Comparing algorithms for aggressive driving event detection based on vehicle motion data

Comparing Algorithms for Aggressive Driving Event Detection Based on Vehicle Motion Data

Valentina Gatteschi, *Senior Member, IEEE,* Alberto Cannavò, *Member, IEEE,*
Fabrizio Lamberti, *Senior Member, IEEE,* Lia Morra, *Senior Member, IEEE,* Paolo Montuschi, *Fellow, IEEE*

***Abstract*—Aggressive driving is one of the main causes of fatal crashes. Correctly identifying aggressive driving events still represents a challenge in the literature. Furthermore, datasets available for testing the proposed approaches have some limitations since they generally (a) include only a few types of events, (b) contain data collected with only one device, and (c) are generated in drives that did not fully consider the variety of road characteristics and/or driving conditions. The main objective of this work is to compare the performance of several state-of-the-art algorithms for aggressive driving event detection (belonging to anomaly detection-, threshold- and machine learning-based categories) on multiple datasets containing sensors data collected with different devices (black-boxes and smartphones), on different vehicles and in different locations. A secondary objective is to verify whether smartphones could replace black-boxes in aggressive/non-aggressive classification tasks. To this aim, we propose the AD^2 (Aggressive Driving Detection) dataset, which contains (i) data collected using multiple devices to evaluate their influence on the algorithm performance, (ii) geographical data useful to analyze the context in which the events occurred, (iii) events recorded in different situations, and (iv) events generated by traveling the same path with aggressive and non-aggressive driving styles, in order to possibly separate the effects of driving style from those of road characteristics. Our experimental results highlighted the superiority of machine learning-based approaches and underlined the ability of smartphones to ensure a level of performance similar to that of black-boxes.**

***Index Terms*—Driving behavior, classification, smartphone, black-box, anomaly detection, threshold, machine learning**

I. Introduction

According to WHO's *Global status report on road safety 2018*, road traffic injuries are the leading killer of people aged between 5 and 29 [1]. The majority of traffic accidents is due to human factors, especially to aggressive driving [2]. Monitoring and logging of driving events proved to reduce aggressive driving behaviors and, consequently, avoid 20% of road traffic accidents [3]; this phenomenon is due to the fact that the drivers acquire a better awareness of dangerous maneuvers they perform [4]. A reduction of aggressive driving behavior also results in a lowering of vehicle consumption and gas emissions, as aggressive driving has been estimated to increase fuel consumption by around 40% [5].

In the literature, several approaches have been exploited to characterize driving behaviors. Traditional approaches are based on questionnaires, in which drivers are asked to report their reaction to common driving situations, or to provide some information about their driving style. Although questionnaires proved to be able to detect, e.g., driving anger, which is one of the most relevant predictors of aggressive driving behavior [6], they rely on self-reported data and, hence, could be affected by social desirability bias [7].

Manuscript received XXX, XX, 2021; revised XXX, XX, 2021.

Another approach leverages driving simulators, in which driving-related data are gathered in a controlled environment [8]. A limitation of driving simulations lays in the fact that, due to their artificial nature, they could elicit driving behaviors that might differ from real-world ones. Furthermore, driving simulators could be characterized by high installation costs.

A third approach consists in exploiting vehicle data gathered in real driving conditions [9], [10], [11]. This approach, which is known as a Naturalistic Driving Study, is the one that can provide the most accurate and objective data [12].

Data from real drives can be collected in two main ways. A first way is to rely on black-boxes/in-vehicle data recorders, i.e., ad-hoc hardware installed on the vehicle able to collect its acceleration (through Inertial Measurement Units, IMUs), location and speed (through Global Positioning System, GPS), and possibly gather additional information from the on-board diagnostics (OBD). A second way, which gained increasing attention in the last years, consists in relying on personal mobile devices. The advantage of smartphones and similar devices is that, thanks to the sensors they are already equipped with, could be used in a way similar to black-boxes for collecting motion data without additional cost for the drivers; furthermore, they have access to communication networks needed for data transfer (possibly with flat Internet plans) and can even process data onboard, being equipped with processors that are generally more powerful than those in black-boxes.

The market for black-boxes and smartphone applications able to record and process driving-related data is growing and could generate new revenue streams [13], especially in fields like fleet management and insurance telematics.

The process of detecting a driver's behavior is usually structured in three phases. First, sensors raw data are preprocessed and cleaned to extract a denoised signal. Then, the signal is used to detect and classify driving events; the classification can be either binary (aggressive or non-aggressive event), or multi-class, i.e., to identify whether the event was related to a (an aggressive) braking maneuver, lane change, acceleration, etc. Finally, detected/classified events are employed in the computation of a kind of "score" for the driver, or used to provide real-time or deferred feedback to it.

A considerable number of recent research works focused on the classification of driving events, since correctly detecting the type of events generated during a drive is the first step

to improve drivers' safety, e.g., by issuing warnings or taking appropriate countermeasures. This research field is relatively new and no winning solutions have been found yet. One of the causes is indeed the shortage of annotated datasets created in real driving conditions. Another aspect that makes it difficult to find "the" optimal solution is that the performance of classification algorithms could be affected by many factors, like the vehicle being considered, the device used for data recording, the position of this device aboard the vehicle, the characteristics of the driver, the driving environment, etc. [14]

With the aim to address these issues, in this paper we provide a comparison of several (14) state-of-the-art algorithms for the binary classification of driving events, by testing them on multiple datasets. The objective is manifold. Firstly, we aim to determine to what extent differences in performance could be attributed to the factors mentioned above. Secondly, we intend to verify if smartphones, when used for data recording, could lead to the same results in terms of algorithms ability to discriminate between aggressive and non-aggressive events. Thirdly, we provide the new AD^2 (Aggressive Driving Detection) dataset for driving events analysis, which has some notable features as listed below.

- It contains data collected using multiple devices, namely an Android smartphone and an AutoPi device (a configurable black-box connected to the OBD-II port, equipped with motion sensors); for the same event two recordings collected in the same conditions are available, to be possibly used for studying the device impact on performance.
- It provides, besides data collected by motion sensors, geographical information on where the event occurred; location data could be used to devise algorithms that consider also road characteristics, the presence of traffic lights, of roundabouts, etc.
- It describes events generated traveling a path designed to include a variety of situations typical of everyday driving like accelerating/braking close to traffic lights, traversing roundabouts with a different radius/size, etc.
- It includes sensors readings related to events triggered by driving the path twice, with both a non-aggressive and an aggressive driving style; when comparing aggressive to non-aggressive events, this information could help to separate the effects due to the driving style from the effects due to the road characteristics.

The rest of the paper is organized as follows: Section II presents relevant works in the field of driving behavior and driving event detection, recalls already existing public datasets, and underlines the motivations behind this work. Section III provides details on the compared algorithms, whereas Section IV illustrates the adopted methodology. Section V discusses the datasets considered in the comparison, including the newly created one. Section VI presents the results of the comparison, whereas Section VII discusses the main outcomes. Finally, Section VIII provides the conclusions and suggests directions for future work. As a side note, in the following we will refer to aggressive events using the words "aggressive", but also "harsh", "unsafe" and "sudden", whereas the term "non-aggressive" will be used to refer to safe driving events.

II. Related Works

As defined in [15], "a driving behavior refers to the high-level global behavior, such as aggressive or conservative driving. Each global behavior consists of one or more underlying specific styles. For example, an aggressive driver (global behavior) may frequently overspeed or overtake (specific styles)". Driving events or maneuvers can be used to classify a driving style [16]. In the literature, various approaches have been proposed for detecting driving events and classifying driving behaviors. A detailed overview of recent advances in this field can be found in [9], [14] and [17].

Works proposed so far had several objectives, such as:

- detecting driver's drowsiness or fatigue by exploiting physiological data (e.g., electroencephalogram readings) [18], [19], visual data (e.g., images of the driver's face) [20], or acceleration, speed, and brake pedal usage data, among others [21], [22];
- supporting the driver in the identification of measures to reduce fuel consumption [23];
- performing driver identification, e.g., to spot unauthorized vehicle usages [24];
- detecting unsafe and potentially unsafe driving behaviors to warn the driver [25] or calculate some metrics (e.g., for insurance purposes) [13], [26].

In the following, we will focus on this latter objective, i.e., the detection of unsafe driving behaviors and, in particular, of aggressive driving events (which are the target of this paper), by presenting relevant studies in this field, existing datasets, and motivations for this work.

A. Detection of Aggressive Driving Events

In the literature, three main approaches have been explored to classify driving events: anomaly detection-, threshold-, and machine learning (ML) classifier-based methods.

Anomaly detection-based methods consider aggressive events as events that deviate from a driver's normal behavior. Among anomaly detection-based methods (especially on time series), a technique which has been frequently used is Dynamic Time Warping (DTW). DTW is a pattern recognition approach that can identify similarities between two series even when the elements in the considered patterns are not exactly aligned with each other. This approach has been used, e.g., in [27] to classify aggressive/non-aggressive turn, acceleration, braking, and swerving events. Similarly, in [28], DTW has been exploited to identify the type of driving events, which have then been classified as safe or unsafe using Bayesian inference. Even though DTW proved to be able to be effective for comparing driving series, its dependency on predefined event templates and threshold values makes it not easily transferable to different datasets [29]. Other works proposed to rely on different anomaly detection techniques. For instance, in [30], the authors stressed the fact that aggressive driving events are characterized by abnormal acceleration data and, hence, can be considered as outliers; thus, they exploited Gaussian Mixture Model (GMM), Partial Least Square Regression (PLSR), Discrete Wavelet Transform (DWT) and Support Vector Regression (SVR) to detect them.

Regarding *threshold-based* methods, fixed and variable thresholds were exploited to detect abnormal events in several works. In particular, in [31], a fixed threshold was applied to longitudinal and lateral acceleration data to distinguish between safe and unsafe acceleration, deceleration and lane change events. Similarly, in [25], three threshold levels were proposed to detect distracted and unsafe behaviors, including unsafe lane drifting, lane weaving, acceleration, braking and turns. The work in [32] focused on the detection of drunk driving by applying a threshold to longitudinal/lateral acceleration in order to identify sudden changes in direction. The authors of [33] and [34] proposed to adopt a variable threshold that adapts to the traveling speed; they showed that relying on a fixed threshold may not be the optimal solution, as different driving speeds or road types could influence detection performance. In fact, as pointed out also in [35], finding a threshold that is able to provide good results under most conditions is quite challenging, since its value is affected by the location and characteristics of the sensors, the road conditions, the traffic flow, etc. Instead of applying thresholds on acceleration data, other works explored thresholding on the *jerk* [36], [37]. The jerk is the second derivative of vehicle speed (or the first derivative of vehicle acceleration), and communicates how quickly the acceleration varies over time. Hence, it could be a more suitable indicator to evaluate how abrupt an acceleration/braking event is [36].

More recent studies focusing on the analysis of driving behaviors leveraged *ML-based classification* method. For instance, the work described in [38] used a ML-based system based on several features extracted from accelerometer data. After the identification of the best features, the authors applied a Random Forest (RF) classifier, which was able to achieve an accuracy of 95.5% in the task of distinguishing safe from unsafe driving events. The authors of [39] addressed driving on curvy roads, and proposed to rely on a Semisupervised Support Machine to reduce the amount of required labeled data (hence the labeling effort) for training a binary classifier. Their system outperformed by about 10% Support Vector Machines (SVM) when small portions of labeled data were available. Rather than relying on a single ML method, in [40] it was proposed to rely on an ensemble learning approach including Decision Tree (DT), SVM, Multi-Layer Perceptron (MLP), and K-Nearest Neighbors (K-NN) techniques to classify safe and unsafe maneuvers, reaching an accuracy and a F-score of about 94% and 93%, respectively.

