

A Neural-Network-Based Framework for Supporting Driver Classification and Analysis

Alfredo Cuzzocrea[1(✉)], Francesco Mercaldo[2], and Fabio Martinelli[2]

[1] University of Calabria, Rende, Italy
alfredo.cuzzocrea@unical.it
[2] IIT-CNR, Pisa, Italy
{francesco.mercaldo,fabio.martinelli}@iit.cnr.it

Abstract. The proliferation of info-entertainment systems in nowadays vehicles has provided a really cheap and easy-to-deploy platform with the ability to gather information about the vehicle under analysis. The infrastructure in which these information can be used to improve safety and security are provided by Ultra-response connectivity networks with a latency of below 10 ms. In this paper, we propose a service-oriented architecture based on a fully connected neural network architecture considering position-based features aimed to detect in real-time: (i) the driver and (ii) the driving style, with the goal of providing an architecture for increasing security and safety in automotive context. The experimental analysis performed on real-world data shows that the proposed method obtains encouraging results.

Keywords: Automotive · Artificial Intelligence · Neural network · Machine learning

1 Introduction

The Internet of Things is represented by the network of physical devices, home appliances, vehicles and a plethora of devices embedded with actuators, sensors, software and, in general, with connectivity [33]. More attention has been recently obtained by the Internet of Vehicle (i.e., IoT related to automotive) [28]. In fact, due to this technology a vehicle can turn from an autonomous and independent system into a component of a more efficient cooperative one [15].

As a matter of fact, vehicles can send data between them or to an external server in order to adapt the driving style to the road condition; traffic lights can send information about the number of cars in a lane in order to suggest to other cars an alternative path to avoid traffic congestion [22]. In general, a very low end-to-end latency of below 10 ms is required for vehicle safety applications. Nevertheless, a latency less than 1 ms will be required considering the bi-directional exchange of data is required with these emerging applications.

© Springer Nature Switzerland AG 2021
O. Gervasi et al. (Eds.): ICCSA 2021, LNCS 12951, pp. 28–39, 2021.
https://doi.org/10.1007/978-3-030-86970-0_3

Nowadays, several useful features from vehicles while are traveling can be gathered by the proliferation of both mobile devices and info-entertainment systems in modern cars. This has boosted the research community to focus about safety and security in the automotive context. In this scenario, data mining algorithms can be useful to analyze features collected from vehicles with the aim to extract knowledge related to the driver behavioral analysis. For this reason, driver detection and driving style recognition are becoming the main research trends. In particular, the main aims regarding driver are to develop systems proposing ad-hoc policies for single driver (for insurance companies) and anti-theft system (for end users).

New anti-theft paradigms are necessary, considering that cars are equipped with many computers on board, exposing them to a new type of attacks [1,20]. As any software, the exposition to bug and vulnerabilities of the operating systems running on cars has to be considered [30]. Auto insurance markets is interested in driver detection. In fact, through these plethora of new data available and by measuring and rating a particular person's driving ability insurance may estimate the individual risk [4]. These paradigms can also be extended via different-but-related initiatives, such as the traditional the context of adaptive and flexible paradigms in web intelligence (e.g., [3,8]).

This idea is defined into the Usage Based Insurance (UBI) concept, introduced into the personal motor insurance market over a decade ago. Basically, two typical models: "pay as you drive" and "pay how you drive" are adopted. Moreover, time of usage, distance driven, driving behavior and places driven to are the main features impacting the premiums. Relating to driving style recognition, this trend is more related to safety than driver detection but currently there are few studies proposing methods to address this issue.

The features gathered from Controller Area Network (CAN) bus, a vehicle bus standard designed to allow micro-controllers and devices to communicate with each other in applications without a host computer are exploited from several methods aimed to the driver detection [24]. Brake usage and fuel consumption rate are just two examples of information passing though this bus.

Different methods proposed in the current literature consider several sets of features extracted from the CAN bus. Thus different vehicle manufacturers provides different features (apart from a small subset common to all manufacturers). Thus, it is possible that a method evaluated on a vehicle produced by a certain manufacturer may not work on a vehicle by another manufacturer or on a new version of the same vehicle.

