Mining Sensor Data for Predictive Maintenance in the Automotive Industry

Flavio Giobergia*, Elena Baralis*, Maria Camuglia†, Tania Cerquitelli*, Marco Mellia‡,
Alessandra Neri†, Davide Tricarico† and Alessia Tuninetti†
*Dipartimento di Automatica e Informatica – Politecnico di Torino, Torino, Italy.
Email: {name.surname}@polito.it
†General Motors Global Propulsion Systems – Torino S.r.l., Italy.
Email:{name.surname}@gm.com
‡Dipartimento di Elettronica e Telecomunicazioni – Politecnico di Torino, Torino, Italy.
Email:{name.surname}@polito.it

***Abstract*—Predictive maintenance is an ever-growing area of interest, spanning different fields and approaches. In the automotive industry faulty behaviors of the oxygen sensor are a key challenge to address. This paper presents OXYCLOG, a data-driven framework that, given a large number of time series collected from a vehicle's ECU (engine control unit), builds a model to predict if the oxygen sensor is currently unclogged, almost clogged (since the clogging of the sensor happens gradually), or clogged. OXYCLOG is characterized by a tailored preprocessing, which includes a custom and interpretable feature selection algorithm, along with a summarization strategy to transform a time-dependent problem into a time-independent one. Furthermore, a semi-supervised labeling methodology has been devised to use different data sources with different characteristics to define meaningful clogging labels. OXYCLOG integrates state-of-the-art classification algorithms – both interpretable and non-interpretable – to process real ECU data with good prediction performance.**

***Index Terms*—Data-driven approach; classification algorithms; semi-supervised labeling; faulty behaviors of oxygen sensors.**

I. INTRODUCTION

Before the introduction of the Internet of Things paradigm, data in vehicles was collected and processed locally by the Engine Control Unit for routine operations managing and monitoring vehicle operativity. The limited ECU (Engine Control Unit) and storage capabilities of vehicles make the collection and the analysis of past data infeasible. In the era of connected vehicles, data can be locally collected and sent to a remote storage location for later analyses with better performing tools. Hence, automotive manufacturers can leverage the data collected by their own on-board systems to offer additional value to their customers. This value can be defined in terms of more transparency as well as additional services and features.

One of the additional services that can be offered is predictive maintenance. Predictive maintenance, or prognostics, aims at the identification of possible malfunctions ahead of time, allowing a prompt intervention before the actual failure. Both manufacturers and customers can benefit from this kind of prediction. The former can issue recalls only when actually needed and before irreversible damage occurs, the latter will not experience unexpected vehicle malfunctions. For these reasons, automotive companies are actively interested in predictive maintenance.

Predictive maintenance can be approached in two different ways: either through a *model-based* approach, where the component of interest is modelled to the appropriate level of detail, or through a *data-driven* approach, where data is collected and processed through pattern recognition and machine learning techniques to infer useful insights and build failure predictors.

Previous works based on the model-based approach typically provide interesting high-level insights of the prognostics field, either discussing approaches to address an issue, or introducing problems to tackle. In [1] an overview of the "prognostics" topic as a whole is presented, also discussing possible algorithms for tackling the problems (e.g., Artificial Neural Networks and Genetic Algorithms), along with references to successful case studies for each algorithm. The work in [2], instead, provides the overview of a specific system, the Electric Vehicle (EV) powertrain, and analyzes possible faults that might occur in it (e.g., in rotors, bearings, inverters).

Other related works, both model-based and data-driven, cover predictive maintenance case studies. An important application area is focused on batteries and electric vehicles. For example the authors in [3] propose a data-driven approach to the estimation of the State of Charge in electric vehicles that overcomes some of the limitations of the model-based solution. Other addressed issues concern, for example, engine oil quality [4], injector cylinders misfiring [5] and many others. Only a small fraction of works explicitly states that the predictive maintenance aspect can be carried out on-board. In other cases, simulations of specific components [6], or experiments with miniaturized mock-ups [3] are carried out instead.

General Motors (GM in the rest of the paper) has been a leader in the application of automotive prognostics, which is marketed in the US since 2015 under the name of OnStar Proactive Alerts. The proactive alerts presently cover the vehicle starting system on millions of production vehicles. However, these alerts all follow the model-based (or physics based) approach, whereas this work is focused entirely on a data-driven one. More specifically, this work is focused on

analyzing the behavior of the oxygen sensor, a sensor that measures the level of oxygen in the exhaust gases. This sensor is subject to problems that hinder its capability of working correctly, thus resulting in a significant drop in performance. The analysis focused on the identification of early symptoms in the degradation of this sensor.

In this paper, we describe OXYCLOG a data-driven framework that, given a large number of time series collected from a vehicle's ECU, builds a model to predict if the sensor is currently unclogged, almost clogged (since the clogging of the sensor happens gradually), or clogged.

This research work provides the following contributions.

- *Tailored preprocessing.* It introduces an ad-hoc approach to preprocessing by (i) defining a custom and interpretable feature selection algorithm and (ii) transforming a time-dependent problem into a time-independent one.
- *Semi-supervised labeling methodology.* Starting from an unlabeled situation, an approach is devised to use different data sources with diverse characteristics (e.g. sampling frequencies, monitored timespans) to define meaningful labels.
- *Real-world data analysis application.* It exploits state-of-the-art classification algorithms – both interpretable and non-interpretable – to process real ECU data with satisfactory results.

The paper is organized as follows. Section II introduces the initial problem and case definition. Section III provides a high level description of the OXYCLOG framework, while the subsequent sections address the most important building blocks of the analytical process. More specifically, Section IV describes data preprocessing, Section V introduces our labeling technique, and Section VI describes the modeling and prediction activities. Experimental results are presented in Section VII, while Section VIII discusses the lessons learned throughout the case study. Finally Section IX draws conclusions and outlines future work. and

II. CASE STUDY

The oxygen sensor is a device used to measure the proportion of oxygen in the exhaust gas of an internal combustion engine. This information is used, in diesel engines, to lower the amount of pollutants in the exhaust gases and to monitor the performance of the injectors and of the fueling system.