ML-based techniques were also exploited to tackle related tasks such as recognizing different driving maneuvers [41], [42], [43], detecting drivers' drowsiness [21], or classifying the overall driver style as calm, normal or aggressive [44]. For instance, the work presented in [42] exploited ML techniques to recognize driving maneuvers such as lane changes, left/right turns, and U-turns from accelerometer, gyroscope and magnetometer data. Different ML approaches were compared such as MLP, SVM, RF, DT, Naïve Bayes (NB) and Bayesian Networks (BNs), with RF and MLP generally achieving superior performances [41], [45]. The authors of [46] performed a comparison of DT, RF, Artificial Neural Network (ANN), SVM, K-NN, NB, and K-Star (K*) for the multi-class classification of harsh events. Their findings underlined the superiority of the K* algorithm.

The authors of [21] addressed a different task, i.e., the detection of drivers' drowsiness. In their work, they compared K-NN, SVM, and ANN to detect drowsy driving on different road segments, highlighting the superiority of SVM for this task. In [30], algorithms belonging to the field of statistical regression, time series analysis, and ML were compared; in particular, the authors found that GMM and SVR achieved better performance than DWT and PLSR.

The vast majority of the available works rely on classical ML techniques after extracting a meaningful set of features from accelerometer, gyroscope, and magnetometer data. The most common features for driving event classification are derived from the instant acceleration along the three axes [38], [41]. The approach proposed in [41] derives a feature vector for each time window by grouping time series samples in short frames (e.g., one-second) and summarizing them according to different functions. This approach is fast to compute and effective in retaining the time series nature of the raw sensor data. Other authors, however, proposed more aggressive, and potentially more computationally demanding, feature engineering. In [38], a variety of features including histogram features, correlation coefficients, threshold violations, jerk profiles, and spectral information were extracted from the accelerometer data and compared; according to the authors, 95.5% classification accuracy could be reached by including the six best features. In [11], a second-order representation of accelerometer data based on the Bag of Words approach was presented, modeling time series of accelerometer readings as they were text documents, grouping together contiguous readings as words, and counting how many times a word appears in a signal.

A completely different approach is offered by the use of deep neural networks, such as Convolutional Neural Networks (CNNs). Very few works have investigated the potential of deep neural networks on this specific task. In a preliminary study [43], a CNN trained on IMU and GPS data outperformed conventional ML algorithms (e.g., RF, SVM, K-NN, Hidden Markov Model – HMM); the authors, however, highlighted the need for larger training sets for this type of models.

Finally, two works focused on comparing devices used to collect data rather than algorithms used to process them [47], [48]. Findings showed that data collection with smartphones may be considered as reliable and accurate as data collection with OBD-II devices [47], [48]. In some situations, however, the smartphone could overestimate critical driving events, especially when freely positioned in the vehicle [48].

B. Publicly Available Datasets

In the literature, several datasets have been proposed to support the investigation of driving events and behaviors. In the following, an overview of the publicly available datasets at the time of writing this paper is provided. The overview will focus on datasets that include data gathered in real-driving sessions which, as said, can be regarded as the most appropriate conditions to perform driving event identification.

Some recent works also explored the use of simulators to generate driving-related data, e.g., modeling aggressive driving behaviors [49]. Indeed, simulators can be used to create large and heterogeneous datasets [50], [51]; unfortunately, simulations cannot recreate (yet) all nuances of real-world data [49]. Given also the other limitations mentioned above, simulated datasets were left out of the analysis.

One of the earliest datasets that considered real driving is the *100-Car Naturalistic Driving Study* database [52]. This dataset includes more than 40,000 hours of data gathered by 100 drivers over 12 months on around 2,000,000 vehicle miles. The objective of the authors was to understand pre-crash causal and contributing factors; hence, the dataset contains only events related to near crashes, crashes and critical incidents.

The *Public UAH-DriveSet* is another real driving dataset that includes around 500 minutes of driving data collected from six different drivers and vehicles [53]. Drivers simulated three different behaviors (normal, drowsy and aggressive) on two types of roads (motorways and secondary roads). The dataset contains raw GPS and accelerometer data, processed data (recognized maneuvers and driving style estimates), as well as video recordings. Although this dataset provides raw motion data for the events, its limitation is that events were detected by an application developed by the authors, rather than independently recorded by an observer while driving; thus, a ground truth is missing.

The authors of [45] proposed another dataset collected using a smartphone app. The dataset contains data gathered by 20 volunteers driving three different cars driving for 8-minutes in an urban context. The objective of the authors was to provide an overall score for the driver rather than identifying the events triggered during the driving session. Hence, the dataset contains, together with IMU data (acquired at 2Hz), the results of the Driving Anger Scale [54]. A limitation of this dataset is the low sampling frequency.

Similarly to what done in [45], in 2015 the *AXA insurance company* shared on Kaggle a dataset with over 50,000 anonymized trips [55]; the dataset was meant to support a competition whose goal was to find the best ways to extract a "telematic fingerprint" capable of distinguishing when a trip was driven by a given driver. Although this dataset was one of the largest ones to be shared publicly, it is no longer available as it has been removed at the end of the competition.

Two datasets generated using a smartphone app were shared in [41] and [11] (later referred to as *Ferreira's* and *Carlos's*, respectively). Ferreira's dataset contains 50 minutes of driving data collected by two drivers on a single vehicle, including raw data for roughly 70 events (harsh braking, harsh acceleration, harsh cornering, harsh line change, and normal events), both aggressive and non-aggressive, as well as raw data associated with normal driving between the events. Carlos's dataset contains raw data gathered on two vehicles related to approximately 750 events (harsh braking, harsh acceleration, harsh cornering, harsh line change and normal events).

Lastly, a recent paper proposed a dataset created using a Raspberry Pi minicomputer [46]. Data were collected on three different vehicles by three drivers performing aggressive maneuvers such as sudden turns, accelerations, and decelerations. Since the goal was to classify aggressive behaviors, the dataset only contains raw data related to harsh events.

C. Motivations Behind this Work

The above review clearly shows that the approaches adopted so far for the binary and multi-class classification of driving events are rather heterogeneous. This situation is summarized in Table I, which also indicates that, when comparisons have been performed, a few algorithms were considered, mainly working on a single dataset collected using only one device.

The objective of our work is to address these limitations by comparing a representative set of state-of-the-art algorithms, testing them on multiple datasets, collected with different devices, mounted on various vehicles driven in diverse locations, with the aim to identify which algorithms achieve better performance in the majority of situations, as well as to determine whether the characteristics of the devices used for data acquisition actually impact on recognized events.

Thus, similarly to [11], [21], [30], [41], [42], [43], [44] and [46], we compare multiple approaches for event classification with a specific focus on binary classification (safe/harsh events). Our aim is to study objective ways to identify harsh and safe driving events, which are the starting point to detect whether a subject drives with an aggressive or a non-aggressive driving style. However, differently than in the mentioned works, we perform the comparison on multiple datasets, including Ferreira's [41], Carlos's [11] and our AD^2 dataset. We excluded datasets that contained only events related to near crashes, crashes and critical incidents [52] or aggressive events [46], which did not include an independent ground truth [53] or with a very low sampling rate (2Hz, compared to 50Hz of most of the other works) [45]. Additionally, in the AD^2 dataset we include, for the same driving event, data collected using an Android phone and an AutoPi OBD-II device. To the best of our knowledge, only two studies compared events detected using smartphone apps and OBD devices [47], [48], and in both cases the comparison was limited to one algorithm.

Algorithms to include in the comparison were selected from the initial set reported in Table I. In particular, for anomaly detection-based approaches, we left out only DTW (since no predefined event templates were available for the considered datasets). Regarding threshold-based approaches, we considered the application of thresholds on both acceleration and jerk. As for ML-based approaches, we selected one representative algorithm per family (selected algorithms are depicted in italics): {ANN: *MLP*, *CNN*}, {*SVM*}, {DT, *RF*}, {*BN*, NB, HMM}, {*K-NN*, *K**}. The final list includes the following 14 algorithms: GMM, PLSR, DWT, SVR, threshold-based approaches and jerk computation, MLP, CNN, SVM, RF, BN, K-NN and K*.

III. Details on Compared Algorithms

In the following, some background information on each algorithm included in the comparison is reported.

TABLE I: Algorithms reported in the literature. Superscript numbers refer to the number of datasets used in a given work, whereas letters refer to the device used to collect the data (S: smartphone, H: ad-hoc device, C: car simulator, O: other).

Alg.	Binary classif.	Multi-class classif.
DTW		[27]$^{1(S)}$, [28]$^{1(S)}$, [56]$^{1(S)}$
GMM	[10]$^{1(O)}$, [13]$^{1(S)}$, [30]$^{1(S)}$, [38]$^{1(O)}$	
PLSR	[30]$^{1(S)}$	
DWT	[30]$^{1(S)}$, [45]$^{1(S)}$	
SVR	[30]$^{1(S)}$	
Thresh.-based	[32]^{1S}, [33]$^{1(S)}$, [34]$^{1(S)}$, [57]$^{1(H)}$	[25]$^{2(S,S)}$, [31]$^{1(S)}$, [47]$^{1(S,O)}$, [48]$^{1(S,O)}$, [58]$^{1(S)}$, [59]$^{1(H)}$, [60]$^{1(S)}$, [61]$^{1(S)}$, [62]$^{1(S)}$
Jerk	[13]$^{1(S)}$, [36]$^{1(O)}$, [37]$^{1(O)}$, [38]$^{1(O)}$, [63]$^{1(H)}$	
MLP	[16]$^{1(H)}$, [21]$^{1(C)}$, [11]$^{3(O,S,S)}$, [40]$^{1(S)}$, [44]$^{1(S,O)}$, [64]$^{1(S)}$	[11]$^{3(O,S,S)}$, [41]$^{1(S)}$, [16]$^{1(H)}$, [42]$^{1(S)}$, [47]$^{1(S,O)}$,[46]$^{1(H)}$
CNN		[43]$^{1(O)}$
SVM	[21]$^{1(C)}$, [39]$^{1(C)}$, [40]$^{1(S)}$, [44]$^{1(S,O)}$	[35]$^{1(S)}$, [41]$^{1(S)}$, [43]$^{1(O)}$, [46]$^{1(H)}$, [64]$^{1(S)}$,
RF	[11]$^{3(O,S,S)}$, [38]$^{1(O)}$, [44]$^{1(S,O)}$	[11]$^{3(O,S,S)}$, [41]$^{1(S)}$, [43]$^{1(O)}$, [46]$^{1(H)}$, [64]$^{1(S)}$
DT	[40]$^{1(S)}$	[28]$^{1(S)}$, [42]$^{1(S)}$, [47]$^{1(S,O)}$, [64]$^{1(S)}$
BN		[28]$^{1(S)}$, [41]$^{1(S)}$
NB	[11]$^{3(O,S,S)}$, [40]$^{1(S)}$	[11]$^{3(O,S,S)}$, [42]$^{1(S)}$, [46]$^{1(H)}$, [64]$^{1(S)}$
HMM		[28]$^{1(S)}$, [43]$^{1(O)}$
K-NN	[11]$^{3(O,S,S)}$, [21]$^{1(C)}$, [40]$^{1(S)}$, [44]$^{1(S,O)}$	[11]$^{3(O,S,S)}$, [27]$^{1(S)}$, [43]$^{1(O)}$, [46]$^{1(H)}$, [56]$^{1(S)}$, [64]$^{1(S)}$
K*		[46]$^{1(H)}$

A. Anomaly Detection-based Approaches

1) Gaussian Mixture Model: GMM is a probabilistic model that exploits soft clustering techniques to assign items to clusters. Some works proposed to exploit GMM to model drivers [65] and detect aggressive driving behaviors [10], [38]. For instance, in [38], it was used to cluster driving events and styles with the aim to spot aggressive ones. In [10], it was exploited to model driving aggressiveness as a transformation operating on driving signals, especially on speed as well as on lateral and longitudinal accelerations.

