
	Min	Max	Avg (Std)	Male	Female	Total
Entire dataset	20.30	83.90	42.75 (9.79)	81	12	93
Selected subset	34.37	52.52	42.56 (4.87)	59	8	67

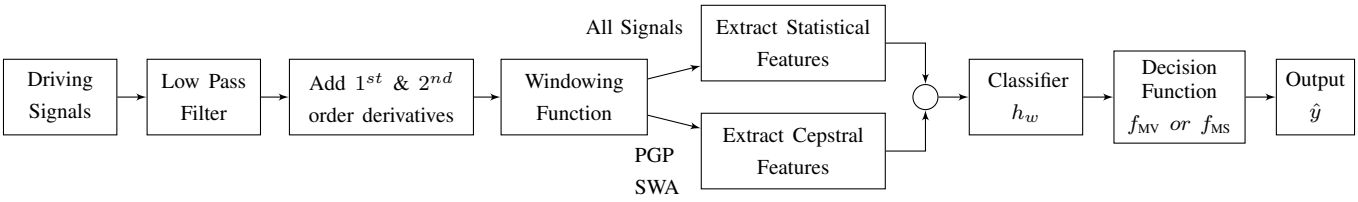


Fig. 1. Driver Identification Process

$C = \{d_1, d_2, \cdots, d_c\}$ denote a finite set of drivers (e.g. family members), then we assume any driving trace is done by a driver in C ($y \in C$). We search for a function h which once trained is able to predict the corresponding driver ($\hat{y}$) for any unseen driving trace x, knowing that $\hat{y} \in C$. Since x is a sequence, classical supervised machine learning (ML) methods are not directly applicable, therefore we apply concept of *sliding windows* to make this possible [9]. We use *window classifier* h_w to map each window frame of length w into individual predictions. Let $d = (w-1)/2$ be the half-width length of window, then for a window frame at time t, h_w makes prediction $\hat{y}_t$ based on window $\langle x_{t-d}, \cdots, x_t, \cdots, x_{t+d} \rangle$. Due to high sampling rate of driving signals it is not feasible to make predictions for every window frame. Therefore we make window predictions for every $k = \lfloor w * (1 - \frac{r}{100}) \rfloor$ samples, where k denote *step size* and r denote the percentage of overlap between two consecutive windows. This results in $N = \frac{T}{k}$ examples. To put this into context, each driving trace (x, y) is converted into window frames, then h_w is trained using feature vector $\mathbf{x}$ computed for each window and original label y. Similarly to classify an unseen driving trace x, it is converted into N window frames, for each window frame feature vector $\mathbf{x}_i$ is computed and h_w makes prediction $\hat{y}_i$ based on $\mathbf{x}_i$. Finally $\hat{\mathbf{y}}$ results from concatenation of all $\hat{y}_i$. We require to assign x to only one driver therefore we define $\hat{y} = f(\hat{\mathbf{y}})$ as *decision function* which maps individual window predictions to one single prediction for the whole driving trace.

Multiclass Classification: Since *driver identification* is performed among more than two drivers (generally a multiclass problem) and many classification algorithms are inherently binary, we choose the one-vs.-one (OvO) approach to convert binary classifiers into multiclass classifiers.

Evaluation Method: In order to address applications with different requirements we perform evaluations for three driver set sizes of $c \in \{5, 15, 35\}$. Identification among drivers with similar driving styles is more challenging therefore we repeat for each c the evaluation $M = 10$ times and each time on a random subset C_m of all drivers C, where $C_m \subset C$ and $|C_m| = c$. For each instance C_m in order to utilize the entire driving traces $D_m = \{(x_i, y_i)\}_{i=1}^{|C_m|}$ for both training and testing, we employ a five fold cross-validation method. Each driving trace (x_i, y_i) is sliced into 5 segments of equal length and evaluations are performed under leave-one-segment-out train and test scheme. Therefore at each fold one slice from all driving traces is kept out of training process, then h is scored based on its prediction performance over the remaining slice. We use accuracy to score h at each fold, which is defined as below:

$$\alpha' = \frac{1}{c} \sum_{i=1}^{c} \mathbb{1}(y_i = \hat{y}_i) \tag{1}$$

where $\mathbb{1}$ is the indicator function, $\hat{y}_i$ and y_i are respectively prediction and true driver for i^{th} driving trace.
The overall accuracy score for h is then obtained by calculating the mean accuracy of all cases as follows:

$$h_{score} = \frac{1}{M \cdot L} \sum_{m=1}^{M} \sum_{l=1}^{L} \alpha'_{m,l} \tag{2}$$

where $L = 5$ is number of cross-validation folds and $M = 10$ is number of repetitions. In the rest of paper we refer to this score as accuracy.