This oxygen sensor is subject to clogging due to the cumulation of soot contained in the exhaust gas the sensor is constantly exposed to. The clogging of this sensor results in slower oxygen measurements: this implies a suboptimal behavior in terms of efficiency, given the role of the oxygen sensor in determining the state of the fueling system. Additionally, given the sensor's relevance for the catalytic converter, slower readings result in more harmful emissions being released in the environment. The goal of this study is to predict when the oxygen sensor is about to get clogged starting from the data collected from a vehicle's ECU (Engine Control Unit). To this aim a set of *cycles* has been analyzed. Each cycle is a recording of the readings of the on-board sensors for a period of approximately 1 hour (3750 seconds). These cycles are recorded in a controlled environment (i.e. a test bench). Each cycle is piloted by a predefined track imposed on the gas pedal. The track used is the same for all cycles. All cycles are collected from a single engine, and they have been recorded using two different software programs, Program A and Program B. These two tools have different characteristics and have been used for similar – yet complementary – purposes. The main differences between the two are reported in Table I. The two programs worked in parallel, collecting data on the same exact cycles, but while Program A covered their entirety, Program B only monitored the final 5-minute period. This final part, as explained in more detail in Section V, will be used to understand whether a cycle should be considered as clogged.

	Program A	**Program B**
Duration (s)	3750	300
Sampling frequency (Hz)	1	320
Number of variables[1]	50	440
Number of cycles	400	388

TABLE I: Differences between Program A and Program B

Program B monitors a number of variables that is larger than Program A (see Table I). The variables collected can be broadly divided into two categories: those collected by the ECU, and those collected by the test bench tools. The former are signals collected by sensors actually accessible by the ECU, the latter are variables measured externally in the controlled environment, that is, the test bench.

Since this study aims at scaling to on-the-road scenarios, the test bench variables have been set aside to better represent the data available on board of the average vehicle. These variables have been handled with a domain-agnostic approach, thus avoiding any kind of context-aware bias. General Motors engineers, though, helped validate or reject results after the initial "blind" analysis.

Finally, although Program A and Program B theoretically recorded the same exact cycles, the actual number of available cycles differs between the two (400 and 388). Possible malfunctions of the recording tools might have resulted in some of the cycles being lost or not stored at all (although this is little more than speculation, having the datasets been recorded by people not involved in this project).

To easily compare the spans of time monitored by the two programs, Figure 1 shows the accelerator signal used as the track for the cycle, as recorded by Program A and Program B. This final part terminates with the so-called *cut-off*, a maneuver consisting in the sudden release of the gas pedal. This cut-off is of particular interest as it will be used to understand whether the cycles are clogged (i.e. the sensor is clogged) or not.

[1]This is the most common number of variables recorded by each tool. Some cycles contained a different number variables, depending on decisions and events that occurred at recording time

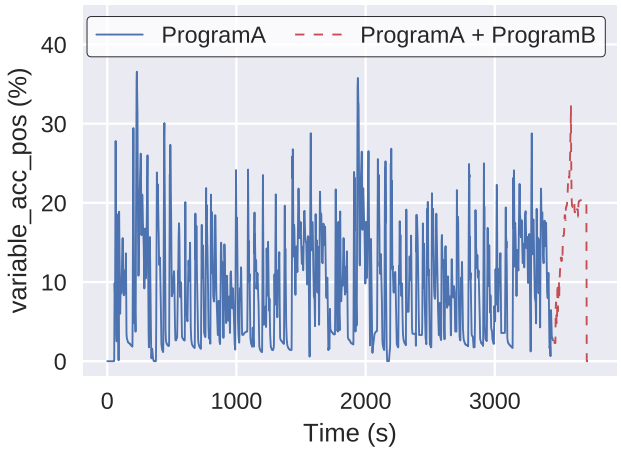


Fig. 1: Acceleration pedal signal, recorded with both Program A and Program B. The final portion of the graph contains the overlapping data from the two programs

III. The OxyClog engine

The early prediction of the oxygen sensor clogging in diesel engines is the research challenge addressed by OxyClog (Oxygen sensor Clogging prediction). To this aim, OxyClog first analyzes a set of engine cycles to model the clogging status of the oxygen sensor and then it exploits the derived model to predict whether the oxygen sensor will be unclogged, almost clogged (since the clogging of the sensor occurs gradually) or clogged at the end of a new cycle whose sensor status is unknown. OxyClog performs the complete pipeline of the data-driven analytics process from data preprocessing to knowledge exploitation. Furthermore, OxyClog includes a semi-supervised methodology to assign a class label to each cycle.

Specifically, OxyClog performs the following four main activities.

Cycle pre-processing. OxyClog analyzes a set of engine cycles. Each one can be defined in terms of a set of signals, each one representing a variable monitored using on-board sensors (the ones monitored by the test bench are discarded a priori). Given the large number of variables monitored for each cycle, OxyClog includes a strategy to analyze redundancy among features with the ultimate goal of removing those features that are strongly correlated to other ones and that, as such, introduce redundancy. Furthermore, OxyClog includes an ad-hoc strategy to represent the monitored signals (time-dependent) in a compact form (time-independent) that enables the application of state-of-the-art classifiers for the learning process.

Cycle labeling. OxyClog includes a semi-supervised methodology to automatically define, for each cycle, the clogging status of the oxygen sensor based on the analysis of the measured response time.

Cycle modelling. OxyClog integrates three state-of-the-art classification algorithms (i.e., decision trees [7], SVM [8] and neural networks [9]) capable of learning which of the input variables are causing the problem at hand (i.e. decision tree) and to build accurate models in complex scenarios (e.g., neural networks and SVM)

Cycle prediction. Given a new, unlabeled cycle represented by the original large set of variables, OxyClog exploits the previously learned model to predict the status of the given sensor.