Furthermore, for exploiting data from CAN bus the proposed methodologies must have access to these data. This results in possible communication delay, critical infrastructure errors, and in possible attacks, as discussed in [23]. Starting from these considerations, in this paper we propose a method aimed to (i) discriminate between the vehicle owner and impostors and (ii) detect the driving style. A fully connected neural network architecture exploiting a feature vector based on positioning systems is proposed. The goal is to detect the driver and the driving style without accessing to critical information CAN bus-related.

The paper is organized as follows: Sect. 2 present the state of the ark works in the field. Section 3 motivate the work and underline the main contribution.

Finally, Sect. 4 show the results of the experiments and Sect. 5 concludes the work and proposes new lines of research.

2 Related Work

Current literature related to the driver behavioral analysis in discussed in following section.

In [34], researchers by analyzing data from the steering wheel and the accelerator and by training a Hidden Markov model (HMM) classifies the driver characteristics. In particular, they build two models for each driver, one trained from accelerator data and one learned from steering wheel angle data. They achieve an accuracy of 85% in identifying the different drivers. Instead, in [32] the authors observe the signals of the driver following another vehicle in order to identify her behavior. They analyzed signals, as accelerator pedal, brake pedal, vehicle velocity, and distance from the vehicle in front, were measured using a driving simulator. The results achieved an identification rates of 73% for thirty real drivers and 81% for twelve drivers using a driving simulator.

For modeling driver behavior a HMM based similarity measure has been proposed in [14]. They employ a simulated driving environment to test the effectiveness of the proposed solution. A method based on driving pattern of the car has been proposed in [18]. In particular, they apply four classifiers: the k Nearest Neighbor, RandomForest, Multilayer Perceptron and Decision Tree to mechanical feature extracted from the CAN and they achieved performance of 84%, 96%, 74% and 93%, respectively. Researchers in [13] applies Temporal and Spatial Identifiers in order to control the road environment and driver behavior profiles to provide a common measure of driving behavior based on the risk of a casualty crash for assessing the effectiveness of a PAYD (i.e., pay-as-you drive) scheme on reducing driving risks. They prove that in order to induce significant changes, in several cases personalized feedback alone are sufficient. Nevertheless, if a financial incentives are applied, the risk reductions is much greater.

In [13] researcher proposed a biometric driver recognition system utilizing driving behaviors. Authors, using Gaussian mixture models, extract a set of features from the accelerator and brake pedal pressure were then used as inputs to a fuzzy neural network system to ascertain the identity of the driver. The effectiveness of relating the use of the FNN for real-time driver identification and verification has been proven in the experiments.

In [17], researchers consider data coming from the real-time in-car video in order to detect near vehicles for safety, autodriving, and target tracing. They propose an approach with the aim to localize target vehicles in video under various environmental conditions. The motion in the field of view is modeled according to the scene characteristic. Furthermore, a HMM is trained by the vehicle motion model with the goal of probabilistically separate target vehicles from the background and track them. By considering different light setting and street types they conclude that their method is robust with respect to side conditions. Working with these data is difficult because of privacy regulation; nevertheless, innovative techniques to deal with this topic have been developed [10].

Trasarti et al. [31] extract mobility profiles of individuals from raw GPS traces studying how to match individuals based on profiles. The profile matching problem is instantiated to a specific application context, namely proactive car pooling services. Then a matching criterion that satisfies various basic constraints obtained from the background knowledge of the application domain is developed.

Another way to collect data useful for observing driver behavior is by means of questionnaire as done in [19]. Despite being subjected to cognitive bias they provide a means for studying driving behaviors, which could be difficult or even impossible to study by using other methods like observations, interviews and analysis of national accident statistics. Their findings demonstrate that bias caused by socially desirable responding is relatively small in driver behavior questionnaire responses.