As suggested in [30], we used GMM to cluster driving events data, and detect aggressive events by comparing predicted acceleration values with the true ones. After clustering, the distance (i.e., the fitting residual) of each data point with respect to the classification center is computed. A high fitting residual is likely to indicate an anomaly and thus, potentially, a sudden change in the driver's behavior.

2) Partial Least Squares Regression: PLSR is a statistical method that could be seen as a Principal Component Analysis (PCA) with regression. It performs two operations: first, it reduces the predictors to a smaller set of uncorrelated components; then, it carries out least squares regression on them rather than on the original data. PLSR has been used for incident detection [66], modeling a traffic incident as an anomaly characterized by an abrupt change in traffic parameters.

In our work, PLSR has been applied following the process described in [30], in which it has been used to detect aggressive driving. Thus, the algorithm has been first exploited to predict, based on accelerometer data acquired at t_i, sensors readings at t_j (with $j > i$); then, the distance between the predicted values and the actual ones has been computed. Considerable differences between the predicted and actual values have been considered as indicators of aggressive driving.

3) Discrete Wavelet Transform: DWT is a technique that is commonly used for signal/image compression and denoising. Wavelets are particularly interesting as they have the capability to detect anomalies of short duration [67]. Hence, some authors proposed to exploit them to assess driving behaviors [30], [45], or to investigate the correlation between road anomalies and driving behavior [67]. Among the wavelet families, one of the most common is the Daubechies wavelet (dbN), which has been frequently used in the processing of accelerometer data [45], [67]. In our work, we adopt dbN (in particular, db4, as suggested in [30]) to detect aggressive driving. The underlying assumption is that an abrupt change in acceleration can be modelled as some noise applied to normal acceleration data. In practice, the db4 wavelet has been used to decompose accelerometer data. The decomposed signal $S(t)$ is then split in two complementary signals, $a(t)$ (the approximation) and $d(t)$ (the detail). Then, the original signal is reconstructed by the inverse wavelet transform after removing $a(t)$ (i.e., setting $a(t) = 0$) and keeping only the $d(t)$ component. Finally, the distance between the reconstructed and the original signal is computed. Events characterized by a distance larger than a predetermined threshold are marked as aggressive.

4) Support Vector Regression: SVR is the application of SVM to regression. With respect to linear regression models, which aim to minimize the sum of squared errors, SVR introduces an additional hyperparameter which considers the amount of error that is acceptable in the model. SVR has been used to detect anomalies during driving. In particular, in [68], it was used to forecast and estimate driver's fatigue based on electroencephalography data. Another work exploited SVR to identify asymmetries in car-following, as well as its impact on traffic flow evolution [69]. In our work, SVR is used, like in [30], to predict sensor readings at t_j based on readings at t_i. Events showing a distance between the prediction and the actual value higher than a threshold are marked as aggressive.

B. Threshold-based Approaches

1) Thresholds on acceleration data: One of the simplest approaches proposed so far to detect aggressive events is the use of thresholds on acceleration data. When a sensor reading exceeds a given threshold (just once, or several consecutive times), a harsh event is identified. Since methods based on thresholds are frequently adopted in black-boxes (due to their low computational footprint), we decided to include them in the comparison. There are many works in the literature that rely on this approach, and proposed many different values for acceleration thresholds. An overview of the most common values is provided in Table II. In this work, we tested this approach in two ways. The first way consists in computing the Residual Sum of Squares (RSS), as suggested in [30], among the sensor readings and zero at each instant; once the RSS has been computed, the threshold that maximizes the F-score

TABLE II: Overview of thresholds presented in literature.

Threshold	Aggr. acceleration	Aggr. braking	Aggr. turn
0.1	[48]	[48]	[58]
0.15	[59]	[59], [60]	
0.25	[60]	[32], [61]	
0.28	[32], [61]		
0.3	[58], [62]	[62]	[58], [62]
0.35		[57]	
0.4		[25]	[32], [61]
0.45		[57]	
0.5	[31]	[31], [58]	

is determined (in the following, we will refer to this approach as "RSS threshold"). The second way consists in "simply" identifying aggressive events when sensors readings exceed a given threshold (in the following, we will refer to this approach as "simple threshold"); in this case, we experimented with all the threshold values reported in Table II in order to determine the one(s) providing the best results.

2) Jerk evaluation: As said, the jerk is the variation rate of acceleration. When it comes to identifying aggressive events, the jerk is an interesting measure to rely on, since it indicates how quickly the acceleration changes over time. Various works proposed to exploit it to detect aggressive driving [13], [38], [63], [36], [37]. For this reason, we decided to include the jerk in our comparison; in particular, events for which the jerk was larger than a given threshold were considered as aggressive.

C. ML-based Classification Approaches

Seven well-known classifiers (MLP, CNN, SVM, RF, BN, K-NN and K*) were considered in the analysis, including five parametric classifiers (MLP, CNN, SVM, RF and BN) and two non-parametric classifiers (K-NN and K*). All parametric classifiers learn a decision function; its characteristics are dependent on the specific classifier, which is trained on the training set and then applied to the new unseen cases at test time. Within the family of ANNs, we included two shallow models: MLPs ([14], [41], [16], [24]) and CNNs [43]. Within the family of non-parametric classifiers, besides k-NN we included also K*, due to its performance in recognizing aggressive/risky driving behaviors [46]. Both algorithms identify the class of a target point by comparing it against labeled samples in the training set, and selecting the k-nearest neighbors in feature space. The main difference is that K* exploits an entropy-based metric, rather than the classical Euclidean metric, to compute the distance among data points.

IV. Methodology

Fig. 1 reports an overview of the whole analysis process. Four phases can be distinguished: *Data acquisition*, *Data transformation*, *Algorithms application* and *Evaluation*. In the following, details on each phase are reported.

A. Data Acquisition

The upper part of Fig. 1 depicts data acquisition. As said, in this work, a new dataset has been created. Data were collected using an Android smartphone (on the left) and an AutoPi

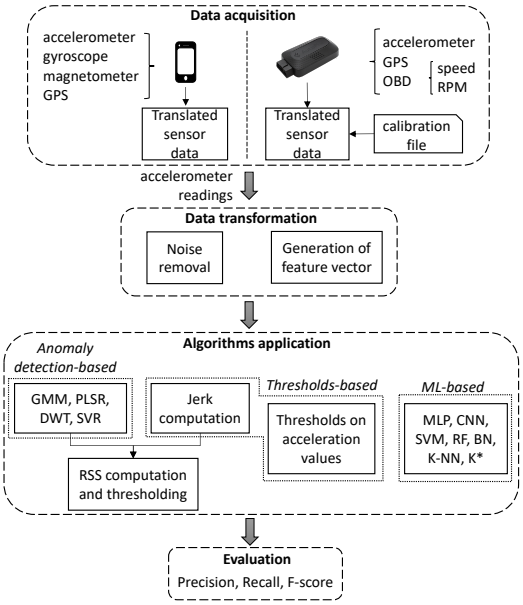

Fig. 1: Overview of the whole analysis process.

OBD-II device (on the right). On the smartphone, accelerometer, gyroscope, magnetometer and GPS sensors have been used to collect motion data, which were translated to Earth coordinates directly onboard. The AutoPi device recorded accelerometer readings, latitude/longitude coordinates, as well as speed and RPM data gathered through the OBD-II port. Data acquired with this latter device were subsequently translated into Earth coordinates using a 10-second calibration file (recorded when the vehicle was steady) by following the approach in [70]. This approach corrects the axes orientation by applying a quaternion rotation transformation to raw data. It is worth remarking that, even though more data were collected, for the sake of comparison, only accelerometer readings were used in the analysis, as they are the common factor of the other considered datasets.

B. Data Transformation

Acceleration data were transformed to remove noise using a second-order Savitzky-Golay filter, with a time window of one second, as suggested in [63]. For ML-based approaches (MLP, CNN, SVM, RF, BN, KNN and K*) a feature vector was built based on the process in [41].

In particular, a sliding window of ns seconds (encompassing seconds from s_0 to s_w, with $w = ns - 1$) was defined. Sensor data were then aggregated by computing the following summarizing functions SFs over one-second long frames: mean M, median MD, standard deviation SD and tendency T, as reported in Eq. 1.

$$\begin{aligned}
&M_0 = M(s_0, s_w);\ M_1 = M(s_1, s_w); && ...M_w = M(s_w)\\
&MD_0 = MD(s_0, s_w); MD_1 = MD(s_1, s_w); && ...MD_w = MD(s_w)\\
&SD_0 = SD(s_0, s_w);\ SD_1 = SD(s_1, s_w); && ...SD_w = SD(s_w)\\
&T_0 = \frac{M(s_0)}{M(s_w)};\ T_1 = \frac{M(s_1)}{M(s_w)}; && ...T_{w-1} = \frac{M(s_{w-1})}{M(s_w)}
\end{aligned} \tag{1}$$

Thus, each sliding window of ns seconds contained ns features for each SF, with $SF(s_i)$ summarizing sensors data for second s_i, and $SF(s_i, s_j)$ summarizing data for seconds from i to j. Summarizing data in Eq. 1 were separately computed for all the three axes x, y, z and then concatenated to form the final feature vector. As done in [41], feature vectors were built for windows with different number of seconds (e.g., $ns = 4, 5, 6, 7, 8$); the aim was to verify which window size was most suitable, as driving events had different lengths.

C. Algorithms Training and Assessment

The algorithms described in Section III were run on the extracted features using the configurations given in Table III.

For anomaly detection-based approaches, the fitting residual between the original signal and the predicted one was evaluated to identify whether the event should be classified as aggressive or non-aggressive. In order to further reduce the noise (which was already removed using the above-mentioned process), the following steps were followed. First, sliding windows of ns seconds were extracted, with $ns = \{1, 4, 6, 8, 12, 16\}$.

The windows were built as in Eq. 2,

$$\begin{aligned} w_0 &= (s_0, s_1, ..., s_{ns*sr-1}); \\ w_1 &= (s_1, s_2, ..., s_{ns*sr}); \\ w_2 &= (s_2, s_3, ..., s_{ns*sr+1}); \\ &... \end{aligned} \tag{2}$$

with sr being the sampling rate.

For each time window, the number of elements showing a fitting residual higher than a threshold τ was computed and, if it was larger than a predefined amount n, the whole window was classified as aggressive, otherwise as non-aggressive.

Based on the above methodology, the GMM was applied as follows: it received as input all the sensors readings/features (i.e., denoised x, y and z values) and grouped them in k clusters (with k varying as reported in Table III). Then, for each data point, the RSS w.r.t. the cluster center was computed as $RSS = (\gamma - \hat{\gamma})^2$, where γ is the actual value, and $\hat{\gamma}$ is the predicted one [30]. The result was then compared with the threshold τ. Similarly to GMM, the PLSR and SVR algorithms took as input all the features, and were parametrized based on previous work [30]. In this case, the number of components n_c was set to two [30], and the RSS was computed on the difference between the values of sensors readings in the next instant and the value predicted by the algorithm. For the DWT approach, the db4 wavelet was instead applied to each axis independently, as done in [30].

With regard to threshold-based approaches and, in particular, the "RSS threshold" approach, the RSS of the distance between each sensor reading and zero was calculated. When at least n elements in a window of ns seconds presented a RSS value higher than a given threshold τ, the window was classified as aggressive. This mechanism was applied to both acceleration values and jerk. The values of ns and τ were dynamically changed to find the ones maximizing the F-score. For the , each data point related to x and y sensors readings was labeled as aggressive when, for y readings (frontal accelerations), the acceleration was higher than a threshold τ_{accel} or lower than a threshold τ_{brak} and when, for x readings (lateral accelerations), the absolute value of the acceleration was higher than a threshold τ_{turn}. We did not consider z, as we did not find in the literature suitable thresholds. The set of values experimented for the thresholds is that reported Table II.

