IET Cyber-Physical Systems: Theory & Applications

Research Article

Performance validation of vehicle platooning through intelligible analytics

ISSN 2398-3396
Received on 28th February 2018
Revised 10th August 2018
Accepted on 17th October 2018
doi: 10.1049/iet-cps.2018.5055
www.ietdl.org

Maurizio Mongelli[1] ✉, *Enrico Ferrari*[2], *Marco Muselli*[1], *Alessandro Fermi*[2]

[1]*Institute of Electronics, Computer and Telecommunication Engineering, National Research Council of Italy, Genova, Italy*
[2]*Impara S.r.l., Genova, Italy*
✉ *E-mail: maurizio.mongelli@ieiit.cnr.it*

Abstract: The study deals with intelligible analytics for performance modelling of vehicle platooning. Knowledge extraction with rules is targeted to understand safety regions (collision avoidance) of system parameters. Results are shown by feeding data through simulation to the train of different rule extraction mechanisms. Safety regions are evidenced on test data with statistical error very close to zero.

1 Introduction

Safety in *cyber-physical systems* (CPS) [1] means preventing all the conditions that may lead to dangerous trajectories of the system. The approach of machine learning (ML) consists of extracting a model from samples of data that map system parameters into safe and unsafe predictions. The model drives a forecast of unsafe conditions as it states if a specific setting of parameters would bring into danger in the near future. ML is simply based on available data; so no specific statistical assumptions are made on the system. This is a great advantage as finding closed-form formulas of control performance over communication channels in CPS constitutes a formidable problem (see, e.g. [2, 3]).

1.1 Vehicle platooning (VP)

VP is taken as a reference here as being representative of one of the most challenging CPS of the automotive sector [4]. The main goal in VP is finding the best trade-off between performance (i.e. maximising speed and minimising vehicles reciprocal distance) and safety (i.e. avoiding collision) [5]. Most of the literature on this topic focuses on advanced control schemes while abstracting the communication medium. Delay of communication is typically considered as fixed or described through probabilistic models. This allows the analytical derivation of stability models under some hypotheses of the dynamical system [6], but it may be unreliable under realistic conditions. Two branches are evident from the literature in this respect: the derivation of simple models of the delay bound that guarantees safety (see, e.g. Section IV.C of [7]) and extensive simulation with visualisation of safety regions under subsets of parameters when addressing realistic communication [8, 9] and realistic vehicles [10].

1.2 Contribution

A step forward in these two branches relies on (this paper is an extended version of [11])]:

1. A synthetic representation of the system, still joining all the parameters involved.
2. Intelligible analytics to drive knowledge extraction from data through interaction with the user (e.g. what-if analysis [12]).
3. Identification of the largest set of working conditions still preserving collision avoidance.
4. Sensitivity analysis of collision avoidance.

1.3 Intelligibility

Intelligible models are topical in this perspective, as they drive situational awareness for human operators [13]. Intelligibility means that the model is easily understandable, e.g. when it is expressed by Boolean rules. Decision trees (DTs) are typically used towards this aim. The comprehension of neural network models (and of the largest part of the other ML techniques) reveals to be a hard task (see, e.g. Section 4 of [14]).

Together with DT, we use logic learning machine (LLM), which may show more versatility in rule generation and classification precision. The application of other black-box ML approaches to problem 3, such as through the neural network or support vector machine, may be a hard task, due to the complex interrogation required on the trained classifier (see, e.g. [15, 16]). The recent approach of [17] is interesting in this perspective as well, being based on a learning phase in which the system under test is validated with respect to performance constraints. Our approach is different as it relies on triggering validation through rules inference from data and their calibration around safety regions of the system.

2 System under test

2.1 Metrics

The following scenario is considered. Given the platoon at a steady state of speed and reciprocal distance of the vehicles, a braking is applied by the leader of the platoon [7, 18]. The behaviour of the dynamical system is investigated with respect to the following metrics. *Safety* is referred to a collision between adjacent vehicles [in the study, it is actually registered when the reciprocal distance between vehicles achieves a lower bound (e.g. 2 m)]. For both safety and driving comfort, *string stability* (SS) is also important. It means that speed and acceleration fluctuations should be attenuated downstream the string of vehicles. This is studied in particular in the presence of disturbances on the leader movements. In the literature [18, 19] are two good examples to understand how setting the control parameters to achieve stability may be not a straightforward process.

2.2 Dynamics

The following dynamics are considered in the study.

2.2.1 [18, 19]: The *cooperative adaptive cruise control* (CACC) is considered, having the following time dynamics. Let $\chi_i = x_{i-1} - l_{i-1} + g_i^{\text{des}}$ be the desired ith vehicle position (x_{i-1} and l_{i-1} being the position and length of the preceding vehicle,

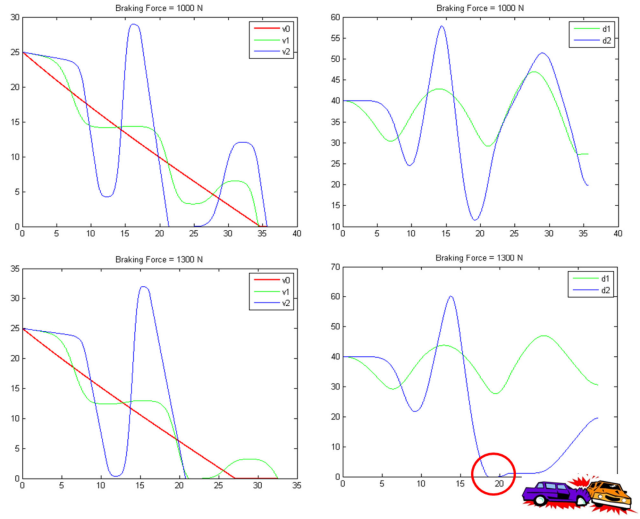

Fig. 1 *Example: three cars, change of braking force*

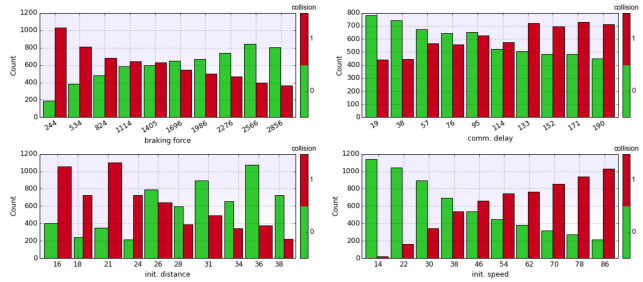

Fig. 2 *Example: histograms of parameters*

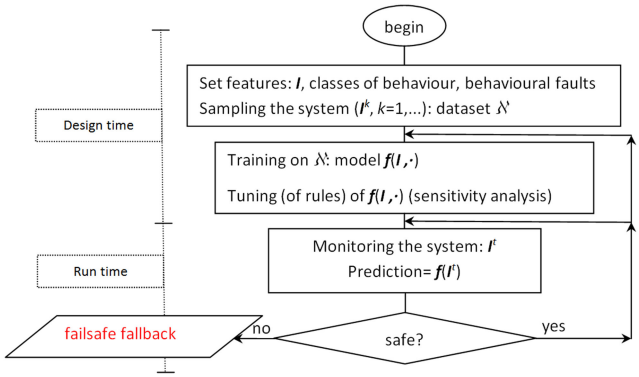

Fig. 3 *Diagram of the methodology*

respectively, and g_i^{des} the intervehicle desired gap), the desired acceleration is tuned according to