The proposal of an architecture able to identify the driver and the driving style exploiting features related to the accelerometer sensor available from the info-entertainment system or in a mobile device fixed in the vehicle is the main difference with regard to the elicited works. This represents the first attempt to detect the driver and the driving style with the same feature set, overcoming the state of the art literature in terms of performances. This innovation is the first attempt to infer both the identity of the driver and her driving style with the same feature set.

3 Modeling and Analyzing the Driver Behavior

In the following we describe the high-level architecture and the method we proposed aimed to: (i) identify the driver and (ii) detect the driving style. Table 1 shows the set of features available from the accelerometer.

Table 1. Features involved in the study.

Feature	Description	Measured in
F1	Acceleration in X	(Gs)
F2	Acceleration in Y	(Gs)
F3	Acceleration in Z	(Gs)
F4	Acceleration in X filtered by KF	(Gs)
F5	Acceleration in Y filtered by KF	(Gs)
F6	Acceleration in Z filtered by KF	(Gs)
F7	Roll	(degrees)
F8	Pitch	(degrees)
F9	Yaw	(degrees)

Those features are sampled by the inertial sensors (available on mobile phones and on info-entertainment systems) at a frequency 10 Hz. In particular, since

mobile phone sampled with a frequency 100 Hz the time series is reduced by taking the mean of every 10 samples. With the aim to gather data, the mobile device is fixed on the windshield at the start of the travel, in this way the axes are the same during the whole trip. In the calibration process lateral axis of the vehicle is set to the Y axis(reflects turnings) while the longitudinal axis is set to the X axis (positive value reflects an acceleration, negative a braking).

A Kalman filter (KF) is applied to the observations [26]. KF is an algorithm exploiting a series of measurements observed over time, containing statistical noise and other inaccuracies, and produces estimates of unknown variables that tend to be more accurate than those based on a single measurement alone, by estimating a joint probability distribution over the variables for each timeframe. Guidance, navigation, and control of vehicles, particularly aircraft and spacecraft are common applications [12]. In order to test our method with all the possible noises in the real application we consider real-world data and not simulated one. This enables us to consider all the possible variables that are not considerable from simulated environments.

3.1 The Classification Process

We consider supervised techniques. To build models we consider the following three main steps: (*i*) data instances *pre-processing* (data cleaning and labeling), (*ii*) *training* the designed neural network and (*iii*) classification performance assessment (*Testing*). Cleaning of the raw instances of the feature by removing incomplete and wrong instances is the main process of *pre-processing*.

Figure 1 show the architecture of the *Training* phase: the inferred function is generated starting from the features gathered from the acceleration data. The inferred function, provided by the neural network, should be able to discriminate between accelerometer features belonging to several classes i.e., the function should define the boundaries between the numeric vectors belonging to several classes.

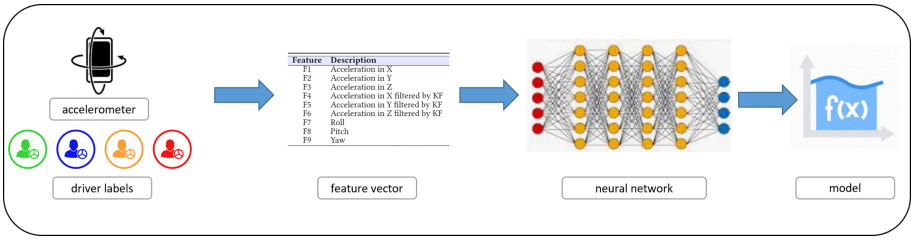

Fig. 1. The *Training* step.

Drivers and the driver behaviors are the two classes of labels used to discriminate. Those labels must be computed free from any bias for the training set in order to let the supervised machine learning techniques to infer the right

function. The third step, once generated the model is the evaluation of its effectiveness: the *Testing* one, is depicted in Fig. 2.