TABLE III: Configurations for the considered algorithms.

GMM	$k \in (1, 10)$
PLSR	Number of components n_c = 2
SVR	Radial basis kernel function, penalty $C = 2$
Thresh.	Values reported in Table II
MLP	one hidden layer with size $H \in \{(attribs + classes)/2, 40, 30, 20, 10\}$
CNN	Hyperparameters set as in [43]
SVM	Polynomial and radial basis kernel functions; cost values $C \in (2^5, 2^{13.5})$; gamma values $G \in (2^{-7.5}, 2^{-1})$
RF	Number of iterations $I \in \{100, 200\}$, number of features to randomly investigate $K \in \{log_2(\#predictors) + 1, 10, 15\}$
BN	Search algorithms: K2, TAN, Repeated Hill Climber
K-NN	Number of nearest neighbors $K = \{1, 3, 5, 7, 9\}$
K*	Blend setting $B = \{10, 20, 30, 40\}$

Lastly, for ML-based approaches the hyper-parameter configuration was chosen experimentally, as well as taking into account the results of [41] for MLP, SVM, RF and BN, and [43] for the CNN. The most relevant hyper-parameters for each algorithm were optimized experimentally using grid search. Hyper-parameter configurations are reported in Table III.

D. Evaluation

Each algorithm configuration was evaluated on the binary classification task by computing precision, recall and F-score metrics. Metrics were calculated separately for the "aggressive" and "non-aggressive" classes, and then the weighted average was taken in order to account for the different number of events per class.

Metrics were calculated on the entire dataset for threshold-based algorithms, and using k-fold cross validation (with $k = 10$) for those algorithms requiring a training phase. Since events may have a different duration, each time window belonging to the event was labeled as "aggressive" or "non-aggressive". If at least 50% of the time points belonging to the window were related (in the ground truth) to an aggressive event, the whole window was labeled as aggressive, otherwise as non-aggressive. To minimize correlation between different training samples, non-overlapping time windows were extracted from each dataset. In the case of consecutive short events, it is possible that a given time window contains or overlaps with more than one event. For use cases that require a precise separation between close events, an alternative strategy could be to exploit the time window for classifying only the central time point. At inference time, when the window slides by one frame at the time (stride set to 1), it is possible to obtain a fine-grained frame-by-frame classification at the expenses of higher computation times. For the purpose of evaluating the ability of different ML algorithms to separate "aggressive" vs. "non-aggressive" events, both approaches are viable.

The selected 14 algorithms were evaluated according to two different experimental setups. The first setup focuses on the

classification of *labeled events* as in previous work [41]. The second setup includes *all available acceleration data*, including those recorded during normal driving situations; since the performance is evaluated on a continuous data stream, this type of assessment more closely reflects a realistic scenario and frames the task as truly a *detection*, rather than a classification, task. This second set of experiments was performed only on the Ferreira's and AD^2 datasets. To calculate the performance metrics, the unlabeled portion of the data was split into non-overlapping segments of duration equal to eight seconds, all labeled as non-aggressive. Furthermore, we decided to limit the range of ns values used for the Ferreira's and Carlos's datasets (setting the maximum value to 8 and 4, respectively) in order to reduce the overlap among subsequent events, as the events included in these datasets are generally shorter than such time intervals.

V. Datasets

To test the different approaches, three datasets have been exploited, whose characteristics are presented below.

A. AD^2 Dataset

The AD^2 dataset was collected during a driving session performed in the city of Turin (Italy). The session lasted more than two hours. The vehicle used was a 2007 Ford Fiesta, the roads were paved with asphalt and dry, the weather was sunny, and the level of traffic was medium-high. The car was driven by a driver with 20 years of experience.

As said, data have been acquired using an Android app (a modified version of the app presented in [41]) and an AutoPi device. The app was installed on a Google Pixel 2 smartphone (equipped with Android 11), which was placed on the anterior car windshield using a suction-cup smartphone mount. The smartphone orientation was vertical. The AutoPi device was connected to the OBD-II port of the car and was installed using velcro straps near the steering wheel, with an inclination with respect to the ground of around 40 degrees.

The path followed during the driving session was composed of several sub-paths, which were planned in advance in order to represent as much as possible an everyday urban driving scenario encompassing, among others, acceleration and braking events close to traffic lights, left/right turns with different radius, and u-turns. Furthermore, we devoted particular attention at including in the sub-paths also roundabouts (of different radius and with different number of lanes), an element which is becoming more and more frequent in many cities.

Each sub-path was traversed twice, a first time with a non-aggressive driving style, a second time with an aggressive driving style. This choice was made to collect data on driving events with the two styles under the same or at least very similar (road, weather, and traffic) conditions. The path traveled to reach the sub-path locations was generally traversed with a non-aggressive driving style even though, occasionally, some aggressive events were triggered (mainly harsh acceleration and braking events, due to traffic conditions). The events generated during the non-aggressive or the aggressive driving were labeled accordingly (safe or harsh). For each event, an independent observer on board of the vehicle recorded its start and end timestamps to generate the ground truth, together with a label identifying the event type. It is worth remarking that, similarly to other datasets such as the Ferreira's and the Carlos' ones, the labels assigned to events contained in the AD^2 dataset are strongly influenced by the subjectivity of the driver/evaluator. In particular, when asked to travel the path with an aggressive driving style, the driver adopted a driving style that subjectively seemed aggressive (to the driver), still respecting everyone's safety. Nonetheless, data collected (and labels assigned) by other more/less experienced drivers could be different than those collected in the AD^2 dataset. A discussion on how to increase the objectivity of collected data is provided in Section VII.

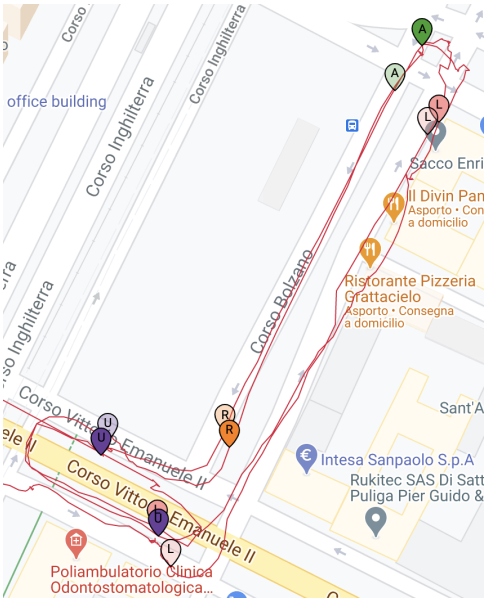

Fig. 2: Example of recorded driving events (green "A" marker : acceleration, orange "R" marker : right turn, pink "L" marker : left turn, "U" purple marker : u-turn, transparent marker : non-aggressive event, opaque marker : aggressive event, red line : path traveled by the vehicle).

Fig. 2 depicts a subset of the recorded events. Latitude and longitude data recorded at the start time of each event have been used to draw markers on the map. The map shows non-aggressive/aggressive acceleration and right/left/u-turn events.

Table IV reports the number and type (label) of recorded events. The set of event types considered in this dataset was based on that used in [26] and [41], which was enlarged to include u-turns and events in roundabouts. In total, 135 (48 aggressive, 87 non-aggressive) events were recorded using the Android app, 126 (46 aggressive, 80 non-aggressive) by the AutoPi device. This difference was due to the late startup of the latter device after engine start. Additionally, the dataset includes 124 minutes of normal driving, which are not labeled.

As said, the Android application collected accelerometer, gyroscope, magnetometer and GPS data. For the AutoPi device, accelerometer and GPS data provided by the device sensors were recorded (the device was not equipped with gyroscope and magnetometer); this device also enabled the

TABLE IV: Events recorded in the AD^2 dataset.

	Smartphone		AutoPi	
Event type	**Aggr.**	**Non-aggr.**	**Aggr.**	**Non-aggr.**
Acceleration	7	12	7	11
Braking	10	20	10	20
Lane change		4		4
Left turn	9	11	8	10
Left turn in roundabout	1	4	1	3
Right turn	8	15	7	13
Right turn in roundabout	2	5	2	4
Straight roundabout	4	8	4	8
U-turn	5	6	5	5
U-turn in roundabout	2	2	2	2
Total	**48**	**87**	**46**	**80**

TABLE V: Events recorded in Ferreira's and Carlos' datasets.

Event type	**Ferreira's**	**Carlos's**
Aggressive braking	12	150
Aggressive acceleration	12	146
Aggressive turns	11 left, 11 right	149
Aggressive lane changes	4 left, 5 right	149
Non-aggressive event	14	148
Total	**69**	**742**

recording of speed and RPM data, retrieved from the OBD-II port. The sampling rate was 1Hz for GPS, speed and RPM, 50Hz for the other sensors. It shall be recalled, though, that for the comparison we only used accelerometer data. The created dataset is publicly available at http://tiny.cc/1tsxtz.

Fig. 3a and 3b show some sample plots for acceleration data (reported on the y axis) recorded using the Android app for a non-aggressive u-turn event and an aggressive u-turn event recorded in the same location. Fig. 3c and 3d show acceleration data for the same events recorded using the AutoPi device. A visual comparison clearly shows similarities between the acceleration curves of the two devices, with larger acceleration values (as expected) for aggressive events.

B. Ferreira's Dataset

This dataset contains accelerometer, gyroscope and magnetometer data gathered through an Android app during four car trips of around 13 minutes each [41]. The driving sessions were carried out on a 2011 Honda Civic by two experienced drivers on sunny days and on dry roads paved with asphalt. The phone was mounted on the car windshield by means of a car mount. The sampling rate varied between 50Hz and 100Hz depending on the sensor. Ferreira's dataset contains 69 events, 55 aggressive and 14 non-aggressive. Table V (second column) reports an overview of events recorded in this dataset. In particular, this dataset contains sensors data acquired during the entire duration of the driving sessions, i.e., both data related to relevant (i.e., labeled) safe/harsh events, as well as data related to events not considered as relevant from the driver's perspective (events typical of everyday driving situations, which could be generally regarded as non-aggressive).

C. Carlos's Dataset

The authors of this dataset [11] gathered acceleration data related to aggressive and non-aggressive events using again an Android app on two vehicles, namely, a Honda Accord and a Nissan Altima. Around 40 minutes of events were collected. The smartphone was freely positioned in the vehicle, in the driver's door lower compartment or in the cup holder, depending on the vehicle. The sampling rate was 50Hz. Table V (third column) reports an overview of the events recorded in this dataset. Differently than with the Ferreira's and our dataset, the Carlos's dataset only contains relevant events occurred during the driving session rather than data for the whole driving session; hence, it does not completely reflect an everyday driving situation in which a high number of non-aggressive events are generally present.

VI. Experimental results

The first set of experiments evaluated the classification performance on labeled aggressive vs. non-aggressive events. Table VI reports the best results achieved by each algorithm (the corresponding configurations that provided the best performance are given in Table VII). Within anomaly detection-based approaches, GMM outperformed the other methods (PLSR, db4, and SVR) on all the datasets. The db4 algorithm almost matched GMM performance, especially on Carlos's and AD^2 Smartphone dataset. The ranking among GMM, PLSR and db4 is in line with previous findings [30]. The optimal value for ns depends on the dataset, with shorter windows for Ferreira's ($ns = 1$) and Carlos's ($ns = 4$), and longer windows for AD^2 especially using db4. This difference can be explained by the fact that our dataset contains also roundabout and u-turn events, which are generally characterized by a longer duration compared to, e.g., acceleration or turn events.

With threshold-based approaches, the "simple threshold" method was not able to provide good results for binary classification for any of the thresholds presented in the literature (F-score ranging between 0.62 and 0.67). Furthermore, as reported in Table VII, if for the Ferreira's and Carlos's dataset a low threshold (0.1g for harsh acceleration, braking and turns) was selected as the best configuration, our dataset required generally higher thresholds (0.25– 0.35g). The best configurations of ns reflect the considerations made above, as a high value was required on our dataset, whereas on Ferreira's and Carlos's datasets lower values worked better.