$$\ddot{\chi}_i = (1-C_1)\ddot{x}_{i-1} + C_1\ddot{x}_0 - (2\xi - C_1(\xi+\sqrt{\xi^2-1}))\omega_n\dot{\epsilon}_i \\ -(\xi+\sqrt{\xi^2-1})\omega_n C_1(v_i - v_0) - \omega_n^2, \quad (1)$$

where C_1, ω_n and ξ are gain, damping ratio, and bandwidth of the controller, v_i is the speed of vehicle i and $\epsilon_i = d_i + l_{i-1} + g_i^{\text{des}}$. Stability is guaranteed under $\xi \geq 1$ and $C_1 < 1$ [19, 10] (Section IV.E) outlines how CACC may become unstable under 60% of packet losses.

2.2.2 [7]: We also consider the following system dynamics:

$$\dot{v}_i = \frac{1}{m_i}(F_i - (\alpha_i + b_i \cdot v_i^2)); \quad \dot{d}_i = v_{i-1} - v_i, \quad (2)$$

where α_i is the tyre/road rolling resistance, b_i is the aerodynamic drag, $i = 0,\ldots,N-1$, and control law $F_i = \varphi(d_i)$, $i \geq 1$:

$$\varphi(d) = \max\{k \cdot [50(d-27) + (d-27)^3, -10^4]\} \quad (3)$$

The nonlinearities of both dynamics (2) and control law (3) make the analytical derivation of SS conditions intractable.

2.3 Information vector

The behaviour of the dynamical system is synthesised by the following vector of features:

$$\boldsymbol{I} = [N, \iota(0), F_0, \boldsymbol{m}, \boldsymbol{q}, \boldsymbol{p}]. \quad (4)$$

$N+1$ being the number of vehicles in the platoon [subscript $i=0$ defines the index of the leader], $\iota = [\boldsymbol{d}, \boldsymbol{v}, \boldsymbol{a}]$ are the vectors of reciprocal distance, speed, and acceleration of the vehicles, respectively [$\iota(0)$ denotes that the quantities sampled at time $t=0$, after which a braking force is applied by the leader [7]. Simulations are set in order to manage possible transient periods and achieve a steady state of ι before applying the braking.], $\boldsymbol{m}$ are the vectors of weights of the vehicles, F_0 is the braking force applied by the leader, $\boldsymbol{q}$ is the vector of quality measures of the communication medium, fixed *delay* and *packet error rate* (*PER*) are considered in the study, $\boldsymbol{p}$ is the vector of tuning parameters of the control scheme.

2.4 Example

An example may help understand [complete presentation available in [20]]. The system of Section 2.2.2 is considered. Fig. 1 shows speed (left) and reciprocal distance (right) of three vehicles when braking force changes from 1000 N (no collision) to 1300 N (collision). Fig. 2 shows the histograms of the parameters with respect to the two classes; looking at every single variable independently does not let understand conditions of the collision. Scatter plots in two dimensions do not help as well. This means a complex relation exists in the separation of the two classes and that is where ML comes into play to derive a model of the system (e.g. is a collision going to happen?), based on selected features (e.g. speed and distance of the vehicles). The model studied here is based on rules (e.g. speed v below a given threshold ($<v^*$), distance d above a threshold ($>d^*$)).

2.5 Behaviour prediction

The problem consists of predicting whether the dynamics may achieve undesired behaviour of the system in the near future after braking. The study relies on constraints that restrict the runtime behaviour of the platoon, in order to anticipate at design time the circumstances that can occur at runtime [4].

Fig. 3 summarises the basic steps of the approach. At design time, the features of interest, $\boldsymbol{I}(\cdot)$, are selected to stimulate the classes of the behaviour of the system. Safety analysis introduces potential malfunctions (behavioural faults) to be taken into account as well (an example is reported in the performance evaluation). Training is applied to derive the prediction model $f(\boldsymbol{I}(\cdot), \cdot)$ (next Section 3), which is interrogated at run time to infer undesired trajectories. An automatic method for the restriction of the behaviour at run time is thus derived to avoid that, e.g. a braking force with specific speeds and distances will lead to a collision. Run time application means monitoring the information vector (e.g. current speed and distance of vehicles); $f(\boldsymbol{I}(\cdot), \cdot)$ becomes an attractor of desired behaviours, namely, if current speed and distance are classified for collision, proper actions are taken to prevent danger (e.g. emergency braking).

Before deployment at run time, the model may be further interrogated to extract further knowledge on the system (e.g. sensitivity analysis). As ML is simply based on the available data, simulations or historical data collected on the real system may be both exploited for building the dataset for training. New monitoring data may be even exploited in a new design phase to build a new model. The design phase is not only important for the specification of features, classes and behavioural faults, but also for the sampling procedure, which should collect all the classes of

interest. Monitoring during normal operations may lead to the collection of desired trajectories only. In both cases, ML may operate predictions, but the former is more efficient due to the superior knowledge registered in the dataset. The latter deals with one class supervised learning which may introduce arbitrary operations in the definition of anomalies.

3 Problem

The prediction function $f(\boldsymbol{I}(\cdot), \cdot)$ is now investigated by posing a supervised learning problem. Let $\omega = 0$ or 1 denote the under and over threshold events related to circumstances of interest, such as under the metrics defined in Section 2.1. Let $\aleph = \{(\boldsymbol{I}^{\kappa}, \omega^{\kappa}), \kappa = 1, \ldots, \beth\}$ be a dataset corresponding to the collection of events representing the platoon evolution (ω) under different system settings ($\boldsymbol{I}(\cdot)$). The collection of points in $\aleph$ is derived by iteration over the available system under test (platooning simulators in this case).