IV. Cycle pre-processing

This component addresses the two main tasks required to prepare the data of a given cycle for the subsequent analytics steps: (i) *feature selection* and (ii) *data transformation*. First, in order to reduce the number of variables collected by Program A, OxyClog analyzes any existing redundancies to remove them through a feature selection process. Then, given the large number of possible inputs (i.e. $n_{variables} \cdot n_{samples}$, with $n_{samples} = 3750$) compared to the low number of cycles (≈ 390), OxyClog includes an ad-hoc data transformation to allow state-of-the-art classifiers to make meaningful predictions. These steps are described in more detail in the following subsections.

A. Feature selection

The feature selection block in OxyClog aims at reducing data redundancy from the cycles collected by Program A. Some of the advantages that come with the reduction of the number of variables are:

- Better collection of data on the field: considering the long-term applications of this process, collecting a lower number of signals results in a reduction in terms of both costs required for the sensors used and bandwidth needed for the transmission of the signals to a centralized server.
- More concise representation of each cycle: this makes the entire problem easier for the classifiers to handle, particularly in light of the limited number of cycles available.
- Easier understandability of the classifier: some of the classification models tested are interpretable. If a lower number of variables is provided, the generated output will consequently be more concise and understandable.

With the support of GM experts some Program A variables can be discarded a priori. These variables are:

- Test bench variables that will not be normally available in vehicles;
- Discrete variables (including constant ones) that are hardly comparable to continuous ones by means of a correlation coefficient (which will be used to identify similarities among signals): these variables can be analyzed, as needed, at a later stage;
- Categories of variables that have been deemed unrelated by GM experts;
- "Unaligned" variables, i.e. variables that have only been recorded for some cycles and that would therefore not be always available.

This reduces the number of variables from the original 50 down to 31.

To deal with data redundancy different methods have been proposed in literature. However, some of the common approaches used for dimensionality reduction that consist in combining available features to produce new ones (e.g. the Principal Component Analysis [10]) do not yield interpretable results. Given the strong collaboration with the domain experts, those solutions have been set aside and a different one that maintains the original variables has been chosen instead. To discover redundant features OXYCLOG analyses the correlation among signals. Given any two signals, the Pearson correlation coefficient between the two sequences has been taken as an indication of how related the two variables are. For each pair of variables (X,Y), OXYCLOG computes the correlation coefficient through the Pearson correlation defined as $\rho_{X,Y} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y}$, where $\mathrm{cov}(X,Y)$ is the covariance between X and Y, σ_X is the standard deviation of X and analogously σ_Y for Y. Figure 2 shows the correlation matrix plotted as a heatmap of the correlation coefficients computed for each pair of variables for one of the available cycles. At a glance, the plot shows that actual redundancy exists within the variables.

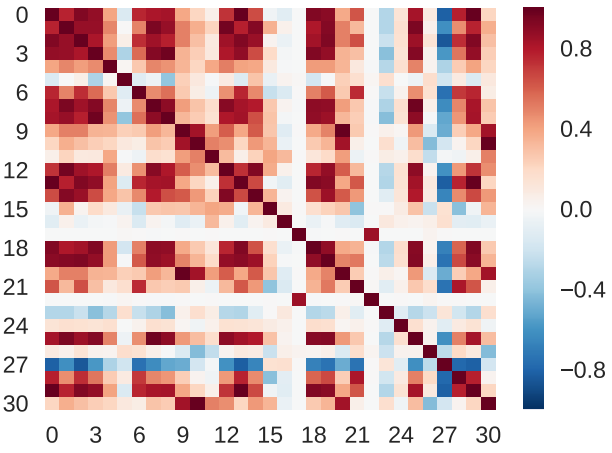


Fig. 2: Heatmap for the correlation matrix for a given cycle. The variable names have been replaced with numbers for visualization's sake

To properly handle the data redundancy and iteratively extract the most representative variables from the initial pool of available variables, OXYCLOG introduces an ad-hoc algorithm named CORR-FS (CORRELATION-BASED FEATURE SELECTION). CORR-FS identifies a list of independent variables (where "independent" can be defined in terms of correlation) ordered by descending degree of representativeness of the other signals. Specifically, The CORR-FS algorithm requires a single parameter r_{min} to be selected. This parameter represents the minimum correlation coefficient below which two signals/variables are considered as not strongly correlated: the selected features will all be correlated with a coefficient smaller than r_{min}. Redundant features that are dropped by the algorithm, on the other hand, are correlated with one of the "representative" features with a coefficient larger than or equal to r_{min}. The value for the r_{min} parameter has been defined empirically (see Section VII-A2 for more details). CORR-FS performs the following steps:

1) For each cycle k, the correlation coefficient between each pair of variables i and j is computed and stored as $r_{k,ij} = r_{k,ji}$.
2) For each pair of variables i and j, the overall correlation coefficient $r_{ij} = r_{ji}$ is computed as the average correlation coefficient for that pair of variables over all the cycles: $r_{ij} = \frac{1}{n} \sum_{k=1}^{n} r_{k,ij}$
3) The list of "remaining variables" L is initialized with all the variables available
4) For each variable i in L, the sum of squared correlation coefficients s_i is computed: $s_i = \sum_{j \in L} r_{ij}^2$
5) The variable
$$b = \underset{i \in L}{\mathrm{argmax}}\, s_i$$
is extracted from L as the most representative of the variables left
6) All variables $v \in L$ such that $r_{vb} \geqslant r_{min}$ are extracted from L in that they are well represented by b
7) If L is empty, the algorithm terminates, otherwise it continues with Step 4

For this case study, the CORR-FS algorithm narrowed the number of Program A variables from 31 down to 14.

B. Data transformation

Since Program A collects approximately 3750 samples for each variable in a given cycle, the number of potential inputs to be managed by the classifier is very significant. While this number of features is not, by itself, unapproachable, it is when backed by a limited number of instances (i.e. cycles). Considering each sample of each signal as a feature, a total of $3750 \cdot 14 \approx 50,000$ features could be identified. Even the most generous of estimates [11] for the adequate features/instances ratio is far from considering 388 instances as enough to handle 50,000 features. Given the high frequency of the signals, sampling the dataset down to a manageable number is not feasible: getting the total number of features down to, for example, 1,000 would require a sampling period of 50 seconds, an unacceptable value.