TABLE III
SELECTED SIGNALS FOR DRIVER IDENTIFICATION

Sensor	Derivations 1st	Derivations 2nd	Cepstral
Percentage Gas Pedal (PGP)	Yes	Yes	Yes
Steering Wheel Angle (SWA)	Yes	Yes	Yes
Vehicle Speed (VS)	Yes	Yes	No
Engine RPM (ERPM)	Yes	Yes	No
Yaw Rate (YR)	Yes	Yes	No

B. Pre-processing and Feature Extraction

Two kinds of signals are used for feature extraction: 1) Original signals 2) Derived signals. Original signals are the original sensor values from CAN-Bus, while derived signals are the first and second derivative of original signals. To motivate this choice let us take vehicle speed as an example: its first and second derivative represent longitudinal acceleration and jerk, which have shown to be representative of particular driving styles [10], therefore these signals are useful for discriminating between drivers.
Figure 1 shows the various stages that the original signals go through before the feature extraction step. In this study we use five original signals which are listed in Table III. All these signals are obtained from CAN-Bus. First a low-pass filter (FIR) is applied to original signals to remove high frequency noise and smoothen the signals. Then for each signal we use the Savitzky-Golay [11] filter to derive their first and second order derivatives and add two additional signals. Then, all the signals go through a windowing function which takes two parameters, w to determine the length of window and r to specify amount of overlap between two consecutive windows. At this point each signal is broken down into windows of size w and is ready to supplied into feature extraction stage.

Two kinds of features are extracted from each window:

Statistical features: A set of descriptive statistics is selected to be most representative of the distribution of sensor values covered by the window. This set includes minimum, maximum, mean, median, standard deviation, kurtosis, skewness. This category of features is extracted from all the signals.

Spectral features: Studies have shown that cepstral analysis is suitable for driver identification [1] [5] [6]. Cepstral analysis has various applications including in speech and speaker recognition [12]. In our implementation we multiply a Hamming window of length w to limit the signal and also reduce the edge effects. We define $c(k)$ to be the first k cepstral coefficients and higher order coefficients are discarded. Equation to derive $c(k)$ is:

$$c(k) = \left|\mathcal{F}^{-1}\left\{\log(|\mathcal{F}\left\{x_t w_t\right\}|)\right\}\right| \tag{3}$$

Where x_t denote signal values covered by current window, w_t denote Hamming window with the same length as window and $\mathcal{F}$ and $\mathcal{F}^{-1}$ denote discrete Fourier transform (DFT) and inverse-DFT (IDFT) respectively. We extract spectral features from two signals, gas pedal position (PGP) and the steering wheel angle (SWA) and keep the first $k = 32$ cepstral coefficients as features.

C. Classification Algorithms

We select five classification algorithms to be used as *window classifier* h_w. We benchmark them and pick the best performing algorithm to conduct further experiments. In our selection we favored ensemble methods because it has been shown that they generally achieve better predictive performance [13]. The following is the list of classification algorithms used in this study along with their corresponding parameters.

- **AdaBoost (AB)** - 500 decision trees as weak learners, learning rate = 0.75
- **Gradient Boosting (GB)** 500 estimators, maximum depth = 6
- **Random Forest (RF)** 500 estimators, maximum depth = 6, minimum samples per leaf node = 2
- **Extra Trees (ET)** 500 estimators, maximum depth = 8, minimum samples per leaf node = 2
- **Support Vector Machine (SVM)** linear kernel, C = 0.025

For ensemble learners in order to have a balance between prediction performance and execution time we choose 500 as number of weak learners/trees. In our experiments we use the implementations from the scikit-learn software package, all other parameters are set to their default values as of scikit-learn version 0.18.1 [14].