The classification problem consists of finding the best boundary function $f(\boldsymbol{I}(\cdot), \cdot)$ separating the $\boldsymbol{I}^{\kappa}$ points in $\aleph$, according to the two classes $\omega = 0$ or $\omega = 1$. Several algorithms are available to find $f(\boldsymbol{I}(\cdot), \cdot)$ according to the supervised learning literature [21]. If the feature vector lies in (or may be projected into) a bi-dimensional (or three-dimensional) space, e.g. through principal component analysis [22], $\aleph$ may be easily visualised to give a first insight into the difficulty of the problem (i.e. how much the available samples of the feature vector are separable with respect to the two classes). The projection of $\boldsymbol{I}(\cdot)$ in dual spaces [23] helps to tackle with the nonlinearity of $f(\boldsymbol{I}(\cdot), \cdot)$ on the original space [24]. However, any complexity of the mathematical expression of $f(\boldsymbol{I}(\cdot), \cdot)$ or of any pre-processing on $\boldsymbol{I}(\cdot)$ leads to the so-called *black-box* models, which may be hardly understood to extract further knowledge on the system. Here, we concentrate the attention on *white-box* forms of $f(\boldsymbol{I}(\cdot), \cdot)$ to derive further knowledge on the system.

4 Methodology

4.1 Logic learning machine

The derivation of $f(\boldsymbol{I}(\cdot), \cdot)$ is made by DT and LLM [the analysis was performed through the Rulex software suite, developed and distributed by Rulex Inc. (http://www.rulex.ai/)]. They are both based on a set of intelligible rules of the type **if** (*premise*) **then** (*consequence*), where (*premise*) is a logical product (AND, $\wedge$) of conditions and (*consequence*) provides a class assignment for the output. In the present study, the two classes correspond to the presence or the absence of anomalous patterns. LLM rules are obtained through a three-step process. In the first phase (*discretisation* and *latticisation*) each variable is transformed into a string of binary data in a proper Boolean lattice, using the inverse only-one code binarisation. All strings are eventually concatenated in one unique large string per each sample. In the second phase (*shadow clustering*) a set of binary values, called *implicants*, are generated, which allow the identification of groups of points associated with a specific class. [An implicant is defined as a binary string in a Boolean lattice that uniquely determines a group of points associated with a given class. It is straightforward to derive from an implicant an intelligible rule having in its premise a logical product of threshold conditions based on cut-offs obtained during the discretisation step. The optimal placement of these cut-offs is, therefore, an important phase to extract the highest information gain before clustering [25].] During the third phase (*rule generation*) all implicants are transformed into a collection of simple conditions and eventually combined in a set of intelligible rules. The interested reader on shadow clustering and algorithms for efficient rule generation is referred to [26] and references therein.

4.2 LLM versus DT

When building the model, DT adopts a *divide and conquer* approach that subsequently adds conditions (nodes) to the tree based on smaller and smaller subsets of training data. While this method can lower the overall computational time, it also has the side effect of reducing the information available when selecting conditions after the first one. As a consequence, the rules in the resulting models are disjoint, i.e. there is a strong dependency among them. For this reason, DT may experience over sensitivity to highly relevant attributes, may depend too much on the training set and may be corrupted by irrelevant attributes and noise (see, e.g. Chapter 9 of [27]). In contrast, in LLM, all the implicants are generated through shadow clustering by looking at the whole training set; in this way, resulting rules can overlap and represent different relevant aspects of the underlying phenomenon. The higher precision of LLM derives from the mapping of data on implicants, which reveals to be a smoother operation than the hard separation provided by DT while going deeply in the tree construction.

4.3 Feature ranking

Having a set of n rules $r_1, \ldots, r_n$, rule r_i has *covering* $C(r_i)$, i.e. how many points are covered by r_i (typically expressed for each class separately), and *error* $E(r_i)$, i.e. how many points are misclassified by that rule. Rule r_i has m_i conditions $c_{i1}, \ldots, c_{m_i1}$. Rule r_i has output $\hat{y}_i$. The weight of each condition c_{ij}, denoted by w_{ij} is given by the increase of error when removing that condition. The same principle is applied for feature ranking and rule scoring.

To obtain a measure of the importance of the features included in the rules and rank these features according to this value, we utilised a measure called relevance $R(c)$ of a condition c. Consider the rule r' obtained by removing that condition from r. Since the premise part of r' is less stringent, we obtain that $E(r') \geq E(r)$ so that the quantity $R(c) = (E(r') - (r))C(r)$ can be used as a measure of relevance for c. Let $\boldsymbol{x}$ be an input vector, the relevance for the jth feature component is

$$R_v(x_j) = 1 - \prod_k (1 - R(c_{kh})) \tag{5}$$

the product being computed on the rules r_k that include a condition c_{kh} on the variable x_j. Feature ranking consists of ordering the features with respect to that measure. It is typically more robust than statistical tests (e.g. through entropy, Pearson, Spearman or Kendall tau metrics) because it derives from a trained model and less computationally expensive than ranking inside the training process itself (as with support vector machine [28]). Feature ranking may help knowledge extraction when the user tries to interact with the rules (which features should be chosen for further tuning).

4.4 Safety regions

The availability of rules allows to study the sensitivity to the problem (Section 5.1 of the results), as well as calibrating false negatives through information provided by feature ranking (Section 5.2). However, this may be not sufficient to discriminate between the two classes with (statistical) zero error. The final goal of the validation process is to identify the largest subset of parameters with no false negatives (i.e. prediction of no collision, but a collision in reality). To do this, three methods are studied here.

(1) First, data visualisation of the 2 (or 3) most important features after ranking (5) may be applied to manually inspect the most relevant regions for safety.
(2) Secondly, the LLM is trained with zero error when developing rules. It is able to identify safe points with a safe margin because the cut-offs of the discretisation step are assigned by taking as a reference a boundary placed in the middle among the classes. The rules for the safe class with the largest covering are then joined together under logical OR ($\vee$) to build the predictor. This approach builds a strict boundary around safe points that could be even too conservative, especially if only the first rule with the highest

Data set $\aleph$ divided into κ portions $\aleph_\kappa$.
$\aleph_{1\kappa}$ includes only unsafe points from $\aleph_\kappa$

repeat κ times
  repeat
    1. Train($\aleph_\kappa + \aleph_{1\kappa}$)
    2. $R_{0\kappa}$ subset of 'safe' rules
    3. Manual inspection $R_{0\kappa} \rightarrow R'_{0\kappa}$
    4. Apply($R'_{0\kappa}, \aleph_\kappa$)
  until $\aleph_\kappa$ contains safe points only

Choose most stringent conditions from $R'_{0\kappa}$.