OXYCLOG integrates an alternative solution to sampling based on summaries of each signal using a limited number of statistics: the number of statistics used is independent of the number of samples, thus allowing to decide how many are to be used. The following considerations regard the summary statistics that have been used. Specifically, OXYCLOG compactly represents the signal for each variable as follows.

The *mean and the standard deviation* for each of the sampled signals. These two values allow identifying behaviors where the signal is offset by a positive or negative amount

(mean) and the entity of the oscillations around the mean, summarizing the entity of the amplitude of the signals (standard deviation). The mean and the standard deviation are enough to describe a Gaussian distribution, but this assumption of normality does not hold for the distribution of samples for the signals.

The percentiles (i.e. the value below which a given percentage of samples fall) from the 10^{th} to the 90^{th}, in increments of 10, have been used and yielded satisfactory results.

For each of the 14 variables, 11 statistics have been used (9 percentiles, mean, standard deviation), for a total of $14 \cdot 11 = 154$ features. Since the above statistics summarize the distribution of values, not the way those values evolve over time, OXYCLOG also represents each signal with *the distribution of the derivatives' values* to partially model the time component. Slower signals will have a distribution of derivatives shifted towards lower absolute values and vice versa. The "derivative" of a signal is computed as the difference between subsequent samples over the time delta elapsed between the two: as such, a more accurate definition would be difference quotient. For practical reasons, the terms will be used interchangeably throughout the text. The computation of percentiles, mean and standard deviation "derivatives" of each signal doubles the overall number of input features (from 154 to 308), but still keeping it down to an acceptable number.

V. CYCLES LABELING

OXYCLOG needs to approach a dataset comprised of data collected from different sources with different characteristics (e.g. sampling frequencies, monitored timespans): starting from an unlabeled situation, these sources are used to define meaningful labels – thanks to a SEMI-SUPERVISED LABELING algorithm, named SSL.

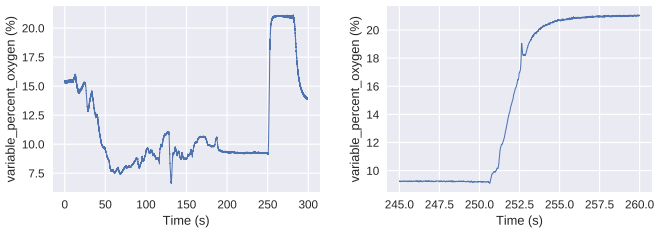
Specifically, recording the cut-off part with a higher sampling frequency – as is done by Program B – makes the labeling of the cycles possible but, since having access to data covering a larger timespan could offer more useful insights, the data from Program A will be used as well.

SSL performs three main steps:

- **Response time measurement**: in order to understand whether a cycle is clogged or not, the response time after the cut-off needs to be measured, using the definition provided by GM on the Program B dataset.
- **Labeling**: based on the response time, cycles are labeled as either clogged or not or, to predict future clogging situations, as "almost clogged".
- **Mapping**: the unlabeled dataset resulting from the data transformation step receives the labels computed at the labeling step, producing the final labeled dataset. The mapping of the Program A to the Program B cycles requires some attention in terms of implementation, but these are not details worth mentioning in this place.

A. Response time measurement

The oxygen level measured in the exhaust gas is directly influenced by the state of the acceleration pedal, since this drives the load on the engine. During the cut-off, when the pedal is released, the oxygen level rises to the 21% value found in the atmosphere. The time needed to reach this value is called response time.



(a) $0s \div 300s$ (b) $245s \div 260s$

Fig. 3: Oxygen level, as measured in Program B

Figure 3a shows the oxygen level as reported throughout the final 300 seconds by Program B, while Figure 3b zooms in to provide a better view of the oxygen level during the cut-off.

GM definition of response time is the time it takes for the oxygen level to reach 63% of the transitory from the initial value to 21%, as explained in Equation 1

$$t_r = t(O_2 = 0.63 \cdot (O_{2end} - O_{2start})) - t(O_2 = O_{2start}) \quad (1)$$

Figure 4 provides a visual interpretation of what the response time represents.

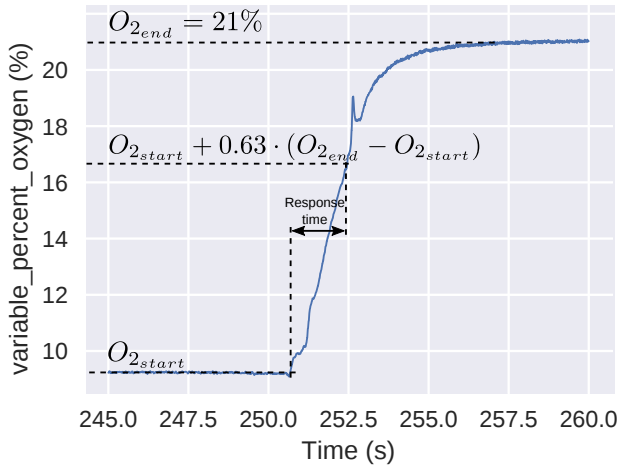


Fig. 4: Response time measurement process

Since the order among cycles is known, a graph representing the response time trend from the beginning to the end of the experiments can be plotted and is shown in Figure 6.

The response times ranges anywhere between 1 and 2 seconds. Measuring these responses using the 1 Hz sampling frequency provided by Program A would have yielded useless results: this justifies the utilization of the Program B dataset for this operation.

B. Labeling

Three classes of clogginess have been defined based on the response time: green for "unclogged" cycles, yellow for intermediate and red for clogged ones. The threshold values then need to be defined for each class. Given the physical constraints of the response time, it is reasonable to assume the lower bound of the green class to be 0 and the upper bound of the red class to be $+\infty$. This requires the definition of the two intermediate thresholds. Without any domain-driven insights on the identification of the thresholds, these have been defined so as to have the red class as a minority one (as is expected to be the case, given that the "red" class represents a state of malfunctioning) and the other two classes with similar cardinalities. This decision has been mainly piloted by the distribution of response times, which is presented in Figure 5, along with the already-defined classes.