D. Decision Functions

The decision function f determines the final prediction $\hat{y}$ based on window predictions $\hat{\mathbf{y}}$. We define for each window k a vector $\hat{\mathbf{y}}_k = (\alpha_1 \ \cdots \ \alpha_c)$ where $\alpha_{k,j}$ corresponds to the classification score on the window k for driver j. Below we introduce two decision functions to obtain $\hat{y}$ and evaluate their performance later in the paper:

1) Majority vote (MV): For each temporal window k, we find the identifier of the driver with largest classification score $I_k = \arg\max \alpha_i$. Then We assign to each temporal window a ranking score vector $\mathbf{RS_k} = \left(\beta_1^k, \cdots, \beta_c^k\right)$ where $\beta_{I_k}^k = 1$ and all other $\beta_i^k = 0$. Finally $\hat{y}$ is the identifier of the driver who most frequently obtains the highest classification score, i.e., $\hat{y} = \arg\max_i \sum \beta_i^k$

2) Maximum score (MS): We set $\hat{y}$ to be the identifier of the driver that has the largest cumulative classification score, i.e., $\hat{y} = \arg\max_i \sum_{k=1}^{N} \alpha_i$

To motivate the introduction of the two decision functions let us take the following example which consists of classification scores of three drivers for five windows:

$$(\hat{y}_k)_{k=1}^5 = \begin{pmatrix} 0.15 & 0.40 & \mathbf{0.45} \\ 0.18 & 0.39 & \mathbf{0.43} \\ 0.27 & 0.34 & \mathbf{0.39} \\ 0.06 & \mathbf{0.73} & 0.21 \\ 0.09 & \mathbf{0.81} & 0.1 \end{pmatrix}$$

In MV we simply count how many times each driver has the highest score and choose the most frequent as prediction:

$$\hat{y} = f_{\text{MV}}((\hat{y}_k)_{k=1}^5) = \arg\max_i (0 \ 2 \ \mathbf{3}) = d_3$$

MV is sensitive to miss-classifications at h_w. Suppose d_2 is the true driver, for the first three windows d_2 and d_3 are scored very closely but at window level choice is always with d_3. Looking at the next two windows and the whole picture d_2 is more likely to be the correct prediction. Next we compute the prediction based on MS:

$$\hat{y} = f_{\text{MS}}((\hat{y}_k)_{k=1}^5) = \arg\max_i (0.75 \ \mathbf{2.67} \ 1.57) = d_2$$

While MV falls short at this example MS handles such instances more robustly.

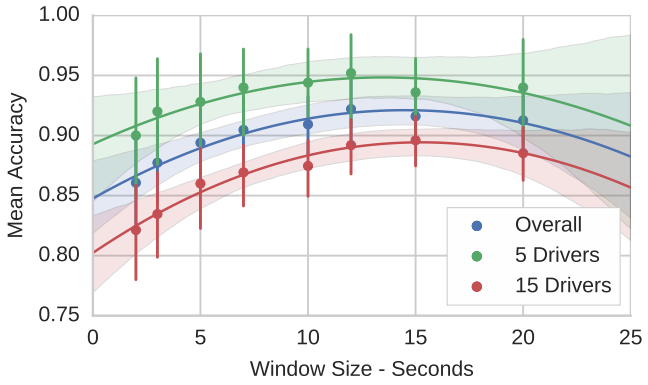
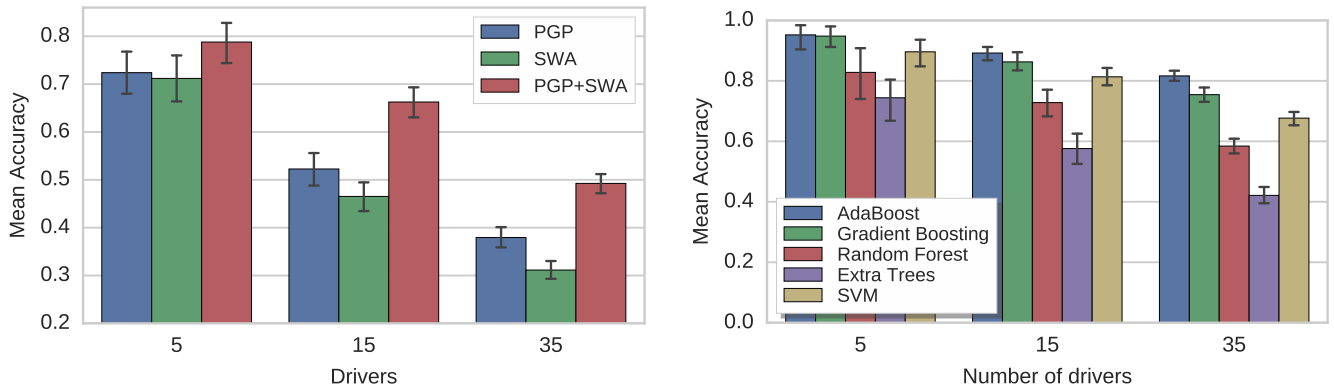