Fig. 4 *Procedure for safety rule extraction*

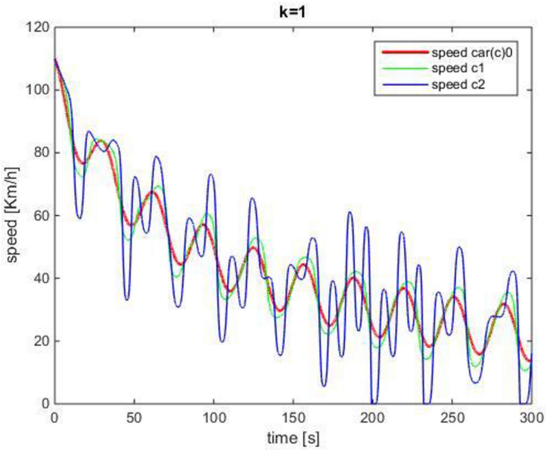

Fig. 5 *Sinusoidal decrease of leader.* $d(0) = 40\,m$; $v(0) = 110 km/h$. *Vehicles speed.* $k = 1$. *Large oscillations:* $J = 2.744$

covering is used, but with the necessary guarantee to limit false negatives.

(3) A more refined procedure is elaborated as well. Since trained classifiers with zero error have the risk to polarise the model on training data, thus precluding the capability for generalisation to new (test) datasets, we still train the classifiers while keeping a non-zero margin for error (e.g. 5%), but we progressively extract unsafe points from the original dataset until only safe points are obtained. The mechanism is shown in Fig. 4. The principle of κ-fold cross-validation has been taken as a reference. [It consists of subdividing the total dataset into κ parts of equal number and, at each step, the κth part of the dataset comes to be the validation dataset, while the remaining part constitutes the training dataset.] In the beginning, $\aleph_{1\kappa}$ is already included in $\aleph_\kappa$ at step 1. However, the presence of duplicated data is ignored by the training algorithm. Step 2 presents rules in the $R_{0\kappa}$ subset, ordered by covering. The subsequent step selects the first rules having a difference in covering between 5% and compares them as follows. If they express identical features, the most stringent thresholds are selected for the feature (e.g. if two rules state $v(0) < 30 \wedge d(0) > 10$ and $v(0) < 29 \wedge d(0) > 15$, respectively, the output becomes $v(0) < 29 \wedge d(0) > 15$). This elaboration gives in output the $R'_{0\kappa}$ subset and may be easily done by manual inspection on $R_{0\kappa}$. The procedure is repeated because the training error precludes the extraction of safety points in one shot of steps 1–4. The final step formulates a rule from the κ available $R'_{0\kappa}$. The most stringent condition is taken on the same feature from different rules. If the rules include different features, they are joined together (in logical $\wedge$), e.g. if two rules state $v(0) < 40$ and $d(0) > 25$, the output is $v(0) < 40 \wedge d(0) > 25$.

4.4.1 Sensitivity analysis: The three methods lead to the identification of the safe class by a small set of rules or even a single rule. The sensitivity analysis on the error achieved by violating or by restricting the rules is straightforward. Let δ be the variation applied to a threshold in a condition of a rule, $d(0) > (d^* + \delta)$, the proportion of false negatives $\epsilon(\delta)$ may be computed according to the decrease or increase of δ, thus quantifying the distance of the region from unacceptable situations, such as $\epsilon \geq 1\%$; the smaller the $\partial\epsilon(\delta)/\partial\delta$ is, the more stable is the region. Feature ranking may help in this respect as it suggests the most sensitive features to the problem and to apply sensitivity analysis on their thresholds.

The second approach (just selecting the safety rules with the highest covering) may be easily automated due to its intrinsic simplicity. The third one should be more precise, but not so easy to deploy, in virtue of the manual calibration of rules. It is finally worth noting that the third approach applies to rules, independently of the used classifiers.

5 Performance evaluation

One of the most important issues of intelligible analytics is interaction with the user. Interaction means let the expert understand the rules and (i) modify them or (ii) come up with new rules in order to extract further knowledge. Section 5.2 is an example of (i) and Section 5.1 of (ii). [Code and datasets of performance evaluation available in [20].]

5.1 String stability

A discrete time simulator of (2) has been developed, by also introducing a delay, *del* in [ms] in a hop-by-hop information exchange between the vehicles. This means speed and distance of vehicle i are known at vehicle $i + 1$ with delay. Since equations are updated accordingly, delayed reactions are propagated downstream the platoon. The following setting is applied: $N = 2$ (three cars in the platoon), $\alpha_i = 0$, $b_i = 0.43$, $m_i = 1050$ kg. Let t be the sampling index over discrete time over an observation horizon having duration $T = 300$ s. The braking force applied by the leader contains a disturbance [18]: $F_0 = -500 \cdot \sin(0.2t)$ Newton (N). Delay is considered fixed, for now, as the following assumption holds: time-varying delays are enclosed within a unique upper bound [18]. Parameters are set as follows [from now on, $d(0)$ and $v(0)$ are presented without subscript i as they are assumed equal for all vehicles] (similarly to [7]):

$$\text{del} \in [1, 15] \text{ ms}, \quad d(0) \in [25, 40] \text{ m}, \quad v(0) \in [90, 110] \text{ km/h}.$$

The performance metric to measure SS is

$$J = \sum_t \left| v_2(t) - v_0(t) \right| \tag{6}$$

A family of control laws of the type as in (3) are also introduced with $k \in [1, 20]$. Extensive simulations, not reported here for the sake of synthesis, corroborate that $J < 1$ values are associated with stability. An example is presented in Fig. 5. The dataset $\aleph$ is built by simulating the system under the chosen conditions and setting ω in correspondence of registered stable and unstable behaviours of the platoon: $\omega = \{$'0' if $J < 1$, '1' otherwise $\}$. $\beth = 2000$ samples of system conditions are derived with a simulation time of <10′ over an IntelCore i7 @2.4 GHz. The LLM is then applied to map the variables into the ω output. We want to find the relation among minimum distance, minimum braking force (i.e. minimum k) and maximum delay, still achieving stability.