When it comes to ML-based approaches, the algorithm that achieved, on average, the best results is SVM. Then, four algorithms provided results just slightly worse than SVM, i.e., CNN, RF, MLP and K-NN. When applied to the Carlos's dataset, these four algorithms had mostly the same performance (F-score of 0.98). CNN showed the second highest performance on Ferreira's, Carlos's, and the part of the AD^2 dataset collected with the smartphone, whereas had lower performance on data gathered using the AutoPi device. RF had better performance than the remaining algorithms, especially on the AutoPi part of our dataset, on which it achieved the second highest F-score, and on the Android part (it ranked third), whereas it performed slightly worse than MLP and K-NN on the Ferreira's dataset. On this dataset, MLP and K-NN had better performance than RF. MLP was the fourth algorithm in terms of performance, as it achieved the same results of K-NN on the Ferreira's, Carlos's and smartphone

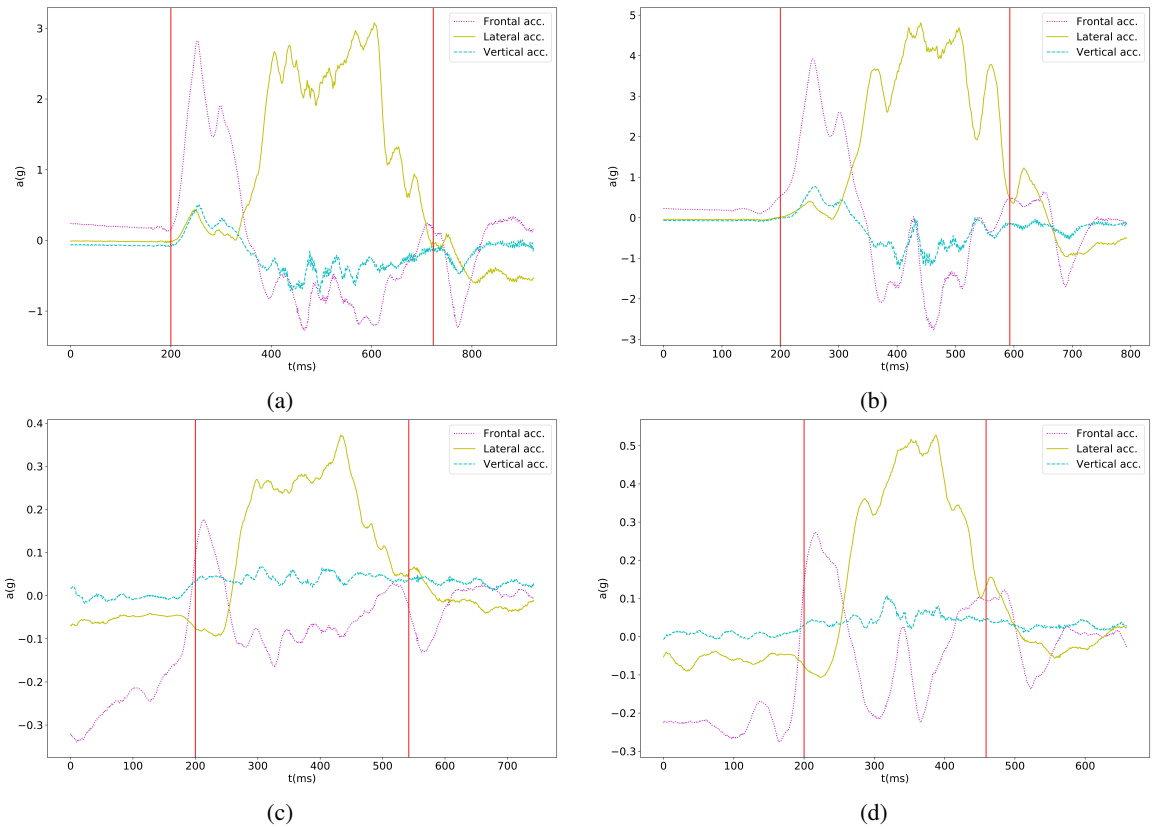

Fig. 3: Acceleration data related to a non-aggressive (a, c) and an aggressive u-turn event (b, d) acquired using the Android app (a, b) and the AutoPi device (c, d). The vertical lines represent the start and end time of the event, respectively.

TABLE VI: Metrics related to the evaluated datasets, computed on acceleration data related to labeled events.

	Ferreira's			Carlos's			AD^2 smart.			AD^2 AutoPi			Average		
Algorithm	**F-Sc.**	**Prec.**	**Rec.**	**F-Sc.**	**Prec.**	**Rec.**	**F-Sc.**	**Prec.**	**Rec.**	**F-Sc.**	**Prec.**	**Rec.**	**F-Sc.**	**Prec.**	**Rec.**
GMM	**0.81**	**0.81**	**0.82**	**0.95**	**0.95**	**0.95**	**0.77**	**0.84**	0.73	**0.85**	**0.88**	**0.82**	**0.85**	**0.87**	**0.83**
PLSR	0.73	0.71	0.77	0.91	0.91	0.91	0.65	0.67	0.65	0.65	0.65	0.65	0.74	0.73	0.75
db4	0.76	0.76	0.76	0.94	0.94	0.94	0.76	0.78	**0.75**	0.77	0.80	0.75	0.81	0.82	0.80
SVR	0.74	0.75	0.73	0.81	0.82	0.84	0.54	0.56	0.53	0.55	0.54	0.57	0.66	0.67	0.67
RSS thresh.	**0.81**	**0.79**	**0.83**	**0.95**	**0.95**	**0.95**	0.74	0.73	0.75	0.79	0.79	0.79	0.82	0.82	0.83
Simple thresh.	0.66	0.68	0.75	0.62	0.74	0.70	0.67	0.68	0.66	0.66	0.68	0.65	0.65	0.70	0.69
Jerk	0.76	0.74	0.81	0.92	0.92	0.93	**0.78**	**0.79**	**0.78**	**0.87**	**0.87**	**0.87**	**0.84**	**0.83**	**0.85**
MLP	0.96	0.96	0.96	0.98	0.98	0.98	0.82	0.82	0.82	0.82	0.82	0.82	0.89	0.90	0.89
CNN	**0.97**	**0.98**	0.96	0.98	0.98	0.98	0.84	0.85	0.84	0.81	0.84	0.81	0.90	0.91	0.90
SVM	**0.97**	0.97	**0.97**	**0.99**	**0.99**	**0.99**	**0.87**	**0.87**	**0.87**	**0.88**	**0.88**	**0.88**	**0.93**	**0.93**	**0.93**
RF	0.94	0.95	0.94	0.98	0.98	0.98	0.83	0.84	0.83	0.85	0.85	0.85	0.90	0.90	0.90
BN	0.91	0.91	0.91	0.96	0.96	0.96	0.78	0.78	0.78	0.78	0.79	0.79	0.86	0.86	0.86
K-NN	0.96	0.96	0.96	0.98	0.98	0.98	0.82	0.83	0.82	0.80	0.81	0.81	0.89	0.90	0.89
K*	0.92	0.93	0.92	0.91	0.92	0.90	0.72	0.73	0.74	0.74	0.74	0.74	0.82	0.83	0.83

part of our dataset, whereas it performed slightly better than K-NN on the AutoPi part. K-NN ranked fifth, still presenting good performance. The performances of BN and K* were largely lower than those achieved by the first five algorithms (average F-score of 0.86 and 0.81, respectively).

As illustrated in Table VII, in the case of ML-based algorithms, higher ns values consistently yielded the best results. This finding is rather interesting, especially for Ferreira's dataset, on which lower ns values were optimal for anomaly detection-based approaches. This behavior highlights how summarizing data on larger windows can improve performance, as speculated in [41].

The second set of experiments, for which results are given in Table VIII, evaluated detection performance on the entire dataset. Concerning anomaly detection-based approaches, GMM again yielded, on average, the best F-score (0.81). Similarly, db4 was the second-best algorithm in terms of performance, followed by PLSR and SVR. The best configurations, reported in Table IX, are very similar to those obtained on the labeled dataset (in Table VII). On Ferreira's dataset low ns values still provided the best results.

With respect to threshold-based approaches, thresholds on jerk and on acceleration data RSS outperformed again the simple application of thresholds presented in literature. The

TABLE VII: Best configurations (labeled events).

Algor.	Ferreira's	Carlos's	AD^2 smart.	AD^2 AutoPi
GMM	ns=1, k=1	ns=4, k=1	ns=4, k=8	ns=16, k=2
PLSR	ns=1	ns=4	ns=8	ns=6
db4	ns=4	ns=4	ns=16	ns=16
SVR	ns=1	ns=4	ns=16	ns=1
RSS thresh.	ns=4	ns=4	ns=4	ns=8
Simple thresh.	τ_{accel}=0.1g, τ_{brak}=0.1g, τ_{turn}=0.1g	τ_{accel}=0.1g, τ_{brak}=0.1g, τ_{turn}=0.1g	τ_{accel}=0.1g, τ_{brak}=0.25g, τ_{turn}=0.3g	τ_{accel}=0.3g, τ_{brak}=0.35g, τ_{turn}=0.1g
Jerk	ns=1	ns=4	ns=6	ns=12
MLP	ns=8, H=20	ns=4, H=40	ns=8, H=40	ns=8, H=a
CNN	ns=7	ns=4	ns=7	ns=5
SVM	ns=8, C=7.5, G=-1.5	ns=4, C=3.5, G=-1.5	ns=8, C=7, G=-5.5	ns=8, C=3.5, G=-3
RF	ns=8, I=100, K=10	ns=4, I=200, K=10	ns=7, I=200, K=10	ns=8, I=200, K=10
BN	ns=7, K2	ns=4, K2	ns=7, TAN	ns=6, TAN
K-NN	ns=8, K=3	ns=4, K=5	ns=8, K=3	ns=8, K=3
K*	ns=8, B=10	ns=4, B=40	ns=8, B=40	ns=8, B=30

jerk provides slightly better results on AD^2 smartphone data, and RSS-based thresholding works better on the Ferreira's dataset and on the AD^2 AutoPi device data.

With regard to ML-based approaches, all tested algorithms showed similarly good performance, with a F-score ranging between 0.96 and 0.98. SVM was ranked first, followed by MLP, RF, K-NN, CNN and BN. The lowest ranking was obtained by K*. Still considering ML-based approaches, it shall be noticed that the best ns value did not generally differ from that chosen in the first evaluation phase.