Fig. 2. Window Size Comparison

V. Identification Performance

In this section we present results of our experiments and their corresponding discussions. Each subsection is focused on a certain component of the proposed method, therefore in order to avoid repetition unless otherwise is stated experiments are performed using configurations introduced in Section IV, with AdaBoost as h_w, MS as f, $w = 12s$ and $r = 20\%$.



(a) Feature Benchmark

(b) Classifier Benchmark

Fig. 3. Identification Performance

A. Optimum Window Size

Since feature vectors are computed on window frames, a key parameter in our method is the choice of w and r (overlap percentage). We perform experiments for window sizes[1] from 2 to 50 seconds, with r fixed at 20%. According to results shown in Fig. 2, as w increases accuracy also improves until reaching its peak at 12 seconds then starts to decline. Moreover, not only we observe a decline in accuracy, but also that larger windows result in larger latencies (at this case with fixed r). In practice a small window results in larger number of training examples which leads to larger computational expenses for both training and prediction. Moreover each window frame may not contain the entire duration of driving events such as a turn and over-take. On the other hand longer windows results in fewer examples therefore are computationally cheaper. We do not investigate impact of window overlap (r). However, larger overlaps increase the number of window frames which potentially leads to better identification performance, but greatly increases computations.

Enev et al. [7] perform a similar experiment but they conclude that 3s results in the best performance, while by our experiments 12s yields the best results. We hypothesize that this large difference is caused by the chosen features; some features are most informative when computed over longer periods while others not, therefore it is expected that optimum window size differs for various feature sets. The choice of window size and overlap size is highly application dependent. For example in case a fleet manager wants to validate the actual driver for a long trip a fairly large window size and small overlap would be sufficient. On the other hand if it is desired to identify the driver as quickly as possible, for example to customize the driving experience, then having a large overlap and reasonably small window size is preferable.

B. Feature Comparison

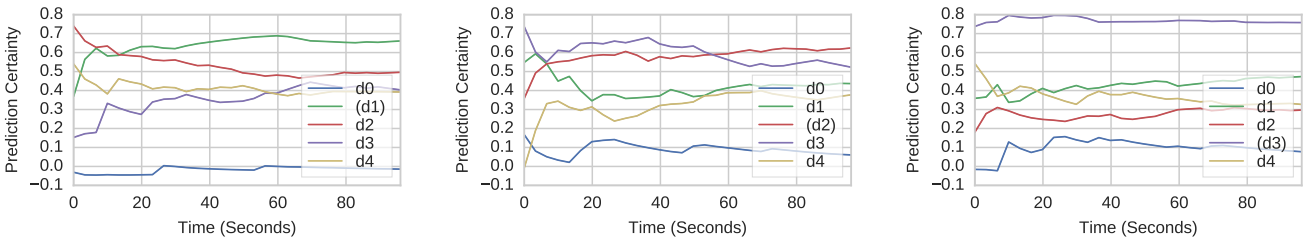
Some works in the literature such as [1], [5], [6] show that when cepstral analysis is applied to gas and brake pedal pressure signals it improves driver identification accuracy. Such signals are obtained by fitting external sensors under the pedals and are not available in production vehicles. Instead we choose PGP and SWA that are available on unaltered vehicles from CAN-Bus. We choose SWA because similar to pedals steering wheel is also operated directly by driver and its movements potentially reflects some unique characteristics of driver.