First insight into the problem. To understand how tricky the problem is, we start from data visualisation. Fig. 6 shows the scatter plots of the $k -$ del and $k - d(0)$ couples, in which complex boundaries among the variables may be intuitively inferred. The proportion of unsafe points with respect to the vertical axis of the right side of Fig. 6 ($d(0)$) suggests that information with $d(0) \leq 28$ should be filtered out, $d(0) = 28$ lying on the border of stable/unstable classes. Univariate histograms of $d(0)$ (not reported here)

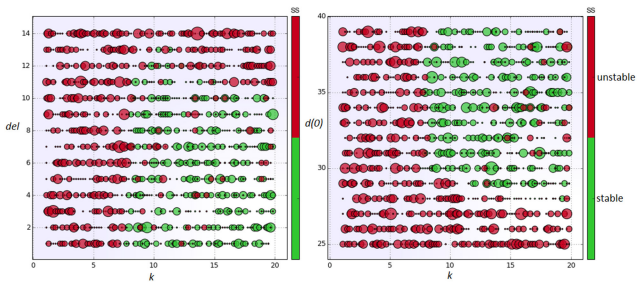

Fig. 6 *Scatter plots of $k - del$ and $k - d(0)$*

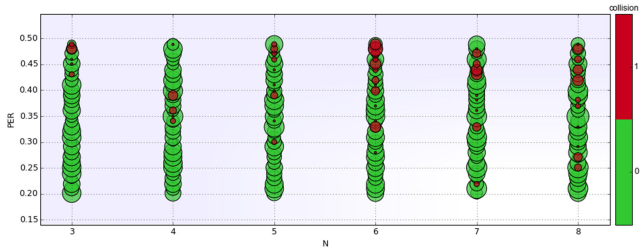

Fig. 7 *Investigated dataset with respect to scattering plot of N-PER*

confirm the analysis. We then apply the LLM over the subset of data corresponding to $d(0) \in [d_{\min}, 40]$; $d_{\min}(0) = 27$ is chosen to validate intuition inferred from the scatter plot. DT is disregarded, for now. The rules obtained are reported below. For the sake of synthesis, only the first 5 rules with the highest covering for the stable class are reported. We recall here that the covering metric, C, measures how many points of a class are synthesised by a rule (e.g. for rule 1: 91% of 'stable' points satisfy the rule; the error, E, measures how many points are misclassified by that rule on the other classes (e.g. for rule 1: 4% of 'unstable' points are misclassified, i.e. assigned to 'stable' class).

One may identify $\text{del} = 10$ as the maximum delay with minimum $k = 9$ as stability condition over the entire range $d(0) \in [30, 40]$. The reasoning is as follows. $k = 9$ being a safe margin over $k = 8.4$ of the last rule, $\text{del} = 10$ being registered by the first three rules with higher covering and as a compromise between eight and 13 of the last two rules, $d(0) \geq 30$ as suggested by the last rule. This was validated over another test set of 2000 points (only stable class has been registered in the test set). This could conclude the analysis if the analyst wants to simplify the procedure. However, a more insightful view of the border of safe and unsafe points may be investigated as well.

Refining the analysis. The situation with $d(0) \in [25, 30]$ is much more complicated: only instability seems to arise over that subset of data. Is there any setting of the other parameters to still achieve stability? The analysis on the new subset: $k \geq 9$, $d(0) \in [25, 30]$, over a new training set with 2000 points, leads to an unbalanced proportion of registered unstable outputs, while a balanced proportion between the two classes appears in the previous analysis. Moreover, the $d(0) \leq 27$ rule has been obtained by the LLM in 69% of the unstable cases. These two facts give a clear indication that the $d(0) \in [25, 30]$ interval should be avoided or restricted even more: $d(0) \in [28, 30]$. By further restricting the analysis on that interval, several rules (similar to the ones already reported) are found with a covering <40%. In virtue of such reduced covering, the most restricting thresholds from the rules are extracted, $\text{del} = 2$, $k = 11$, and successfully validated over a new test set of 2000 points. To summarise, SS conditions are $\text{del} = 10$, $k = 9$ with $d(0) \in [30, 40]$ and $\text{del} = 2$, $k = 11$ with $d(0) \in [28, 30]$, independently to $v(0) \in [90, 110]$.

if $(\text{del} \leq 10) \wedge (d(0) > 28) \wedge (k > 7.845)$ **then stable** $C = 91\%, E = 4\%$;
if $(\text{del} \leq 10) \wedge (d(0) > 27) \wedge (k > 8.145)$ **then stable** $C = 88\%, E = 3\%$;
if $(\text{del} \leq 10) \wedge (d(0) > 28) \wedge (k > 7.575)$ **then stable** $C = 88\%,\ E = 2\%$;
if $(\text{del} \leq 8) \wedge (d(0) > 27) \wedge (k > 7.255)$ **then stable** $C = 73\%, E = 4\%$;
if $(\text{del} \leq 13) \wedge (d(0) > 30) \wedge (k \in (8.415, 17.69])$ **then stable** $C = 63\%, E = 4\%$.

Bounded versus average delay. Safety critical systems require considering exceptional situations. The impact is relevant if the hypothesis above about fixed delay does not hold anymore. To show this, we consider the case when delay becomes an average. Values of delay in the range [1, 15] ms are still considered, but they now constitute the mean of a Gaussian distribution with variance 0.09 [7]. By repeating the analysis [In simulations, a collection of 100 traces are collected to infer whether the platoon is stable or not, for each allocation of the other parameters $(d(0), v(0), k$, whose ranges are left unchanged). Instability is registered if at least one trace collects a sample with $J \geq 1$ (stability otherwise).], the following final SS condition is found: $\text{del} = 4, v(0) < 95, d(0) > 34, k > 11$. The restriction on the behaviour of the system is very relevant from bounded to average delay.

5.2 Safety

The Plexe simulator [10, 18] is used to register $\beth = 15 \times 10^3$ samples of the model presented in Section 2.2.1 [(7)–(12) of [29] are implemented in the simulator as stated in [18].], under the following ranges:

$N \in [3, 8]$, $F_0 \in [-8, -1] \times 10^3$ N [from now on, the notation $(\times 10^3)$ is omitted when referring to thresholds applied to F_0.], PER $\in [0.2, 0.5]$, $d(0) \in [4, 9]$ m, $v(0) \in [10, 90]$ km/h.

The control gain of (1) is set to $C_1 = 0$ [19] (see Section 5.5). A collision is registered when two vehicles are near below 2 m. The performance of the two classifiers is presented with respect to the following metrics: *false negative rate* (FNR), *false positive rate* (FPR), size of safety regions and Fscore. The three methods defined in Section 4.4 are then applied to infer safety regions. The dataset is divided into 3×10^3 points for training (at design time as to Fig. 3) and 12×10^3 for the test (at run time). Rules are derived on the training set; performance evaluation derived from validation of the test set. Such a larger test set stresses the validation process. Feature ranking defines the following decreasing order of importance of the features: PER, $N, v(0), d(0), F_0$. Fig. 7 highlights the complex profile of the two classes; the first two features of feature ranking are used.