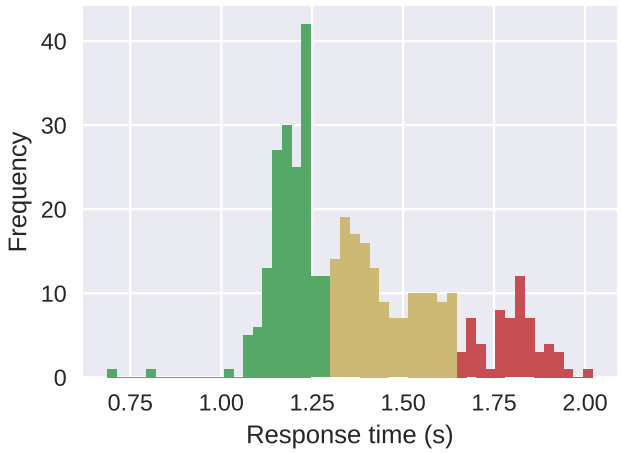


Fig. 5: Distribution of response times: the colors represent the assigned classes based on the defined thresholds

Figure 6 illustrates the response time evolution throughout the cycles of the experiment. The horizontal lines represent the upper and lower bounds for the yellow class. The jagged profile of the curve represents a problem: if labels were to be assigned based on the simple comparison against the defined thresholds, those sequences of cycles that are crossing the thresholds would be alternatively assigned different labels. This behavior is counterintuitive and undesired, as the cumulation of soot is expected to occur gradually.

In order to lessen the severity of this problem, a moving average has been introduced: each sample in the sequence is averaged with the previous k and the following k samples. The width of the time window (k) has been set empirically (see Section VII-A1 for more detail).

VI. Cycle modeling and prediction

In literature, a myriad of classifiers have been presented, each with its strengths and weaknesses: there is no classifier that is overall preferable to others. In order to get the best

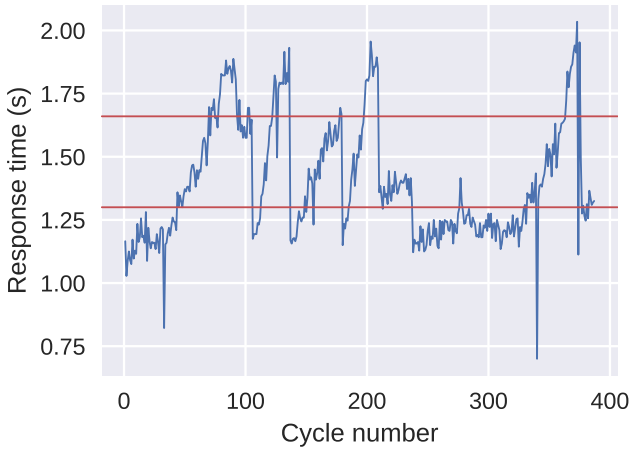


Fig. 6: Response time trend throughout the experiment. The horizontal lines show the positions of the thresholds

results, OxyClog integrates different classifiers with different advantages and disadvantages. The selection of the classification algorithms has been carried out with two purposes in mind: the first one is to learn which input variables mainly drive the prediction problem, the second is to build an effective classification algorithm. Because of this, the chosen algorithms leverage either of these two aspects. Decision trees [7] have been selected because of their interpretability, while multilayer perceptron neural networks [9] and SVM [8] have been chosen for their capability of separating classes with functions other than hyperplanes that are orthogonal to the dimensions available, as is the case with decision trees. Here we briefly describe each selected classifier.

Decision trees [7] are a well-known popular and mature technique capable of reaching both a good accuracy and easy model interpretability, with the latter being a highly-valued feature for domain experts. A decision tree classifier is grown in a recursive fashion by iteratively partitioning the training data (cycles for which the clogging status is known) into successively purer subsets. In the tree structure, each node specifies a test on a data feature, and each branch descending from that node corresponds to one of the possible values for that feature. Each leaf node represents class labels associated with the instances having, as attribute values, the values appearing in the path connecting the root node to the leaf one.

Artificial neural networks [9] simulate biological neural systems. A network consists of an input layer, n hidden layers, and an output layer. Each layer is made up of nodes. Each node in a layer takes as input a weighted sum of the outputs of all the nodes in the previous layer, and it applies a nonlinear activation function to the weighted input. The network is trained with backpropagation and learns by iteratively processing the set of training data objects. For each training data object, the network predicts the target value.

Then, weights in the network nodes are modified to minimize the mean squared prediction error. These modifications are propagated backwards, that is, from the output layer through each hidden layer down to the first one. The neural network used is a multilayer perceptron one.

Support Vector Machine Support Vector Machines (SVM) [8] have been first proposed in statistical learning theory. SVM is able to deal with high-dimensional data and it generates a quite comprehensive (geometric) model. An SVM predictor is based on a kernel function K that defines a particular type of similarity measure between data objects. Examples of kernel functions are linear, RBF (Radial Basis Function), polynomial, or sigmoid kernel. The SVM learning problem can be formulated as a convex optimization problem, in which different algorithms can be exploited to find the global minimum of the objective function.

VII. Experimental results

Experimental validation has been carried out to assess OxyClog's performance in terms of predicting capabilities and to offer support to the definition of the values of the framework's parameters.

A large number of experiments has been performed on a real dataset including a set of cycles -collected in a controlled environment and for a specific engine. Program A and B (as discussed in Section II) have been exploited to collect data. The average size for a single Program A cycle is roughly of 1.6 MB, while the average Program B file is 43.7 MB large. Given the cardinalities for the two datasets, the overall data size is roughly $1.6\,MB \cdot 400 + 43.7\,MB \cdot 388 \approx 17.2\,GB$.