In order to evaluate the impact of the SWA signal on driver identification, we consider three cases. First we limit the features only to spectral analysis of PGP, then only to spectral analysis of SWA, and finally both signals. Results are depicted in Fig. 3a, as one can see when each signal is used alone, PGP yields better results than SWA (1.4 to 22.58% improvement), however when used together results significantly improve (8.86 to 22.44%) from the best single signal case, this is particularly noticeable when the number of drivers increases. When some of the drivers operate the gas pedal or the steering wheel similar to one another; the use of PGP and SWA at the same time helps the classifier to discriminate the drivers.

C. Classifier Benchmarks

Identification results for various classification algorithms are depicted in Figure 3b. AdaBoost clearly provides the best performance with $95\%, 89\%$ and 82% accuracy for groups of 5, 15, 35 drivers respectively. Runner up classifier is Gradient Boosting with $95\%, 86\%, 75\%$ accuracy. The third best performance is achieved using SVM with $90\%, 81\%$ and 68% accuracy for groups of $5, 15, 35$ drivers. The results show the dominance of Boosting algorithms and relatively weak performance of Random Forest and Extra Trees (Bagging algorithms). Nevertheless, SVM shows fairly good results but noticeably lower than AdaBoost and Gradient Boosting. Superior performance of Boosting learners and particularly AdaBoost can be explained by their underlying mechanics, which consists of combining a large number of weak learners to make a prediction. During the training AdaBoost iteratively identifies misclassified instances and tweaks subsequent learners to predict them correctly. Furthermore both Bagging and Boosting algorithms we used here internally perform a feature

[1] Window sizes of length 2, 3, 5, 7, 10, 12, 15, 20, 30, 40, 50 Seconds



(a) Case I - Target driver d1

(b) Case II - Target driver d2

(c) Case III - Target driver d3

Fig. 4. Prediction over time.

selection step which is not the case for SVM therefore it suffers from sensitivity to training features. The the results achieved using AdaBoost are better than earlier works in the literature, except for [7] in which authors achieve 100% accuracy for 15 drivers, however they use 15 signals from CAN-Bus and perform spectral analysis on all signals, while we only use 5 signals and apply spectral analysis on two of them. It is worth mentioning that in this work we try to keep the study realistic therefore we do not selectively filter out the sections of trip that are not the most informative. For example in [6] authors only use the reference-driving segment of the trips, or in [1] they only consider the times that the car is on the move. Such assumptions although suitable for preliminary studies they do not reflect the real world conditions.

D. Decision Functions

The final stage of the proposed methodology is the decision function f. To investigate performance of each decision function we perform several experiments using MV and MS and compare the results, for these experiments we use AdaBoost, as it was shown to yield the best identification performance. First we consider the case that the entire test-data is used for prediction. Results are summarized in Table IV. We can see in average MS results in 2% to 7% higher accuracy.

In another experiment in order to evaluate how performance changes with increase in amount of test-data. We keep the amount of training data constant ($\frac{4}{5}$ of the entire trip, like earlier experiments) but for test-data, instead of using the remaining $\frac{1}{5}$, we only employ a fraction of test-data and gradually increase the ratio[1] and observe changes in accuracy. Fig. 5a shows the corresponding results. We observe that MS delivers better identification accuracy (up to 16%) for shorter driving traces, as well as for larger number of drivers. This gain in performance comes with a cost, since the implementation of MS assumes calibrated probability outputs [15] from the classifier h_w which may not be provided by the classifier or require additional computational costs to be obtained. On the contrary, MV only requires the prediction labels.