By inspecting Fig. 7, it is intuitive to identify the following safety region (first method of Section 4.4): $(N \leq 6) \wedge (\text{PER} < 0.26)$ or by a more accurate inspection:

manual calibration:

if $((N = 6) \wedge (\text{PER} < 0.253))$ **then safe**;
if $((N = 5) \wedge (\text{PER} < 0.258))$ **then safe**;
if $((N = 4) \wedge (\text{PER} < 0.325))$ **then safe**;
if $((N = 3) \wedge (\text{PER} < 0.42))$ **then safe**.

LLM and DT are then tuned according to the second and third method outlined in Section 4.4:

LLM: (see equation below) **DT**:

if $(v(0) \leq 28)$ **then safe** $C = 59\%, E = 0\%$.

The quantities δ_{PER} and δ_N in LLM are related to sensitivity analysis (Section 4.4.1) and are ignored, for now (i.e. they are set to 1). Two iterations of the inner steps of the rule extraction procedure (Fig. 4) are needed to derive the DT rule; at the first iteration, the rule found after the first training $(v(0) \leq 42)$ still leaves 1% of unsafe points. The most stringent condition on $v(0)$ is chosen after $\kappa = 5$ cross validation. On the other hand, no further iterations have been applied to LLM as it identifies only safe points at the subsequent iteration. The original first four LLM rules included F_0, but, being the last ranked feature according to (5), it

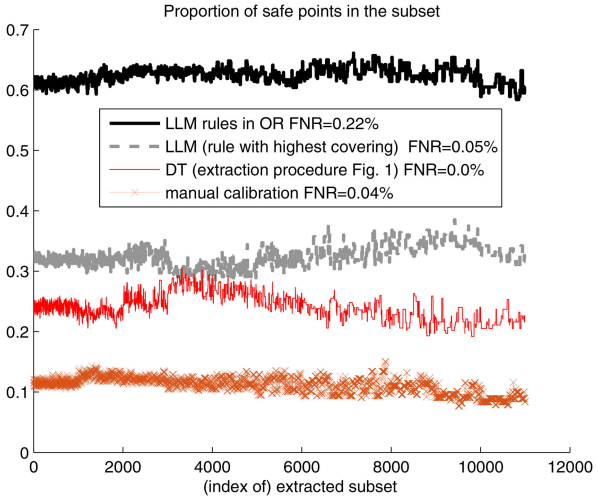

Fig. 8 *Size of safety regions and FNR on the test set ($\delta = 1$)*

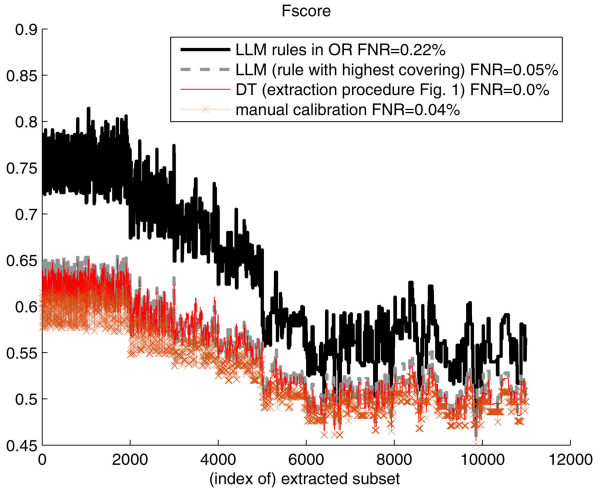

Fig. 9 *F-score on the test set*

was discarded to simplify the model. This introduces a very small increase of the prediction error as expected by the feature ranking.

Fig. 8 presents how many points are safe in different extractions of 10^3 subsets with different sizes from 8% to 50% of the total points available for test (12×10^3); 11×10^3 trials in total. LLM is considered with respect to the five rules in logical OR above ($\vee$) and to the first one with highest covering. The LLM follows the boundary in Fig. 7 better than the other methods, thus extracting a higher number of safe points. FNR is always 0% for DT and very close to 0% for the other techniques. The averages of FNR over the trials are 0.0, 0.05, 0.22 and 0.04% for DT, LLM single rule, LLM 5 rules, and manual calibration, respectively. 99 percentiles are: 0.0, 0.27, 0.73 and 0.22%. Averages of FPR are 7.4, 6.8, 3.8 and 7.8%. The F-score, F, is also considered as performance comparison, defined as

$$F = \frac{2 \cdot \mathrm{TP}}{2 \cdot \mathrm{TP} + \mathrm{FN} + \mathrm{FP}} \tag{7}$$

where TP, FN, and FP are the number of true positive, false negative, and false positive alarms, respectively. Values of F close to 1 denote better discriminant capacity of the classifier. It is shown in Fig. 9. LLM 5 rules achieve the best performance in virtue of the more complex profile of the rules. Decreasing values of F are related to lower sizes of extracted subsets in which less TP are present and FPs become sensibly higher.

5.3 Discussion

The analysis gives a clear indication about the number of safe points in the chosen ranges of the parameters: up to 60% if 0.22% of FNR may be accepted, <30% if one wants to identify 0% of FNR. The gap between the safety regions under 0.22 and 0% of FNR is considerable and denotes the difficulty of the problem. The corresponding DT rule for the latter is simple, $v(0) \leq 28$ km/h, but impractical as it leads to platoons working at low speed. LLM 5 rules include the DT rule (last rule of LLM) but also identifies safe platoons with higher speeds. For example, the first LLM rule selects safe points independently to $v(0) \in [10, 90]$ km/h and the following three rules accept $v(0) > 28$ km/h.

5.4 Sensitivity analysis

The errors of LLM (0.22% and 0.05%) are due to the size of the training set that is significantly smaller than the one of the test set (3×10^3 versus 12×10^3). To further reduce them, one may perform the sensitivity analysis outlined in Section 4.4.1. Feature ranking suggests acting on *PER* and N. The quantities δ_{PER} and δ_N are calibrated on the test set to achieve no error while keeping the largest number of safe points. The optimal setting is $\delta_N = 6/7$ (i.e. one car is eliminated from the platoon to become safe), $\delta_{\mathrm{PER}} = 0.97$ with LLM single rule and $\delta_{\mathrm{PER}} = 0.86$ with LLM 5 rules and is validated on an additional test set of 5×10^3 points. In the new test set, the FNRs of the LLM single rule and five rules are 0.038 and 0.18%, respectively, and the new δ setting leads to zero FNR. Fig. 10 shows the inherent reduction of safe points with respect to Fig. 8. Despite the reduction, LLM 5 rules preserve a more complex profile of safety regions with zero FNR. Under this methodology, one may easily interact with the rules by calibrating error versus a number of safe points (i.e. closest δ to 1 with no error). More complex mathematical functions, e.g. deriving from the neural network, can be hardly applied with similar simplicity.