ML-based techniques achieve higher performance as they are able to capture complex, non-linear relationships between different features. To gain further insights on the importance of each feature, a univariate analysis was performed on the complete AD^2 AutoPi dataset. Let us recall that, for a window of length ns seconds, $12 \times ns$ features are computed by aggregating frontal (x), lateral (y) and vertical (z) accelerations using different SFs (mean, median, standard deviation and trend). The discriminative power of each individual feature was assessed by computing the ANOVA F-score between the type of driving event and the feature itself. Based on the score, each feature is then ranked from 1.0 (most important) to 0.0 (least important), as reported in Fig. 4a. The distribution of the feature values for time windows labeled aggressive or non-aggressive (normal) is further illustrated by the boxplots in Fig. 4b; for the sake of clarity, the features obtained by each SF are averaged over the entire time window, but similar trends are observed also at the level of each individual feature. Local variations in the acceleration, especially along the lateral axis, are the most discriminative features, as they are probably associated with aggressive turns and lane changes; lateral accelerations characterize the majority of aggressive events in the AD^2 dataset, i.e., left/right turns, roundabouts and U-turns. The mean and median accelerations are, when taken in isolation, less informative, except for the vertical axis, which is probably associated with aggressive braking and acceleration events. When considering only the labelled portion of the

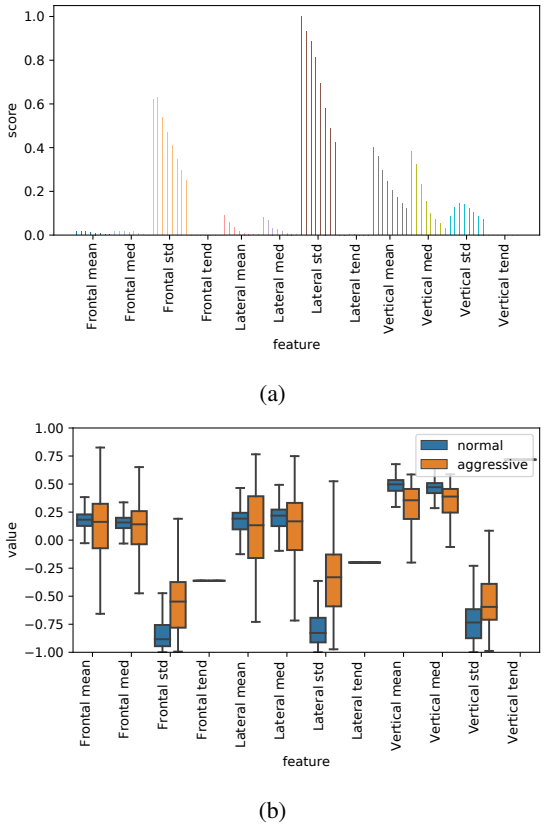

Fig. 4: Results of the univariate analysis of the features extracted from the AD^2 AutoPi dataset. Frontal, lateral and vertical acceleration are separately aggregated every second using different SFs (mean, media, standard deviation and tendency). (a) Feature importance, from highest (1.0) to lowest rank (0.0) based on ANOVA F-score. (b) Feature distribution (box-plot) for each axis and SF, aggregated over the entire time window.

dataset, the feature ranking was slightly different, with the "frontal standard deviation" and "vertical median" features ranked lower than "vertical standard deviation". This result suggests that it is important to account for all driving situations in order to achieve a robust classification.

To complement the evaluation, we also measured the computation time of each technique. In particular, the measurement was performed on both a 2018 MacBook Pro with a 2,7 GHz Intel Core i7 CPU and 16 GB RAM, and a Raspberry Pi 3 (this latter device was chosen to mimic the hardware of the AutoPi exploited for the data collection). Table X reports the average computation time per ns (in milliseconds) and the standard deviation for the selected algorithms. The average computation time was calculated as follows: first, the time required by each algorithm, in its best configuration, was determined. Then, this value was divided by ns, and the normalized values computed for each dataset were used to obtain the average time and the standard deviation. From the table, it is possible to notice that GMM, db4, "RSS threshold" and "simple threshold" have the lowest computation time per ns on both the devices, followed by PLSR, SVR and JERK. ML-

TABLE VIII: Metrics related to the evaluated datasets, computed on all acceleration data included in the dataset.

	Ferreira's			AD^2 smart.			AD^2 AutoPi			Average		
Algorithm	**F-Sc.**	**Prec.**	**Rec.**	**F-Sc.**	**Prec.**	**Rec.**	**F-Sc.**	**Prec.**	**Rec.**	**F-Sc.**	**Prec.**	**Rec.**
GMM	**0.81**	0.81	**0.81**	**0.77**	**0.84**	0.73	**0.85**	**0.88**	**0.82**	**0.81**	**0.84**	**0.79**
PLSR	0.80	**0.84**	0.78	0.64	0.65	0.62	0.65	0.63	0.68	0.70	0.71	0.69
db4	**0.81**	0.82	0.80	0.74	0.72	**0.76**	0.71	0.69	0.74	0.75	0.74	0.77
SVR	0.74	0.75	0.73	0.54	0.56	0.53	0.55	0.54	0.57	0.61	0.62	0.61
RSS thresh.	**0.81**	**0.81**	**0.81**	0.74	0.70	**0.81**	**0.81**	**0.80**	**0.82**	**0.79**	0.77	**0.82**
Simple thresh.	0.72	0.69	0.75	0.69	0.76	0.65	0.71	0.69	0.73	0.71	0.71	0.71
Jerk	0.78	0.78	0.77	**0.78**	**0.86**	0.74	0.79	0.77	0.80	0.78	**0.80**	0.77
MLP	0.97	0.97	0.97	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	0.97	**0.98**
CNN	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	**0.98**	0.97	0.97	0.97
SVM	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**
RF	0.97	**0.98**	**0.98**	0.97	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**	**0.98**
BN	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97
K-NN	**0.98**	**0.98**	**0.98**	0.97	0.97	**0.98**	0.97	0.97	**0.98**	0.97	0.97	**0.98**
K*	0.97	0.97	0.97	0.97	0.97	0.96	0.96	0.97	0.97	0.97	0.97	0.97

TABLE IX: Best configurations (all events).

Algor.	**Ferreira's**	**AD^2 smart.**	**AD^2 AutoPi**
GMM	ns=1, k=1	ns=4, k=8	ns=16, k=2
PLSR	ns=4	ns=16	ns=4
db4	ns=4	ns=12	ns=12
SVR	ns=1	ns=16	ns=1
RSS thresh.	ns=1	ns=4	ns=16
Simple thresh.	τ_{accel}=0.15g, τ_{brak}=0.15g, τ_{turn}=0.1g	τ_{accel}=0.25g, τ_{brak}=0.25g, τ_{turn}=0.3g	τ_{accel}=0.3g, τ_{brak}=0.3g, τ_{turn}=0.1g
Jerk	ns=4	ns=6	ns=6
MLP	ns=8, H=30	ns=8, H=40	ns=8, H=40
CNN	ns=8	ns=6	ns=8
SVM	ns=8,C=3.5,G=-3	ns=8,C=5.5,G=-5	ns=8,C=3,G=-3
RF	ns=8,I=200,K=15	ns=8,I=200,K=15	ns=7,I=100,K=15
BN	ns=8, K2	ns=8, K2	ns=5, K2
K-NN	ns=8, K=1	ns=8, K=3	ns=8, K=3
K*	ns=8, B=40	ns=8, B=20	ns=8, B=40

TABLE X: Computation time per ns (in milliseconds)

	MacBook Pro		Raspberry Pi	
Algorithm	**Average**	**Std. dev.**	**Average**	**Std. dev.**
GMM	0.5	0.3	2.8	2.5
PLSR	5.4	4.9	46.1	42.8
db4	0.5	0.3	3.7	2.6
SVR	1.2	0.9	49.5	59.0
JERK	5.6	5.4	47.2	37.6
RSS thresh.	0.7	1.3	6.3	11.8
Simple thresh.	1.4	0.4	3.9	2.5
MLP	7.6	0.4	93.4	2.6
CNN	8.3	0.8	99.6	6.7
SVM	7.8	0.5	95.5	4.1
RF	7.7	0.4	94.9	4.3
BN	7.6	0.5	93.4	2.2
K-NN	9.1	1.4	117.4	33.6
K*	123.2	80.8	1538.8	1081.7

based approaches, instead, show higher computation times, with all of them, except K-NN and K*, being very similar. Not surprisingly, instance-based classifiers are the slowest ones, as their complexity grows with the data. It must be said, though, that feature extraction and data preprocessing are the most computationally expensive steps for ML-based techniques, accounting for around 83% of the overall time (83.1% on the MacBook Pro and 83.2% on the Raspberry Pi). For anomaly detection-based methods, the above steps account for 39.8% (MacBook Pro) and 45.3% (Raspberry Pi) of the overall time, whereas for threshold-based methods, they account for 41.4% (MacBook Pro) and 53.5% (Raspberry Pi). Data reported in Table X show that, on the Raspberry Pi (the device with the most limited resources) real-time processing can be achieved only for a subset of the tested algorithms. In fact, since generally the optimal configuration for the selected algorithms requires a ns value higher than 1 second, the average time should be multiplied several times (generally, 8 times, but in some cases even 16 times). By taking into account this aspect, it appears that only GMM, db4, "RSS threshold" and "simple threshold" (when run using the best configuration hyperparameters) can complete the computation in less than 0.2 seconds. The remaining algorithms show an execution time between 0.2 and 1 second, with the exception of K*, which could require even more than 10 seconds.

VII. Remarks

Overall, our results confirmed the superiority of ML-based approaches with respect to the other proposed algorithms [14].

Anomaly-based algorithms did not perform better than threshold-based technique in most datasets and in both sets of experiments. Among the evaluated algorithms, GMM usually achieved the best performance, especially when trained and tested only on labeled events.

In all experiments, threshold-based techniques achieved lower results that the other techniques. Among the selected techniques, "RSS threshold" improved the results compared to a simple thresholding on Ferreira's and Carlos's datasets, whereas for our dataset better performance was reached using the jerk. This kind of approach should be extended, e.g., taking into account additional factors such as the amount of time sensors readings that are above a threshold or the number of such readings over that threshold in a given time window.

Our results confirm previous findings that the threshold value is influenced by the position and characteristics of the sensor, the type of the vehicle, as well as the road and traffic conditions, among others [35]. Higher thresholds required on the AD^2 dataset were probably due to the type of vehicle and traffic conditions. Different vehicles could generate high accelerations values for some non-aggressive events, thus

making approaches considering the smoothness or abruptness of an event perform better than a punctual thresholding.

As far as ML-based techniques are concerned, the best results were generally obtained by the SVM and RF techniques. This is inline with previous literature, and can be attributed to the ability of these techniques to model non-linear relationships and to the fact that they are generally robust w.r.t. high number of input features and data imbalance. A potential advantage of RF over SVMs is their higher interpretability, as they permit to compute the importance of individual features [38]. Non-parametric approaches like k-NN were ranked lower than parametric approaches and, in addition, are computationally expensive at inference time, as highlighted in Table X. Both sets of experiments highlighted how summarizing data on larger windows can improve algorithms performance, as speculated in [41]. However, a larger window also results in a higher computational cost for ML-based techniques, since the cost of feature extraction is roughly an order of magnitude larger than that of the actual classification step. In this work, we evaluated all algorithms on the entire feature set for ease of comparison; however, feature selection techniques should be considered to further reduce the computational cost at inference time, given that a subset of the features (namely, those related to local variations in acceleration) appear to be strongly related to class separability.

Overall, we observed that ML algorithms provided better results when applied to the whole set of data. However, these results should be interpreted with caution keeping in mind that the different datasets have different levels of data imbalance. In fact, while AD^2 contains a similar number of labelled aggressive and non-aggressive events, Ferreira's and Carlos's datasets are heavily skewed towards the aggressive class; in turn, when including normal driving data, all datasets become heavily skewed towards the non-aggressive class. This outcome entails that absolute performance values cannot be directly compared across datasets; nonetheless, it should be noticed that the relative ranking of the different algorithms is quite stable across datasets. It should also be remarked that ML-based predictions may become skewed towards the majority class when trained on highly imbalanced datasets, whereas threshold- and anomaly detection-based techniques may be more robust in this respect.

In this work, we evaluated only shallow neural networks, including both MLP and CNNs. In the future, it would be interesting to evaluate, both in terms of performance and computational burden, deep neural networks capable of processing directly the raw data stream, bypassing the need for feature engineering. These techniques, however, would certainly require the collection of large-scale datasets for training.

A potential limitation of our experiments, and the literature at large, is the number of drivers, and driving conditions, involved in the data acquisition. This is true for both threshold-based and ML-based approaches, which parameters are influenced by many factors including type of vehicle, characteristics of the sensor, and road conditions. In our study, the optimal hyper-parameters varied among different datasets, but the more general question of whether ML-based algorithms would generalize to new unseen scenarios, sensors, and/or drivers is not yet fully answered. More work is needed to understand, e.g., when/how often the ML models should be retrained.