E. Real-time identification

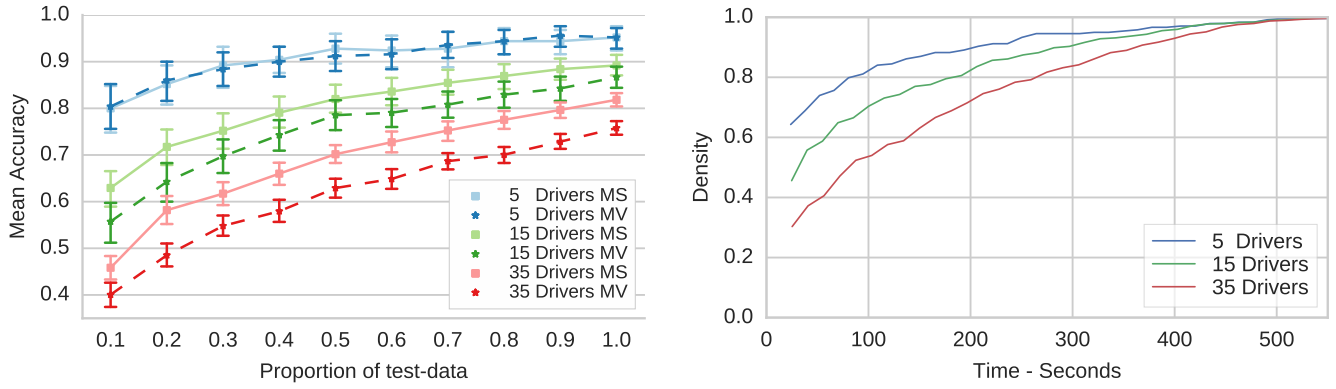
We also investigate how the incremental addition of more data samples impacts the prediction performance. Figure 4 shows three identification cases. In these plots the vertical axis denotes normalized cumulative prediction score for each driver and horizontal axis denote time in seconds. Better prediction score means higher certainty for the corresponding driver therefore as we advance in time and more examples are available for prediction, we ideally expect higher certainty for predicted driver. Case I shows that although for the first $10\,s$, expected prediction could be wrong ($d2$ instead of $d1$) as more samples are collected, the prediction changes in favor of the correct driver. Case II shows a challenging example, as it fails to make the correct prediction before having close to $50\,s$ of driving data (to compute feature vectors from). Case III makes correct prediction right from the beginning and with high certainty. This shows that more data results in gains in identification performance however, at this point it is not clear if this is always the case. To investigate this we refer to an experiment discussed earlier whose results are depicted in Fig. 5a. We can clearly observe that in most cases with increase in amount of test-data (longer driving records) accuracy monotonically increases. For instance in case of 35 drivers, having 5 times more data (use of 50% of test-data instead of 10% for identification) leads to 22% higher overall accuracy (55% increase).

We define *time to detection* as the time it takes to correctly identify a driver. We use this metric to evaluate in average how much driving data would be needed to make a correct identification. The results are presented in Fig. 5b in form of cumulative probability density function (CDF). For 5 drivers, among correctly identified cases (95%) we observe that 75% of them are detected in less than 65. As the number of drivers increase identification task becomes more difficult thus time to detection increases thus for 15 and 35 drivers 75% of drivers are detected in less than about 145s and 228s. These findings are important because for certain applications such as theft detection it is desired to as soon as possible be notified of the incident. Or for a personalized adaptive cruise control (ACC) a short time to detection is required so that before the car gets to a highway, the true driver is already identified.

TABLE IV
IDENTIFICATION ACCURACY - DECISION FUNCTIONS

Decision Function	5 Drivers Avg (Std)	15 Drivers Avg (Std)	35 Drivers Avg (Std)
Majority Vote (MV)	0.95 (0.06)	0.87 (0.05)	0.75 (0.03)
Maximum Score (MS)	0.95 (0.07)	0.89 (0.04)	0.82 (0.02)