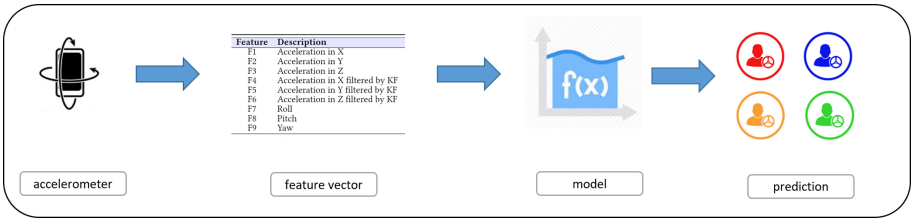

Fig. 2. The *Testing* step.

Well-know information retrieval metrics are used to measure the effectiveness of the models. Furthermore, instances not included into the training data must be used in order to evaluate the (real) performance of the built models. To assure this, we consider the cross validation: we split the full dataset in two parts and we used the first part as training dataset and the second one as testing dataset. In order to evaluate all the instances belonging to the full dataset, this process is repeated considering different instances for both training and testing dataset.

Two different classifiers are defined: the first one is used for the *driver* detection, while the second one is used for her *driving style*. It is worth noting that both classifiers considers the same accelerometer feature set. To train the classifiers, with regard to the *driver* detection, we defined T as a set of labeled traces *(M, l)*, where each M is associated to a label $l \in \{U_1, \ldots, U_n\}$ (where U_n represents the n-th driver under analysis i.e., *Driver #1, Driver #2, Driver #3, Driver #4, Driver #5 and Driver #6*). For the *driving style* classifier, the set of labeled traces *(M, l)* is defined as T. Where M is associated to a label $l \in \{U_1, \ldots, U_n\}$ with U_n representing different driving style i.e., *Normal*, *Drowsy* and *Aggressive*. A feature vector V_f is used as input for the classifier during training for each M.

In order to perform assessment during the training step, a k-fold cross-validation is used [29]: the dataset is subdivided into k subsets using random sampling. We use a subset as a validation dataset for assessing the performance of the trained model, while we use the other $k-1$ subsets for the training phase. We repeat this procedure $k = 5$ times. In each of the ten iterations, just one subset of the k defined has been used as validation set. To obtain a single reliable estimate, the final results are evaluated by computing the average of the results obtained during the ten iterations. As first step the dataset D is partitioned in k slices.

Then, the following steps are executed for each iteration i: (1) the training set $T_i \subset$D is generated by selecting an unique set of k–1 slices from the dataset D; (2) we generate the test set $T_i' = D - T_i$ by considering the kth slice (it is evaluated as the complement of T_i to D); (3) T_i is used as training set for the classifier; (4) the trained classifier is applied to T_i' to evaluate accuracy.

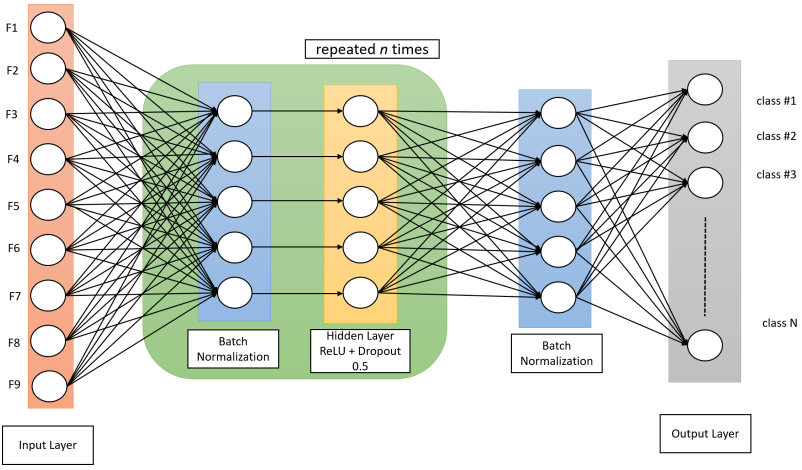

Fig. 3. The designed neural network.