5.5 Behavioural fault

Combining data driven approaches and physics of failure using ML is another important issue [1, 17]. To this aim, we take [17] as a reference. It injects behavioural faults in the platooning system, by over-simplifying the structure of the control parameters. By following the same idea, we have considered so far the worst allocation of C_1 in (1) of the CACC scheme when it is set to 0. This means only precedent vehicle' information is delivered to each vehicle [19]. $C_1 > 0$ leads to better performance as it deals with the broadcast of leader behaviour to all components of the platoon. This is investigated in detail in [19]. By following the same line of Table VII of [19], we investigate the impact of increasing C_1 from 0 to 0.9, by repeating the same DT and LLM analysis on new simulation data, in which all parameters are left unchanged, except C_1 variable. The resulting rules are not reported for the sake of synthesis, but the following qualitative considerations hold true. The improvement is evident as the conditions thresholds are less strict as soon as C_1 increases (e.g. the DT finds increasing acceptable speeds from 28 to 76 km/h). The rules found with

if $((\mathrm{PER} \leq 0.325 \cdot \delta_{\mathrm{PER}}) \wedge (N \leq 7 \cdot \delta_N) \wedge (d(0) > 4.2385))$ $(C = 29.5\%, E = 0\%)$

$\vee$ (**if** $(d(0) > 4.69) \wedge (v(0) \leq 37))$) $(C = 26\%, E = 0\%)$

$\vee$ (**if** $(\mathrm{PER} \leq 0.445 \cdot \delta_{\mathrm{PER}}) \wedge (v(0) \leq 41))$) $(C = 24.5\%, E = 0\%)$

$\vee$ (**if** $(\mathrm{PER} \leq 0.405 \cdot \delta_{\mathrm{PER}}) \wedge (d(0) > 5.5055) \wedge (v(0) \leq 53))$) $(C = 25.1\%, E = 0\%)$

$\vee$ (**if** $(v(0) \leq 28))$) $(C = 24.3\%, E = 0\%)$

then safe

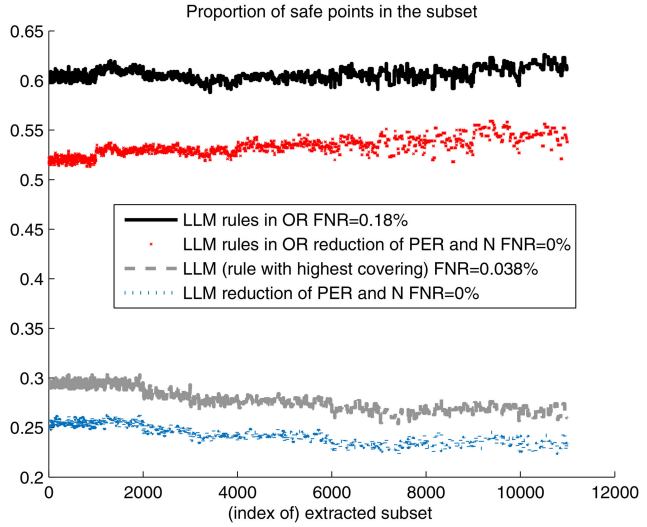

Fig. 10 *Size of safety regions and FNR under more restrictive rules (additional test set) ($\delta < 1$)*

$C_1 = 0$ are thus conservative with respect to the worst condition in the communication exchange.

5.6 802.11p

Additional simulations deal with a realistic scenario as in [18]. The channel model follows Nakagami-3 fading (free-space path loss) with network device IEEE 802.11p with beacon frequency of 10 Hz. All system parameters are the same as above, $C_1 = 0$, except the channel model under *PER*, which is replaced by 802.11p. 15×10^3 samples are generated, 1/4 of which is used for training, the rest for the test as above. Additional human-driven vehicles are present in two additional separate lanes to generate wireless interferences as in [18]. The performance impact is considerable as the real device introduces a *PER* <10% (an average measure of *PER* along simulation traces), thus leading to similar rules with less restrictive thresholds, such as $v(0) \geq 35$ km/h with DT. Qualitatively, the following considerations hold true for LLM. An additional car is accepted in the platoon by the rule with the highest covering and the last rule (previously related to $v(0)$ and equal to the DT one), is now related to $d(0)$ and accepts a larger safe distance over 6 m. This leads to sensibly larger safety regions, whose average coverage is 30.9% with DT, 37.4% with LLM one rule and 66% with LLM 5 rules, with corresponding average FNRs: 0.0, 0.08, and 0.20%, respectively.

6 Conclusion and future work

The study shows how intelligible analytics helps derive synthetic performance modelling of VP. Rules extraction through supervised learning drives the definition of safety regions of system parameters. Results obtained through different simulators and performance metrics corroborate the reliability of the approach.

Future work relies on the theoretical investigation of the profile placed on the border of a predefined threshold of FNs. On-going investigation deals also with simulation scenario extensions, including, but not limited to, more complex behavioural faults, other cases of realistic communication, control parameters setting [6], vehicle actuators impact [10], security [30] and comparison with black-box approaches in ML.

7 Acknowledgments

The work was done in the framework of the project SafeCOP, call H2020-ECSEL-2015-RIA-two-stage, ref. 692529-2.