Furthermore, even though our dataset contains events collected by driving twice the same path, we did not use this information (i.e., GPS data) to fine-tune the hyper-parameters. We do believe that GPS data, together with cartographic information could improve event classification. In particular, as soon as a high number of drivers travel a given path, ad-hoc models for a specific GPS location could be built. Alternatively, GPS and cartographic data could be used to identify, at a higher level, if sensors data related to an event have been gathered in a roundabout, on a highway, etc., and apply the corresponding model.

Both sets of experiments showed that the Android smartphone and the AutoPi device allow to achieve, overall similar performance levels. This result confirms the possibility to exploit smartphones as black-boxes, as previously speculated, e.g., in [47], especially when ML-based approaches are used. With regard to anomaly detection- and threshold-based approaches, some differences between the two devices appear. More specifically, when executed on data acquired using the AutoPi device, the algorithms showed, in some cases, better performance. In this respect, it must be recalled that ML-based approaches relied on statistical (summarized) data over a window of readings, whereas anomaly detection- and threshold-based approaches exploited denoised punctual readings. In this view, the fact that during our drives the smartphone was anchored with a suction-cup phone mount (hence, not as firmly as the AutoPi device) could have added additional vibrations not filtered by the denoising process, but filtered when computing the statistical window summaries.

Calculated execution times show that ML-based approaches, when ran on low-resources devices, require between 0.2 and 1 second. In case results need to be provided in a shorter time, algorithms such as GMM, db4, "RSS threshold" and "simple threshold" should be preferred.

Finally, as mentioned in Section V-A, similarly to Ferreira's and Carlos' datasets, the way events were labeled in the dataset suffer from subjectivity of the driver/observer. A possible way to increase the objectivity of collected data is to involve more drivers in the collection phase. In particular, the drivers could drive on multiple paths, or serve as evaluators of other drivers' events. Video recordings could be used as well, as the same travel could be shown multiple time to different evaluators. Another way to build a more objective dataset is to collect sensors data recorded before accidents in which the driver has been considered as responsible for. However, this approach would ignore those events that were harsh, but did not lead to an accident. Finally, an objective dataset could be built also with simulated data, such as SUMO, Simulation of Urban Mobility (SUMO) [71]. However, as reported in [49], simulations are not (yet) able to recreate all nuances of real-world data.

VIII. Conclusions and Future Work

Aggressive driving behaviors are the leading cause of traffic accidents [2]. Devising approaches capable of detecting them

becomes crucial to increase the safety of drivers, pedestrians and (motor)cyclists [72]. With the aim to foster research in this field, in this paper we carried out a comparison of 14 state-of-the-art algorithms for the classification of (aggressive and non-aggressive) driving events. In particular, we tested the algorithms ability to classify aggressive and non-aggressive events by applying them on two datasets available in the literature and on a new dataset specifically created to this purpose. The latter dataset contains events generated by traveling twice the same path with a different driving style, as well as events generated in a variety of situations (e.g., close to traffic lights, in roundabouts, u-turns, etc.), making the analysis more robust and more representative of everyday conditions. Moreover, since it also contains data acquired simultaneously using an Android smartphone and an AutoPi device connected to the OBD-II port, a comparison between smartphones and black-box like devices on the said task was also performed.

In the comparison, three classes of algorithms have been considered: anomaly detection-, threshold- and ML-based. Experimental results show that ML-based approaches were able to achieve the best performance (especially SVM, followed by RF, CNN, and MLP), with a higher computation time; they also outlined the ability of smartphones to support the identification of aggressive events with performance levels similar to those of black-boxes. Low-resource devices, which may not be capable of running ML models onboard or lack a suitable connection for sending acquired data to a remote processing service, could adopt "traditional" threshold-based approaches, with considerably worse performance; in this case, "RSS threshold", GMM and db4 should be preferred.

Future work could consider two aspects, i.e., a) how data are collected, and b) how data are processed. Concerning data collection, research efforts should be devoted to create large-scale datasets, gathering data from a high number of different drivers. In the creation of the datasets, the ratio between aggressive and non-aggressive events should be carefully calibrated, in order to better reflect actual driving situations. Particular attention should be also paid at improving the objectivity of collected data, e.g., by involving multiple evaluators. Regarding data processing, a more thorough analysis of the impact of data imbalance should be performed, e.g., by evaluating the performance of the different algorithms considered in this study on the new large-scale datasets. Such datasets could also be used to further validate the findings of this paper, focusing in particular on the impact that larger windows could have on ML algorithms performance. Other research activities could be devoted to investigate to what extent considering additional data about the driving context (e.g., on the vehicle type, on road characteristics, on weather and traffic conditions, on the driver, etc.) could improve the detection ability of the available algorithms, or could impact their ability to generalize to novel scenarios. Furthermore, it may be interesting to study whether hyper-parameters could be fine-tuned or algorithms performance could be enhanced by leveraging information on events collected on paths that have been traveled multiple times, i.e., removing/mitigating the impact of road features. Finally, efforts should be devoted to expand the comparison to different types of features, as well as considering algorithms that do not require explicit feature engineering, such as deep neural networks, and could benefit of the availability of large-scale datasets. Furthermore, exploring combinations of ML- and threshold-based approaches could help to fine-tune, and further improve, classification results, while reducing the overall computational footprint.

ACKNOWLEDGMENTS

This work was supported by Reale Mutua Assicurazioni.

REFERENCES

[1] World Health Organization, "Global status report on road safety 2018: Summary," Tech. Rep., 2018.
[2] L. Evans, *Traffic safety*, 2004.
[3] P. I. Wouters and J. M. Bos, "Traffic accident reduction by monitoring driver behaviour with in-car data recorders," *Accident Analysis & Prevention*, vol. 32, no. 5, pp. 643–650, 2000.
[4] K. Bahadoor and P. Hosein, "Application for the detection of dangerous driving and an associated gamification framework," in *Proc. 4th IEEE Int. Conf. on Future Internet of Things and Cloud Workshops*, 2016, pp. 276–281.
[5] A. Alessandrini, A. Cattivera, F. Filippi, and F. Ortenzi, "Driving style influence on car CO2 emissions," in *Proc. International Emission Inventory Conference*, 2012.
[6] E. R. Dahlen and K. M. Ragan, "Validation of the propensity for angry driving scale," *Journal of Safety Res.*, vol. 35, no. 5, pp. 557–563, 2004.
[7] M. J. Sullman and A. N. Stephens, "A comparison of the driving anger scale and the propensity for angry driving scale," *Accident Analysis & Prevention*, vol. 58, pp. 88–96, 2013.
[8] C. Lu, F. Hu, D. Cao, J. Gong, Y. Xing, and Z. Li, "Virtual-to-real knowledge transfer for driving behavior recognition: Framework and a case study," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 6391–6402, 2019.
[9] J. Wahlström, I. Skog, and P. Händel, "Smartphone-based vehicle telematics: A ten-year anniversary," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 10, pp. 2802–2825, 2017.
[10] A. B. R. Gonzalez, M. R. Wilby, J. J. V. Diaz, and C. S. Ávila, "Modeling and detecting aggressiveness from driving signals," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 4, pp. 1419–1428, 2014.
[11] M. R. Carlos, L. C. González, J. Wahlström, G. Ramírez, F. Martínez, and G. Runger, "How smartphone accelerometers reveal aggressive driving behavior?–the key is the representation," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 8, pp. 3377–3387, 2020.
[12] R. K. Satzoda and M. M. Trivedi, "Drive analysis using vehicle dynamics and vision-based lane semantics," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 1, pp. 9–18, 2014.
[13] G. Castignani, T. Derrmann, R. Frank, and T. Engel, "Smartphone-based adaptive driving maneuver detection: A large-scale evaluation study," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 9, pp. 2330–2339, 2017.
[14] Z. E. Abou Elassad, H. Mousannif, H. Al Moatassime, and A. Karkouch, "The application of machine learning techniques for driving behavior analysis: A conceptual framework and a systematic literature review," *Eng. Applications of Artificial Intelligence*, vol. 87, p. 103312, 2020.
[15] R. Chandra, U. Bhattacharya, T. Mittal, A. Bera, and D. Manocha, "Cmetric: A driving behavior measure using centrality functions," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 2035–2042.
[16] P. Brombacher, J. Masino, M. Frey, and F. Gauterin, "Driving event detection and driving style classification using artificial neural networks," in *Proc. IEEE Int. Conf. on Industrial Tech.*, 2017, pp. 997–1002.
[17] T. K. Chan, C. S. Chin, H. Chen, and X. Zhong, "A comprehensive review of driver behavior analysis utilizing smartphones," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 10, pp. 4444–4475, 2019.
[18] C.-S. Hsieh and C.-C. Tai, "An improved and portable eye-blink duration detection system to warn of driver fatigue," *Instrumentation Science & Technology*, vol. 41, no. 5, pp. 429–444, 2013.
[19] H. Su and G. Zheng, "A partial least squares regression-based fusion model for predicting the trend in drowsiness," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 38, no. 5, pp. 1085–1092, 2008.
[20] B.-G. Lee and W.-Y. Chung, "A smartphone-based driver safety monitoring system using data fusion," *Sensors*, vol. 12, no. 12, pp. 17 536–17 552, 2012.

[21] Z. Li, Q. Zhang, and X. Zhao, "Performance analysis of k-nearest neighbor, support vector machine, and artificial neural network classifiers for driver drowsiness detection with different road geometries," *Int. J. Distrib. Sens. Netw.*, vol. 13, no. 9, p. 1550147717733391, 2017.

[22] J. Hu, X. Zhang, and S. Maybank, "Abnormal driving detection with normalized driving behavior data: a deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 6943–6951, 2020.

[23] R. Araújo, Â. Igreja, R. De Castro, and R. E. Araujo, "Driving coach: A smartphone application to evaluate driving efficient patterns," in *Proc. IEEE Intelligent Vehicles Symposium*, 2012, pp. 1005–1010.

[24] Y. Xun, J. Liu, N. Kato, Y. Fang, and Y. Zhang, "Automobile driver fingerprinting: A new machine learning based authentication scheme," *IEEE Trans. Ind. Informat.*, vol. 16, no. 2, pp. 1417–1426, 2019.

[25] L. M. Bergasa, D. Almería, J. Almazán, J. J. Yebes, and R. Arroyo, "Drivesafe: An app for alerting inattentive drivers and scoring driving behaviors," in *Proc. IEEE Int. Vehicles Symp.*, 2014, pp. 240–245.

[26] C. Saiprasert, S. Thajchayapong, T. Pholprasit, and C. Tanprasert, "Driver behaviour profiling using smartphone sensory data in a V2I environment," in *Proc. Int. Conf. on Connected Vehicles and Expo*, 2014, pp. 552–557.

[27] D. A. Johnson and M. M. Trivedi, "Driving style recognition using a smartphone as a sensor platform," in *Proc. 14th IEEE Int. Conf. on Intell. Transp. Syst.*, 2011, pp. 1609–1615.

[28] H. Eren, S. Makinist, E. Akin, and A. Yilmaz, "Estimating driving behavior by a smartphone," in *Proc. IEEE Intelligent Vehicles Symposium*. IEEE, 2012, pp. 234–239.

[29] E. Mantouka, E. Barmpounakis, E. Vlahogianni, and J. Golias, "Smartphone sensing for understanding driving behavior: Current practice and challenges," *Int. J. of Transportation Science and Technology*, 2020.

[30] Y. Ma, Z. Zhang, S. Chen, Y. Yu, and K. Tang, "A comparative study of aggressive driving behavior recognition algorithms based on vehicle motion data," *IEEE Access*, vol. 7, pp. 8028–8038, 2018.

[31] M. Fazeen, B. Gozick, R. Dantu, M. Bhukhiya, and M. C. González, "Safe driving using mobile phones," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 3, pp. 1462–1468, 2012.