Since $k = 5$, for each iteration i 80% of the dataset is used as training set (T_i), while the remaining 20% is used as a test set (T_i'). The final step has the goal of assessing the performances of the method by running the trained classifiers on real user data. The aim is to identify the driver under analysis (with the first model) and the driving style (with the second model) in real-time, the adopted features model and feeding with it the trained classifier.

3.2 The Neural Network Architecture

In order to build the two models related to the *driver detection model* and *driving style detection model* we propose the neural network architecture depicted in Fig. 3. Figure 3 shows the neural network architecture which number of hidden layers empirically set as equal to 25.

With the aim to generate the classifier model, the supervised classification approach considers a neural network architecture composed by the following sets: (*i*) *Input layer*: this layer is used as an entry point into the network, there is a node for each considered feature; (*ii*) *Hidden layers*: artificial neurons such as perceptrons composed it. The output of each artificial neuron is the application of an activation function to the weighted sum of its input. Usually the functions used are sigmoid, a ReLu or a soft plus function; (*iii*) *Dropout layer*: this layer implements a regularization technique, which aims to reduce the complexity of the model with the goal to prevent over-fitting. Several neurons are randomly deactivated following a Bernoulli distribution of parameter p. Then, the obtained network is trained and the deactivated node are set in their original position with their original weights. In the training stages, the "drop" probability for each hidden node is usually in the range [0,0.5] (for this study 0.5 was used because dropping a neuron with 0.5 probability gets the highest variance for the

distribution); (*iv*) *Batch Normalization*: in order to improve the training of feed-forward neural networks, batch normalization is used [16]. It is a method that allows to improve the speed of learning and enables the usage of higher learning rates, more flexible parameter initialization, and saturating non-linearities. We adopted this approach since batch normalized models achieve higher accuracy on both validation and test, due to a stable gradient propagation within the network; (*v*) *Output layer*: the output of the networks are produced in this last layer. The layer used is a dense one using a linear operator in which each input is connected to each output.

Figure 3 shows the neural network architecture used in this work. Cross-entropy has been selected as loss function and the stochastic gradient descent method has been used for learning optimization [21].

3.3 Study Design

We design an experiment aimed to investigate whether the feature set we consider is useful to discriminate between different drivers and different driving style using the proposed neural network architecture. In particular, the experiments aims to verify if the considered features can provide a prediction of unseen driver instances, and unseen driving style. The neural network described in the previous section is used for classification by using the 9 features gathered from accelerometer as input neurons.

The evaluation consists of a classification analysis aimed at assessing whether the considered features are able to correctly discriminate between different drivers and driving styles instances. In order to identify a unique feature of an instance under analysis we use each layer in the neural network [27]. In particular, each layer pick out a particular feature in order to classify the instance. Due to this reason, we choose to not apply a procedure of feature selection. Nevertheless, descriptive statistics and hypothesis testing is used for providing evidence of statistical impact about the effectiveness of the proposed feature vector.

4 Experimental Assessment and Evaluation

In this section we describe the real-world dataset considered in the evaluation of the proposed method and the results of the experiment. The libraries *Tensorflow*[1] (an open source software library for high-performance numerical computation) and the library *Keras*[2] (a high-level neural networks API, able to run on top of TensorFlow) are used to implement the designed neural network architecture. In order to plot the results, *Matplot*[3] is used. We developed the network using the Python programming language. We use an Intel Core i7 8th gen, equipped with 2GPU and 16Gb of RAM to run the experiments.

[1] https://www.tensorflow.org/.
[2] https://keras.io/.
[3] https://matplotlib.org/.

4.1 The Dataset

The dataset considered in the evaluation was gathered from a smartphone fixed in the car using an adequate support [26]. The experiments considers the data collected from six drivers that were involved in the experiment. More than 8 h of observations have been monitored and stored in the dataset. Each observation has been assigned to a particular driver (i.e., *Driver #1*, *Driver #2*, *Driver #3*, *Driver #4*, *Driver #5* and *Driver #6*) and the driving style (i.e., *Normal*, *Drowsy* and *Aggressive*).