[1] We start from 0.1 and increase at 0.1 steps until reach 1.0



(a) Identification performance for variable amounts test-data

(b) CDF of *time to detection*

Fig. 5. Identification Performance

VI. Conclusion

In this paper, we have proposed a mechanism for driver identification based on driving dynamics signals currently available in market cars. In particular, we use simple statistical features from the following 5 signals: PGP, SWA, VS, ERPM, YR and more complex cepstral features only from PGP and SWA. The proposed methodology, collects and filters sensing data in a sliding window iteration, computes statistical and cepstral features and finally provides driver identification for each iteration through a classification process. The driver identification for the on-going driving session is provided through decision functions that take into account the classification scores of previous iterations. We proposed an evaluation study based on a large driving dataset. The results show that adding SWA's cepstral coefficients as features increases identification performance by 22.4% in average. We have shown that Boosting classifiers can provide better predictions in our problem and the best results have been achieved using AdaBoost with $95\%, 89\%, 82\%$ accuracy for $5, 15, 35$ drivers respectively. Moreover, we proved by experimentation that during driving sessions, our methodology composed of classification on sliding windows and cumulative decision functions incrementally improves the performance of the driver identification without declining.

This architecture is practical if: a) enough data is already collected, b) the model is pre-trained on a server. In future work we try to address the above mentioned limitations. To achieve this goal we need to: a) decrease the required computational power and b) fit the model in real-time . To address the first issue we see the need for in-depth feature analysis and feature selection; reducing the dimensions could greatly decrease the computational demand and may also lead to improvements in accuracy. Finally, online or incremental learning can be used in order to fit the model in real-time and more importantly make it evolve as more data are collected.

Acknowledgment

The authors would like to thank Dr. Hüseyn Abut and VPALAB of Sabanci University for providing the dataset.

References

[1] C. Miyajima, Y. Nishiwaki, K. Ozawa, T. Wakita, K. Itou, K. Takeda, and F. Itakura, "Driver modeling based on driving behavior and its evaluation in driver identification," *Proceedings of the IEEE*, vol. 95, no. 2, 2007.

[2] M. V. Martnez, I. D. Campo, J. Echanobe, and K. Basterretxea, "Driving Behavior Signals and Machine Learning: A Personalized Driver Assistance System," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Sep. 2015, pp. 2933–2940.

[3] H. Abut, H. Erdoğan, A. Erçil, A. B. Çürüklü, H. C. Koman, F. Tas, A. Ö. Argunşah, B. Akan, H. Karabalkan, E. Çökelek *et al.*, "Data collection with" uyanik": too much pain; but gains are coming," 2007.

[4] T. Wakita, K. Ozawa, C. Miyajima, K. Igarashi, K. Itou, K. Takeda, and F. Itakura, "Driver identification using driving behavior signals," *IEICE Transactions*, vol. 89-D, no. 3, pp. 1188–1194, 2006.

[5] E. Öztürk and E. Erzin, *Driver Status Identification from Driving Behavior Signals*. New York, NY: Springer NY, 2012, pp. 31–55.

[6] I. del Campo, R. Finker, M. V. Martínez, J. Echanobe, and F. Doctor, "A real-time driver identification system based on artificial neural networks and cepstral analysis," in *2014 International Joint Conference on Neural Networks, IJCNN 2014, Beijing, China, July, 2014*. IEEE, 2014.

[7] M. Enev, A. Takakuwa, K. Koscher, and T. Kohno, "Automobile Driver Fingerprinting," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 1, Jan. 2016.

[8] C. Miyajima, T. Kusakawa, T. Nishino, N. Kitaoka, K. Itou, and K. Takeda, "On-going data collection of driving behavior signals," in *In-Vehicle Corpus and Signal Processing for Driver Behavior*. Springer, 2009, pp. 45–54.

[9] T. G. Dietterich, "Machine learning for sequential data: A review," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2002, pp. 15–30.

[10] F. Sagberg, Selpi, G. F. Bianchi Piccinini, and J. Engström, "A review of research on driving styles and road safety," *Human factors*, vol. 57, no. 7, pp. 1248–1275, 2015.

[11] M. J. E. Savitzky, A. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Anal. Chem.*, vol. 36, 1964.

[12] A. M. Noll, "Cepstrum pitch determination," *The journal of the acoustical society of America*, vol. 41, no. 2, pp. 293–309, 1967.

[13] D. Opitz and R. Maclin, "Popular Ensemble Methods: An Empirical Study," *Journal of Artificial Intelligent Research*, vol. 11, 1999.

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[15] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005, pp. 625–632.