[32] J. Dai, J. Teng, X. Bai, Z. Shen, and D. Xuan, "Mobile phone based drunk driving detection," in *Proc. 4th Int. Conf. on Pervasive Computing Technologies for Healthcare*, 2010, pp. 1–8.

[33] L. Eboli, G. Mazzulla, and G. Pungillo, "Combining speed and acceleration to define car users' safe or unsafe driving behaviour," *Transportation Research Part C: Emerging Technologies*, vol. 68, pp. 113–125, 2016.

[34] ——, "How to define the accident risk level of car drivers by combining objective and subjective measures of driving style," *Transportation Res. Part F: Traffic Psychology and Behav.*, vol. 49, pp. 29–38, 2017.

[35] M. R. Carlos, M. E. Aragón, L. C. González, H. J. Escalante, and F. Martínez, "Evaluation of detection approaches for road anomalies based on accelerometer readings—addressing who's who," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 10, pp. 3334–3343, 2018.

[36] Y. L. Murphey, R. Milton, and L. Kiliaris, "Driver's style classification using jerk analysis," in *Proc. IEEE Workshop on Computational Intelligence in Vehicles and Vehicular Systems*, 2009, pp. 23–28.

[37] O. Bagdadi and A. Várhelyi, "Jerky driving – An indicator of accident proneness?" *Accid. Anal. Prev.*, vol. 43, no. 4, pp. 1359–1363, 2011.

[38] G. Zylius, "Investigation of route-independent aggressive and safe driving features obtained from accelerometer signals," *IEEE Intell. Transp. Syst. Magazine*, vol. 9, no. 2, pp. 103–113, 2017.

[39] W. Wang, J. Xi, A. Chong, and L. Li, "Driving style classification using a semisupervised support vector machine," *IEEE Trans. on Human-Machine Systems*, vol. 47, no. 5, pp. 650–660, 2017.

[40] M. M. Bejani and M. Ghatee, "A context aware system for driving style evaluation by an ensemble learning on smartphone sensors data," *Transp. Res. Part C Emerg. Technol.*, vol. 89, pp. 303–320, 2018.

[41] J. Ferreira, E. Carvalho, B. V. Ferreira, C. de Souza, Y. Suhara, A. Pentland, and G. Pessin, "Driver behavior profiling: An investigation with different smartphone sensors and machine learning," *PLoS One*, vol. 12, no. 4, p. e0174959, 2017.

[42] H. R. Eftekhari and M. Ghatee, "A similarity-based neuro-fuzzy modeling for driving behavior recognition applying fusion of smartphone sensors," *Journal of Intell. Transp. Syst.*, vol. 23, no. 1, pp. 72–83, 2019.

[43] M. U. Ahmed and S. Begum, "Convolutional neural network for driving maneuver identification based on inertial measurement unit (IMU) and global positioning system (GPS)," *Frontiers in Sustainable Cities*, vol. 2, p. 34, 2020.

[44] I. Silva and J. Eugenio Naranjo, "A systematic methodology to evaluate prediction models for driving style classification," *Sensors*, vol. 20, no. 6, p. 1692, 2020.

[45] H. R. Eftekhari and M. Ghatee, "Hybrid of discrete wavelet transform and adaptive neuro fuzzy inference system for overall driving behavior recognition," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 58, pp. 782–796, 2018.

[46] A. Yuksel and S. Atmaca, "Driver's black box: A system for driver risk assessment using machine learning and fuzzy logic," *Journal of Intell. Transp. Syst.*, pp. 1–48, 2020.

[47] E. I. Vlahogianni and E. N. Barmpounakis, "Driving analytics using smartphones: Algorithms, comparisons and challenges," *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 196–206, 2017.

[48] J. Paefgen, F. Kehr, Y. Zhai, and F. Michahelles, "Driving behavior analysis with smartphones: insights from a controlled field study," in *Proc. 11th Int. Conf. on Mobile and Ubiq. Multimedia*, 2012, pp. 1–8.

[49] M. Matousek, E.-Z. Mohamed, F. Kargl, C. Bösch *et al.*, "Detecting anomalous driving behavior using neural networks," in *Proc. IEEE Intelligent Vehicles Symposium*, 2019, pp. 2229–2235.

[50] L. Morra, F. Lamberti, F. G. Pratticó, S. L. Rosa, and P. Montuschi, "Building trust in autonomous vehicles: Role of virtual reality driving simulators in HMI design," *IEEE Trans. Veh. Technol.*, vol. 68, no. 10, pp. 9438–9450, 2019.

[51] F. G. Pratticò, F. Lamberti, A. Cannavò, L. Morra, and P. Montuschi, "Comparing state-of-the-art and emerging augmented reality interfaces for autonomous vehicle-to-pedestrian communication," *IEEE Trans. Veh. Technol.*, vol. 70, no. 2, pp. 1157–1168, 2021.

[52] V. L. Neale, T. A. Dingus, S. G. Klauer, J. Sudweeks, and M. Goodman, "An overview of the 100-car naturalistic study and findings," in *Proc. 19th Int. Technical Conf. on the Enhanced Safety of Vehicles*, 2005.

[53] E. Romera, L. M. Bergasa, and R. Arroyo, "Need data for driver behaviour analysis? presenting the public uah-driveset," in *Proc. 19th IEEE Int. Conf. on Intell. Transp. Syst.*, 2016, pp. 387–392.

[54] J. L. Deffenbacher, E. R. Oetting, and R. S. Lynch, "Development of a driving anger scale," *Psychol. Reports*, vol. 74, no. 1, pp. 83–91, 1994.

[55] AXA. Driver telematics analysis – Use telematic data to identify a driver signature. [Online]. Available: https://www.kaggle.com/c/axa-driver-telematics-analysis

[56] A. H. Ali, A. Atia, and M.-S. M. Mostafa, "Recognizing driving behavior and road anomaly using smartphone sensors," *International Journal of Ambient Computing and Intelligence*, vol. 8, no. 3, pp. 22–37, 2017.

[57] A. S. Zeeman and M. J. Booysen, "Combining speed and acceleration to detect reckless driving in the informal public transport industry," in *Proc. 16th Int. IEEE Conf. on Intell. Transp. Syst.*, 2013, pp. 756–761.

[58] Y. Li, F. Xue, L. Feng, and Z. Qu, "A driving behavior detection system based on a smartphone's built-in sensor," *International Journal of Communication Systems*, vol. 30, no. 8, p. e3178, 2017.

[59] K. C. Baldwin, D. D. Duncan, and S. K. West, "The driver monitor system: A means of assessing driver performance," *Johns Hopkins APL Technical Digest*, vol. 25, no. 3, pp. 269–277, 2004.

[60] T. Osafune, T. Takahashi, N. Kiyama, T. Sobue, H. Yamaguchi, and T. Higashino, "Analysis of accident risks from driving behaviors," *Int. J. of Intell. Transp. Syst. Research*, vol. 15, no. 3, pp. 192–202, 2017.

[61] M. A. Ylizaliturri-Salcedo, M. Tentori, and J. A. Garcia-Macias, "Detecting aggressive driving behavior with participatory sensing," in *Proc. Int. Conf. on Ubiquitous Comp. and Ambient Intell.*, 2015, pp. 249–261.

[62] S. Chigurupati, S. Polavarapu, Y. Kancherla, and A. K. Nikhath, "Integrated computing system for measuring driver safety index," *Int. J. of Emerging Technology and Advanced Engineering*, vol. 2, no. 6, 2012.

[63] F. Feng, S. Bao, J. R. Sayer, C. Flannagan, M. Manser, and R. Wunderlich, "Can vehicle longitudinal jerk be used to identify aggressive drivers? An examination using naturalistic driving data," *Accident Analysis & Prevention*, vol. 104, pp. 125–136, 2017.

[64] B. Bose, J. Dutta, S. Ghosh, P. Pramanick, and S. Roy, "Smartphone based system for real-time aggressive driving detection and marking rash driving-prone areas," in *Proc. Workshop Program of the 19th Int. Conf. on Distributed Comp. and Networking*, 2018, pp. 1–6.

[65] W. Wang, J. Xi, and J. K. Hedrick, "A learning-based personalized driver model using bounded generalized gaussian mixture models," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 11 679–11 690, 2019.

[66] W. Wang, S. Chen, and G. Qu, "Incident detection algorithm based on partial least squares regression," *Transportation Research Part C: Emerging Technologies*, vol. 16, no. 1, pp. 54–70, 2008.

[67] S.-R. G. Christopoulos, S. Kanarachos, and A. Chroneos, "Learning driver braking behavior using smartphones, neural networks and the sliding correlation coefficient: Road anomaly case study," *IEEE Trans. on Int. Transportation Systems*, vol. 20, no. 1, pp. 65–74, 2018.

[68] Y.-T. Liu, Y.-Y. Lin, S.-L. Wu, C.-H. Chuang, M. Prasad, and C.-T. Lin, "EEG-based driving fatigue prediction system using functional-link-

based fuzzy neural network," in *Proc. International Joint Conference on Neural Networks*, 2014, pp. 4109–4113.
[69] D. Wei and H. Liu, "Analysis of asymmetric driving behavior using a self-learning approach," *Transportation Research Part B: Methodological*, vol. 47, pp. 1–14, 2013.
[70] M. D. Tundo, E. Lemaire, and N. Baddour, "Correcting smartphone orientation for accelerometer-based analysis," in *Proc. IEEE International Symposium on Medical Measurements and Appl.*, 2013, pp. 58–62.
[71] M. Behrisch, L. Bieker, J. Erdmann, and D. Krajzewicz, "Sumo–simulation of urban mobility: an overview," in *Proc. 3rd Int. Conf. on Advances in System Simulation*. ThinkMind, 2011.
[72] J. Wang, J. Liu, and N. Kato, "Networking and communications in autonomous driving: A survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1243–1274, 2018.

Valentina Gatteschi (M'19-SM'19) is an assistant professor at the Department of Control and Computer Engineering of Politecnico di Torino, Italy, where she received her B.Sc. and M.Sc. degrees in management engineering and her Ph.D. degree in computer engineering in 2005, 2008 and 2013, respectively. Her research interests include intelligent systems, semantic computing and blockchain technology.

Alberto Cannavò received the B.Sc. degree from University of Messina, Italy, in 2013. Then, he received the M.Sc. and the Ph.D. degrees in computer engineering from Politecnico di Torino, Italy, in 2015 and 2020, respectively. Currently, he is a postdoctoral fellow at the Department of Control and Computer Engineering of Politecnico di Torino. His fields of interest include computer graphics and human-machine interaction.

Fabrizio Lamberti (M'02-SM'14) is a full professor at the Department of Control and Computer Engineering of Politecnico di Torino, Italy, where he has the responsibility for the VR@POLITO hub. His research interests encompass computer graphics, HMI, and intelligent computing. He serves as Associate Editor for several journals including IEEE Transactions on Computers, IEEE Transactions on Consumer Technologies, and IEEE Transactions on Learning Technologies. More information at https://staff.polito.it/fabrizio.lamberti.

Lia Morra (M'17-SM'19) is an assistant professor at the Department of Control and Computer Engineering of Politecnico di Torino, Italy. Previously, she was with im3D (Turin, Italy), where she served as Chief Scientific Officer from 2014 to 2017 developing artificial intelligence systems for medical image interpretation. Her main research interests are computer vision, pattern recognition, and machine learning.

Paolo Montuschi (M'90-SM'07-F'14) is a full professor and Rector's Delegate for IT Systems at Politecnico di Torino, Italy. His research interests include computer arithmetic, computer graphics, and intelligent systems. He is an IEEE Fellow, a life member of the International Academy of Sciences in Turin, and of IEEE Eta Kappa Nu. He serves as the Editor-in-Chief of the IEEE Transactions on Emerging Topics in Computing and the 2020-21 Chair of the IEEE TAB/ARC. More information at http://staff.polito.it/paolo.montuschi.