The unlabeled dataset resulting from the process described in Section IV and the labels identified by the process in Section V are merged together to create the final, labeled dataset. The class assigned to each cycle is either of the labels 'red', 'yellow' or 'green' based on the OxyClog's labeling process.

The current implementation of OxyClog is a project developed in Python exploiting the numpy and pandas libraries for the data preprocessing and labeling operations, while scikit-learn has been used for its classification models. The parameters introduced by OxyClogHave been defined as follows: $r_{min} = 0.8$ and $k = 2$. Subsection VII-A provides further details regarding the identification of these values.

On the other hand, the hyperparameters for the classification models have been tuned using 10-fold cross validation and an exhaustive grid search. In terms of performance, the classifiers have been evaluated using different metrics. The accuracy of the classifiers is not a reliable metric since the red class (that is the most interesting of the three classes) is the one with the lowest number of cycles. The F_1 score for the red class has been considered instead. The configuration of the decision tree selected by the grid search reaches a maximum depth of 6 and, within this constraint, it continues expanding each node that has a minimum number of 4 samples.

The neural network used is a multilayer perceptron one: the number of layers and the number of neurons in each layer have been selected by means of the grid search, which resulted in the selection of two layers with 500 and 100 neurons respectively. In hindsight, the number of layers and their size is overabundant but, for the sake of keeping the tuning process as little influenced by human intervention as possible, this configuration has been kept.

The SVM configuration, finally, uses a linear kernel and a value for the parameter C of approximately 2.5. This parameter defines a trade-off in the maximization of the margin found by the SVM versus the possibility of misclassifying some points. A small value such as the one identified tends to maximize the margin, even though this means introducing some misclassified cycles.

All experiments have been performed on a dedicated server running Ubuntu 16.04, with 12 cores at 2.67 GHz, 32 GB of main memory and 15 TB of storage.

A. OxyClog *parameter setting*

This section contains a discussion of the analysis performed to set the OxyClog's parameters: (i) the width of the window k used to identify the class label for each cycle; (ii) the correlation threshold r_{min}.

1) Time window width for data labeling: The jagged profile of the curve in Figure 6 may be, as already explained, slightly problematic: subsequent cycles whose response times are in the proximity of the defined thresholds may be assigned different labels although the expectation would be that they mostly belong to the same class.

In order to lessen the severity of this problem, a moving average has been introduced: each sample in the sequence is averaged with the previous k and following k samples. k has been identified by analyzing how the number of label switches evolves as k increases, as shown in Figure 7. The knee of the curve indicates what is believed to be an acceptable value.

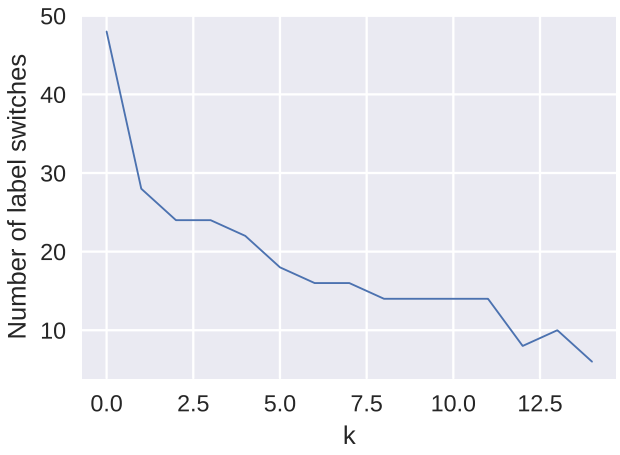
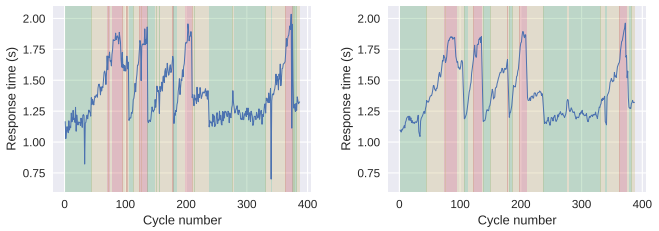


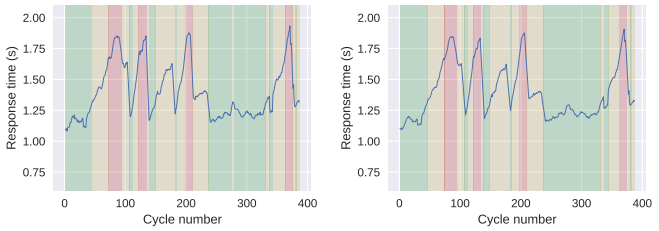
Fig. 7: Number of label switches occurring as k increases

Although the identified value for k is intuitively small, a visual inspection of the profile of the smoothed curve with this value helps understand whether the resulting alternation in values is acceptable or not. Figure 8 shows the results for

4 different values of k (0 through 3): the vertical bands are colored based on the assigned label for each cycle. In general, the profile of the curve for $k = 2$ has been deemed acceptable from this perspective as well.



(a) $k = 0$ (b) $k = 1$



(c) $k = 2$ (d) $k = 3$

Fig. 8: Response time trend smoothed with different k values

From the smoothed values, the labels can be assigned to each Program B cycle: the cardinalities for the three classes are reported in Table II.

Class	Cardinality
Green	164
Yellow	163
Red	61

TABLE II: Cardinalities for the three identified classes

2) Analysis of the correlation threshold: Here we discuss the number of selected variables by varying the input parameter r_{min}. The selection of this value requires finding a trade-off between the number of selected features and their ability to well represent the discarded variables. The value can be selected either by setting an *a priori* constraint on the desired representativeness of the selected features, or by studying the evolution of the number of selected features as the coefficient changes.

Since the CORR-FS algorithm takes the absolute value of each correlation coefficient (because negative correlations are correlations nonetheless), it makes sense to only analyze values for $r_{min} \in [0, 1]$. Intuitively, $r_{min} = 0$ implies that the lowest number of variables is selected (possibly only 1), given that each selected feature "covers" all others that are not completely independent from it (i.e. correlation coefficient not 0). By contrast, setting $r_{min} = 1$ results in the largest number of features selected, as each selected feature only covers for the completely correlated ones.