8 References

[1] Pop, P., Scholle, D., Hansson, H., *et al.*: 'The safecopecsel project: safe cooperating cyber-physical systems using wireless communication'. 2016 Euromicro Conf. on Digital System Design (DSD), Limassol, Cyprus, 2016, pp. 532–538

[2] Xiao, L., Johansson, M., Hindi, H., *et al.*: 'Joint optimization of communication rates and linear systems', *IEEE Trans. Autom. Control*, 2003, **48**, (1), pp. 148–153

[3] Xiao, N., Xie, L., Qiu, L.: 'Feedback stabilization of discrete-time networked systems over fading channels', *IEEE Trans. Autom. Control*, 2012, **57**, (9), pp. 2176–2189

[4] Pop, P., Scholle, D., Sljivo, I., *et al.*: 'Safe cooperating cyber-physical systems using wireless communication', *Microprocess. Microsyst.*, 2017, **53**, pp. 42–50

[5] Jia, D., Lu, K., Wang, J., *et al.*: 'A survey on platoon-based vehicular cyber-physical systems', *IEEE Commun. Surv. Tutor.*, 2016, **18**, (1), pp. 263–284

[6] Oncu, S., van de Wouw, N., Nijmeijer, H.: 'Cooperative adaptive cruise control: tradeoffs between control and network specifications'. 2011 14th Int. IEEE Conf. on Intelligent Transportation Systems (ITSC), Washington, DC, USA, 2011, pp. 2051–2056

[7] Xu, L., Wang, L.Y., Yin, G., *et al.*: 'Communication information structures and contents for enhanced safety of highway vehicle platoons', *IEEE Trans. Veh. Technol.*, 2014, **63**, (9), pp. 4206–4220

[8] Segata, M., Cigno, R.L.: 'Automatic emergency braking: realistic analysis of car dynamics and network performance', *IEEE Trans. Veh. Technol.*, 2013, **62**, (9), pp. 4150–4161

[9] Ge, J.I., Orosz, G.: 'Dynamics of connected vehicle systems with delayed acceleration feedback', *Transp. Res. C, Emerg. Technol.*, 2014, **46**, pp. 46–64. cited By 90

[10] Santini, S., Salvi, A., Valente, A.S., *et al.*: 'A consensus-based approach for platooning with intervehicular communications and its validation in realistic scenarios', *IEEE Trans. Veh. Technol.*, 2017, **66**, (3), pp. 1985–1999

[11] Fermi, A., Mongelli, M., Muselli, M., *et al.*: 'Identification of safety regions in vehicle platooning via machine learning'. 2018 14th IEEE Int. Workshop on Factory Communication Systems (WFCS), Imperia, Italy, 2018, pp. 1–4

[12] Rizzi, S.: 'What if analysis'. Available at http://www-db.deis.unibo.it/~srizzi/PDF/eds-WIA.pdf

[13] Cotroneo, D., Paudice, A., Pecchia, A.: 'Automated root cause identification of security alerts: evaluation in a saas cloud', *Future Gener. Comput. Syst.*, 2016, **56**, pp. 375–387. Available at http://www.sciencedirect.com/science/article/pii/S0167739X15002897

[14] Fisch, D., Hofmann, A., Sick, B.: 'On the versatility of radial basis function neural networks: a case study in the field of intrusion detection', *Inf. Sci.*, 2010, **180**, (12), pp. 2421–2439. Available at http://www.sciencedirect.com/science/article/pii/S0020025510001015

[15] Martens, D., Huysmans, J., Setiono, R., *et al.*: in Diederich, J. (Ed.): '*Rule extraction from support vector machines: an overview of issues and application in credit scoring*' (Springer, Berlin, Heidelberg, 2008), pp. 33–63. Available at https://doi.org/10.1007/978-3-540-75390-2_2

[16] Hailesilassie, T.: 'Rule extraction algorithm for deep neural networks: a review', *Int. J. Comput. Sci. Inf. Secur.*, 2016, **14**, (7), pp. 371–381

[17] Meinke, K.: 'Learning-based testing of cyber-physical systems-of-systems: a platooning study', in: Reinecke, P., Di-Marco, A. (Eds): '*Computer performance engineering*' (Springer International Publishing, Cham, 2017), pp. 135–151

[18] Santini, S., Salvi, A., Valente, A.S., *et al.*: 'A consensus-based approach for platooning with inter-vehicular communications'. 2015 IEEE Conf. on Computer Communications (INFOCOM), Hong Kong, 2015, pp. 1158–1166

[19] Fernandes, P., Nunes, U.: 'Platooning with IVC-enabled autonomous vehicles: strategies to mitigate communication delays, improve safety and traffic flow', *IEEE Trans. Intell. Transp. Syst.*, 2012, **13**, (1), pp. 91–106

[20] Mongelli, M.: 'Git repository of vehicle platooning: simulation and datasets'. Available at https://github.com/mopamopa/Platooning

[21] Theodoridis, S., Koutroumbas, K.: '*Pattern recognition*' (Academic Press, Amsterdam, Netherlands, 2008, 4th edn.)

[22] Cambiaso, E., Aiello, M., Mongelli, M., *et al.*: 'Feature transformation and mutual information for DNS tunneling analysis'. 2016 Eighth Int. Conf. on Ubiquitous and Future Networks (ICUFN), Vienna, Austria, 2016, pp. 957–959

[23] Baudat, G., Anouar, F.: 'Feature vector selection and projection using kernels', *Neurocomputing*, 2003, **55**, (1), pp. 21–38. Support Vector Machines. Available at http://www.sciencedirect.com/science/article/pii/S0925231203004296

[24] Burges, C.J.C.: 'A tutorial on support vector machines for pattern recognition', *Data Min. Knowl. Discov.*, 1998, **2**, (2), pp. 121–167. Available at https://doi.org/10.1023/A:1009715923555

[25] Boros, E., Hammer, P.L., Ibaraki, T., *et al.*: 'An implementation of logical analysis of data', *IEEE Trans. Knowl. Data Eng.*, 2000, **12**, (2), pp. 292–306

[26] Muselli, M., Ferrari, E.: 'Coupling logical analysis of data and shadow clustering for partially defined positive Boolean function reconstruction', *IEEE Trans. Knowl. Data Eng.*, 2011, **23**, (1), pp. 37–50

[27] Maimon, O., Rokach, L.: '*Data mining and knowledge discovery handbook*' (Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005)

[28] Janecek, A., Gansterer, W., Demel, M., *et al.*: 'On the relationship between feature selection and classification accuracy', in Saeys, Y., Liu, H., Inza, I., Wehenkel, L., de Pee, Y.V. (Eds): Proc. Workshop on New Challenges for Feature Selection in Data Mining and Knowledge Discovery at ECML/PKDD 2008. vol. 4 of Proc. Machine Learning Research (PMLR, Antwerp, Belgium, 2008), pp. 90–105. Available at http://proceedings.mlr.press/v4/janecek08a.html

[29] Segata, M., Joerer, S., Bloessl, B., *et al.*: 'Plexe: a platooning extension for veins'. 2014 IEEE Vehicular Networking Conf. (VNC), Paderborn, Germany, 2014, pp. 53–60

[30] Marchesani, S., Pomante, L., Pugliese, M., *et al.*, in 'Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications

Engineering' Zuniga, M., Dini, G. (Eds): '*Definition and development of a topology-based cryptographic scheme for wireless sensor networks*' (Springer International Publishing, Cham, 2013), pp. 47–64. Available at https://doi.org/10.1007/978-3-319-04166-7_4