The three driving styles (*Normal*, *Drowsy* and *Aggressive*) were performed in two different routes, one is 25 km (round trip) in a motorway type of road with normally 3 lanes on each direction and 120 km/h of maximum allowed speed, and the other is around 16 km in a secondary road of normally one lane on each direction and around 90km/h of maximum allowed speed. At the link http://www.robesafe.com/personal/eduardo.romera/uah-driveset/ the interested reader can find the anonymized data used for the tests. The labels driver and the driving style are manually added to each instance of the dataset by researchers in [26].

4.2 Classification Analysis

In the follow we present the classification results obtained from both the neural network and machine learning classification approaches (considered as baseline) on the real-world dataset we evaluate. For the models considering driving style detection, the neural network has 25 hidden layers. Since the neural networks reached the stability with less than 100 epochs, the number of epochs is set to 100.

Figure 4 shows the performances (in terms of accuracy and loss) related to *Driver #1* and *Driver #2* detection. Due to length constraints the plots for the remaining drivers are not reported. Nevertheless, conclusions can be drawn.

The performances of the proposed neural network model for the driver detection problem are the following: (*i*) *Driver #1*: the obtained accuracy is equal to 0.99644 while the relative loss is equal to 0.40479; (*ii*) *Driver #2*: an accuracy of 0.99833 and a loss of 0.40272 is achieved by this driver; (*iii*) *Driver #3*: an accuracy of 0.99833 and a loss of 0.40307 is achieved by the design network for this driver; (*iv*) *Driver #4*: this driver exhibits an accuracy equal to 0.99833 and a loss of 0.40277; (*v*) *Driver #5*: the proposed methodology achieves accuracy equal to 0.99833 and a loss equal to 0.40274; (*vi*) *Driver #6*: an accuracy of 0.99833 and a loss of 0.40273 is achieved for this driver.

In detail we obtain following results: (1) *Normal Style*: the method achieves an accuracy equal to 0.98 and a loss of 0.18596; (2) *Drowsy Style*: for this driving style the accuracy achieved is equal to 0.92003 and the loss is equal to 0.28644; (3) *Aggressive Style*: for the aggressive style the accuracy reached by the designed network is equal to 0.98004 and the loss is equal to 0.17792.

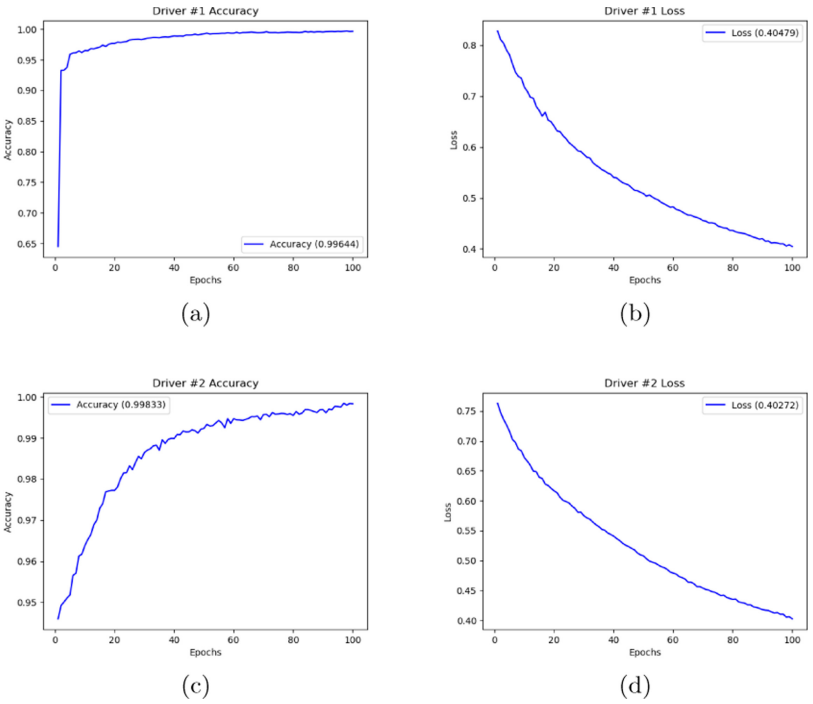

Fig. 4. Driver detection performances: (a) accuracy for *Driver #1*; (b) loss for *Driver #1*; (c) accuracy for *Driver #2*; (d) loss for *Driver #2*.