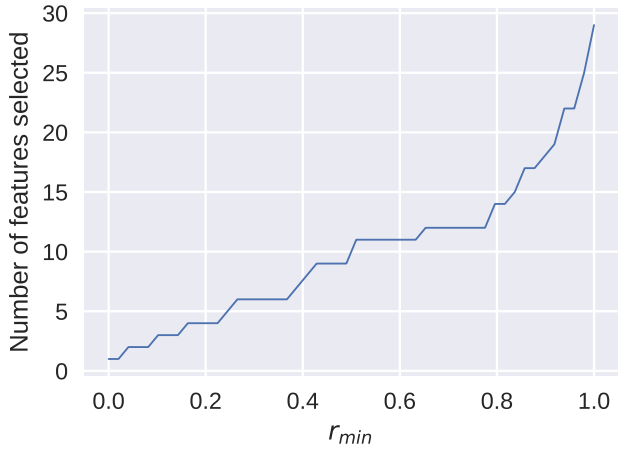


Fig. 9: Number of features selected as r_{min} increases

Figure 9 illustrates how the number of features selected grows as r_{min} increases. The elbow of the curve occurs slightly below the 0.8 value: as such, this has been chosen for r_{min}. Using this large correlation threshold guarantees that all discarded signals have a very strong correlation with the selected ones, keeping the amount of discarded information to a minimum.

The described feature selection algorithm was applied to the available Program A cycles, in order to reduce the overall number of features to be further processed: this resulted in 14 variables being selected.

For each of the 14 variables, OXYCLOG computes $(9 + 2) \cdot 2 = 22$ statistics, for a total of 308 features.

B. OXYCLOG*'s performance*

OXYCLOG measures the quality of the prediction model through different quality indices: accuracy, precision, recall and F_1 score. While the accuracy measures the overall efficacy of the classifier, the others offer insights on the performance of the classifier with respect to a given class.

The precision is the fraction of cycles assigned to a class that has been correctly identified, while the recall is the fraction of cycles belonging to a class that have been correctly identified as belonging to that class. The F_1 score is the harmonic mean of precision and recall. The red class, other than being the minority one (see Table II) is the one of most interest, since it represents the condition of major clogging that this study attempts to predict. As such, the F_1 score for the red class has been used to evaluate the classifiers, but the presented results include precision, recall and F_1 score for the three classes, along with the estimated overall model accuracy.

Table IIIa contains the results for the decision tree classifier, Table IIIb the ones for the neural network and Table IIIc the ones for the SVM.

The overall scores show that the classifiers reached good quality results, significant of the fact that the classifiers can actually handle the red class well, despite its low cardinality.

	Green	Yellow	Red
Precision	0.8365	0.7529	0.8475
Recall	0.8110	0.7853	0.8197
F_1 score	0.8235	0.7688	0.8333
Accuracy		0.8015	

(a) Decision tree results

	Green	Yellow	Red
Precision	0.8869	0.8679	0.9016
Recall	0.9085	0.8466	0.9016
F_1 score	0.8976	0.8571	0.9016
Accuracy		0.8814	

(b) Neural network results

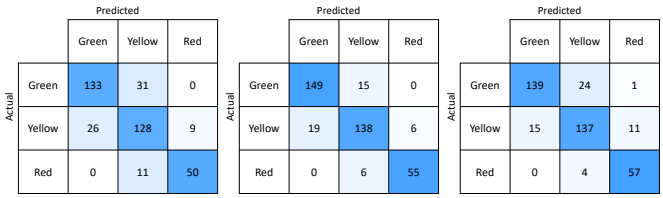
	Green	Yellow	Red
Precision	0.9026	0.8303	0.8261
Recall	0.8476	0.8405	0.9344
F_1 score	0.8742	0.8354	0.8769
Accuracy		0.8582	

(c) SVM results

TABLE III: Results for the classification problem

The decision tree, as expected, performed slightly worse than the alternatives. This underperformance is compensated by the interpretability of the model. Indeed, the tree is the only model for which the internal representation can be meaningful.

A different way of visualizing the performance for each of the classes is by means of confusion matrices. These matrices are reported in Figure 10 for each of the three classifiers. The main diagonals of the matrices contain the large bulk of cycles: this means that most green, yellow and red cycles have been classified correctly by the three models, in accordance with the large values of precision, recall and F_1 score obtained.



(a) Decision tree (b) Neural network (c) SVM

Fig. 10: Confusion matrices for the three classifiers

An interesting insight offered by the confusion matrices is the fact that most misclassifications occur between "adjacent" classes: only one of the misclassifications occurs between green and red cycles. This is representative of the fact that the classifiers "learn" the existence of an order among classes, with green cycles being more similar to the yellow than they are to the red ones.

While the resulting decision tree obtained will not be analyzed in detail, the root node will be discussed as it represents one of the most important findings of this study. The best split identified by the algorithm is on one of the percentiles of the derivative of the oxygen variable. This result may seem intuitive, as cycles are labeled based on the duration of the response time: the measurements of the oxygen variables are slower in clogged cycles, as such the distribution of values of the oxygen derivative are expected to be shifted towards lower values when compared to green cycles. This result, though, is not trivial: it shows how the identification of clogged situations can occur even without the high-frequency sampling of a very specific maneuver, but can instead be inferred from the general utilization of the vehicles for a prolonged period of time.

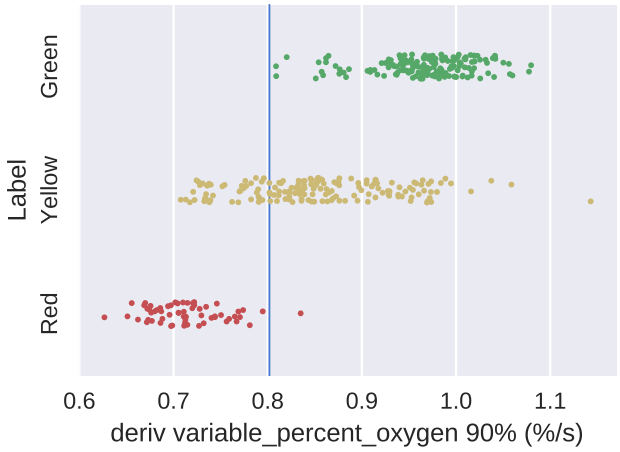


Fig. 11: Distribution of the 90^{th} percentile of the derivative of the oxygen variable, by class

Figure 11 shows the 90^{th} percentile of the derivative of the oxygen variable for each of the cycles, divided by classes. This clearly highlights how the three classes are distributed based on this variable. The green and the red classes are well separated, with the yellow one in between the other two. The plot shows the separating value between the red and green classes (obtained by generating a decision tree without the yellow class).

VIII. Discussion

While the focus of this work was on a very specific sensor, the lessons we learned can be generalized not only to other automotive contexts, but also to other real cases facing similar problems. The proposed framework addressed the following relevant issues.

- *Importance of feature selection interpretability.* The feature selection algorithm is interpretable in terms of results (as it does not alter the input signals) and in terms of configuration (since the only parameter is a threshold value defined in terms of correlation among signals). These characteristics allowed the domain experts to effectively validate both the correlation analysis and the algorithm results.
  The interpretability of the process and results also narrows the amount of data to be collected to the minimum necessary to make predictions. This will facilitate industrial deployment by lowering the bandwidth required to transfer the data through OnStar.

- *Conversion of a time-dependent problem into a time-independent one.* The data transformation process leveraged summary statistics to reduce the dimensionality of the data, while the introduction of the derivatives of the signals preserved high level information about the evolution in time of the considered signals.
  The exploitation of summary statistics to describe the signals will further reduce the required bandwidth for transmission. Summary values can be computed locally and then transmitted, lowering the uploading period down to seconds, rather than minutes or hours.
- *Exploitation of heterogeneous data sources for labeling purposes.* For this case study, as is the case with many real-world problems, the sources of data were multiple and different in nature. By taking them into account simultaneously, it was possible to define a semi-supervised labeling algorithm. This algorithm only partially relies on human intervention for the overall supervision of the process. Human intervention is required to identify the cut-off point in the accelerator track and to select reasonable thresholds and smoothing parameter. After that, the proposed process takes care of labeling any number of cycles without any additional manual operation.

The successful validation of the experimental results by the application domain experts shows the suitability of the presented approach for dealing with time dependent data. The presented methodology has been further validated on a separate dataset collected from a different engine model with similar results, which are not included in this paper due to space constraints.

IX. Conclusions and future work

The OXYCLOG framework addresses the prediction of the faulty behavior of the oxygen sensor by (i) preparing and transforming engine data collected in a test bench, (ii) assigning a label to the collected data by exploiting different data sources, and (iii) predicting the current clogging state of the sensor by means of state-of-the-art machine learning approaches. The objective is decoupling the detection of the clogging problem from the currently used, very specific maneuver, the cut-off. OXYCLOG exploits machine learning techniques to identify the clogging state by analyzing ECU sensor readings during the "normal" usage of the vehicle.

The presented methodology approached the different peculiarities of the case study by devising a tailored – but generalizable – approach that yielded valuable and actionable results for the domain experts. The most important features of the OXYCLOG framework are an interpretable feature selection process, a summarized representation of the time component, and a semi-supervised labeling process based on the availability of data sources with different characteristics.

This study focused on test bench cycles that shared the same gas pedal track. As future work, a generalization of this process for "on-the-road" vehicles is under evaluation. In this case, larger amounts of data can be collected in different conditions and more general models can be generated. Since different driving conditions may affect significantly the behavior of the measured variables, an additional "context identification" step may be needed. Based on the context in which the measurements are collected (e.g., in an urban setting or in a motorway) different tailored models may be selected.

References

[1] Annamalai Pandian and Ahad Ali. "A review of recent trends in machine diagnosis and prognosis algorithms". In: *Nature & Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*. IEEE. 2009, pp. 1731–1736.

[2] Andrea Cordoba-Arenas, Jiyu Zhang, and Giorgio Rizzoni. "Diagnostics and prognostics needs and requirements for electrified vehicles powertrains". In: *IFAC Proceedings Volumes* 46.21 (2013), pp. 524–529.

[3] Somnath Pradhan and Joydeb Roychaudhury. "A machine learning approach to predict future power demand in real-time for a battery operated car". In: *IMpact of E-Technology on US (IMPETUS), 2014 International Conference on the*. IEEE. 2014, pp. 49–56.

[4] S Jagannathan and GVS Raju. "Remaining useful life prediction of automotive engine oils using MEMS technologies". In: *American Control Conference, 2000. Proceedings of the 2000*. Vol. 5. IEEE. 2000, pp. 3511–3512.

[5] Mitchell Lebold et al. *Detecting injector deactivation failure modes in diesel engines using time and order domain approaches*. Tech. rep. Pennsylvania State Univ State College, 2012.

[6] David Gucik-Derigny, Rachid Outbib, and Mustapha Ouladsine. "Estimation of damage behaviour for model-based prognostic". In: *IFAC Proceedings Volumes* 42.8 (2009), pp. 1444–1449.

[7] Leo Breiman et al. *Classification and regression trees*. CRC press, 1984.

[8] Corinna Cortes and Vladimir Vapnik. "Support-Vector Networks". In: *Machine Learning* 20.3 (1995), pp. 273–297.

[9] Jürgen Schmidhuber. "Deep learning in neural networks: An overview". In: *Neural Networks* 61 (2015), pp. 85–117.

[10] Jolliffe I.T. *Principal Component Analysis*. Springer-Verlag New York, 2002.

[11] Jianping Hua et al. "Optimal number of features as a function of sample size for various classification rules". In: *Bioinformatics* 21.8 (2005), pp. 1509–1515.