5 Conclusions and Future Work

In this paper we presented a neural network architecture aimed to provide a framework to analyze driver behaviors aimed to detect the driver and the driving style. The proposed method is evaluated on a real-world dataset using 6 different drivers and by obtaining the analysed feature vector from the accelerometer integrated in a fixed mobile device. The results on the instances considered show results of accuracy equal to 99% for the identification of the driver. Furthermore, the accuracy with respect the driver style detection has been proven to be between 92% and 98%.

As future work, we plan to investigate whether the application of formal methods can be useful for the driver/driving style identification, which have been already successfully used in other domains like, for instance, malware detection [2,7,25] and in system biology [5]. Finally, another interesting line of research consists in extending our proposed framework with features proper of the emerging big data trend (e.g., [6,9,11]).

References

1. Alheeti, K.M.A., Gruebler, A., McDonald-Maier, K.D.: An intrusion detection system against malicious attacks on the communication network of driverless cars. In: 2015 12th Annual IEEE Consumer Communications and Networking Conference (CCNC), pp. 916–921. IEEE (2015)
2. Canfora, G., Martinelli, F., Mercaldo, F., Nardone, V., Santone, A., Visaggio, C.A.: Leila: formal tool for identifying mobile malicious behaviour. IEEE Trans. Softw. Eng. **45**(12), 1230–1252 (2018)
3. Cannataro, M., Cuzzocrea, A., Pugliese, A.: XAHM: an adaptive hypermedia model based on XML. In: Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, pp. 627–634 (2002)
4. Carfora, M.F., et al.: A "pay-how-you-drive" car insurance approach through cluster analysis. Soft. Comput. **23**(9), 2863–2875 (2018). https://doi.org/10.1007/s00500-018-3274-y
5. Ceccarelli, M., Cerulo, L., Santone, A.: De novo reconstruction of gene regulatory networks from time series data, an approach based on formal methods. Methods **69**(3), 298–305 (2014)
6. Chatzimilioudis, G., Cuzzocrea, A., Gunopulos, D., Mamoulis, N.: A novel distributed framework for optimizing query routing trees in wireless sensor networks via optimal operator placement. J. Comput. Syst. Sci. **79**(3), 349–368 (2013)
7. Cimitile, A., Mercaldo, F., Nardone, V., Santone, A., Visaggio, C.A.: Talos: no more ransomware victims with formal methods. Int. J. Inf. Secur. **17**(6), 719–738 (2017). https://doi.org/10.1007/s10207-017-0398-5
8. Cuzzocrea, A.: Combining multidimensional user models and knowledge representation and management techniques for making web services knowledge-aware. Web Intell. Agent Syst. Int. J. **4**(3), 289–312 (2006)
9. Cuzzocrea, A., De Maio, C., Fenza, G., Loia, V., Parente, M.: OLAP analysis of multidimensional tweet streams for supporting advanced analytics. In: Proceedings of the 31st Annual ACM Symposium on Applied Computing, pp. 992–999 (2016)
10. Cuzzocrea, A., De Maio, V., Fadda, E.: Experimenting and assessing a distributed privacy-preserving olap over big data framework: principles, practice, and experiences. In: 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1344–1350 (2020)
11. Cuzzocrea, A., Mansmann, S.: Olap visualization: models, issues, and techniques. In: Encyclopedia of Data Warehousing and Mining, Second Edition, pp. 1439–1446. IGI Global (2009)
12. Einicke, G.A.: Smoothing, filtering and prediction: estimating the past, present and future. Technical report, Intech, Rijeka (2012). ISBN 978-953-307-752-9
13. Ellison, A.B., Bliemer, M.C., Greaves, S.P.: Evaluating changes in driver behaviour: a risk profiling approach. Accid. Anal. Prev. **75**, 298–309 (2015)
14. Enev, M., Takakuwa, A., Koscher, K., Kohno, T.: Automobile driver fingerprinting. Proc. Priv. Enhancing Technol. **2016**(1), 34–50 (2016)
15. Fettweis, G.P.: The tactile internet: applications and challenges. IEEE Veh. Technol. Mag. **9**(1), 64–70 (2014)
16. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning, ICML 2015, vol. 37, pp. 448–456. JMLR.org (2015)

17. Jazayeri, A., Cai, H., Zheng, J.Y., Tuceryan, M.: Vehicle detection and tracking in car video based on motion model. IEEE Trans. Intell. Transp. Syst. **12**(2), 583–595 (2011)
18. Kwak, B.I., Woo, J., Kim, H.K.: Know your master: driver profiling-based anti-theft method. In: PST 2016, pp. 211–218 (2016)
19. Lajunen, T., Summala, H.: Can we trust self-reports of driving? effects of impression management on driver behaviour questionnaire responses. Transport. Res. F: Traffic Psychol. Behav. **6**(2), 97–107 (2003)
20. Lyamin, N., Vinel, A., Jonsson, M., Loo, J.: Real-time detection of denial-of-service attacks in IEEE 802.11 p vehicular networks. IEEE Commun. Lett. **18**(1), 110–113 (2014)
21. Mannor, S., Peleg, D., Rubinstein, R.: The cross entropy method for classification. In: Proceedings of the 22nd International Conference on Machine Learning, ICML 2005, pp. 561–568. ACM, New York (2005)
22. Martinelli, F., Mercaldo, F., Nardone, V., Orlando, A., Santone, A.: Context-awareness mobile devices for traffic incident prevention. In: 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), pp. 143–148. IEEE (2018)
23. Martinelli, F., Mercaldo, F., Nardone, V., Santone, A.: Car hacking identification through fuzzy logic algorithms. In: 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), pp. 1–7. IEEE (2017)
24. Martinelli, F., Mercaldo, F., Orlando, A., Nardone, V., Santone, A., Sangaiah, A.K.: Human behavior characterization for driving style recognition in vehicle system. Comput. Electr. Eng. **83**, 102504 (2020)
25. Mercaldo, F., Nardone, V., Santone, A., Visaggio, C.A.: Download malware? No, thanks. How formal methods can block update attacks. In: 2016 IEEE/ACM 4th FME Workshop on Formal Methods in Software Engineering (FormaliSE), pp. 22–28. IEEE (2016)
26. Romera, E., Bergasa, L.M., Arroyo, R.: Need data for driver behaviour analysis? Presenting the public UAH-DriveSet. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 387–392. IEEE (2016)
27. Schmidhuber, J.: Deep learning in neural networks: an overview. Neural Netw. **61**, 85–117 (2015)
28. Skibinski, J., Trainor, J., Reed, C.: Internet-based vehicle communication network. SAE Trans. 820–825 (2000)
29. Stone, M.: Cross-validatory choice and assessment of statistical predictions. J. Roy. Stat. Soc. Ser. B **36**, 111–147 (1974)
30. Taylor, A., Leblanc, S., Japkowicz, N.: Anomaly detection in automobile control network data with long short-term memory networks. In: 2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA), pp. 130–139. IEEE (2016)
31. Trasarti, R., Pinelli, F., Nanni, M., Giannotti, F.: Mining mobility user profiles for car pooling. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1190–1198. ACM (2011)
32. Wakita, T., et al.: Driver identification using driving behavior signals. IEICE Trans. Inf. Syst. **89**(3), 1188–1194 (2006)
33. Xia, F., Yang, L.T., Wang, L., Vinel, A.: Internet of things. Int. J. Commun. Syst. **25**(9), 1101–1102 (2012)
34. Zhang, X., Zhao, X., Rong, J.: A study of individual characteristics of driving behavior based on hidden Markov model. Sens. Transducers **167**(3), 194 (2014)