A Data-driven ApproachRiconoscimento autori
for Enhanced On-Board Fault Diagnosis to support Euro 7 Standard Implementation

A Data-driven Approach for Enhanced On-Board Fault Diagnosis to Support Euro 7 Standard Implementation

Author, co-author (Do NOT enter this information. It will be pulled from participant tab in MyTechZone)

Affiliation (Do NOT enter this information. It will be pulled from participant tab in MyTechZone)

Abstract

The European Commission is going to publish the new Euro7 standard shortly, with the target of reducing the impact on pollutant emissions due to transportation systems. Besides forcing internal combustion engines to operate cleaner in a wider range of operating conditions, the incoming regulation will point out the role of On-Board Monitoring (OBM) as a key enabler to ensure limited emissions over the whole vehicle lifetime, necessarily taking into account the natural aging of involved systems and possible electronic/mechanical faults and malfunctions. In this scenario, this work aims to study the potential of data-driven approaches in detecting emission-relevant engine faults, supporting standard On-Board Diagnostics (OBD) in pinpointing faulty components, which is part of the main challenges introduced by Euro7 OBM requirements. For this purpose, a data-driven model for the detection and identification of different faults of engine components and sensors, which takes as input available on-board measurements and Engine Control Unit (ECU) signals, has been developed using different classification algorithms. The classification model has been optimized, trained, and tested on simulation data generated by a validated 0-D Simulink model representative of a light-duty Diesel plug-in hybrid electric vehicle (PHEV). The best classification algorithm and configuration of hyperparameters have been chosen, and the selected model has been integrated into the ECU software developed in Simulink®. Possible faults significantly affecting pollutant emissions have been selected and simulated, and the accuracy of fault detection obtained with the implemented classification model has been evaluated. In view of a vehicle on-board application, the developed model has been implemented on a real-time hardware to evaluate its real-time capability. The preliminary results obtained in terms of effectiveness, robustness, and real-world applicability pave the way for further investigations in this field, as a promising solution to help facing the upcoming Euro7 standard.

Introduction

Background and motivations

Fault diagnosis is a crucial aspect to increase safety, reliability, and efficiency of complex dynamic systems, avoiding as much as possible severe consequences on humans and environment, limiting repair costs, and shortening downtime periods which can cause inconveniences and significant economic losses. Therefore, many studies have been carried out, all referable to the subfield of control engineering known as Fault Detection and Isolation (FDI) [1], focused on developing monitoring systems that can recognize when a fault occurs and identify its type and location. FDI approaches can be divided into three macro-categories: model-based FDI, where a mathematical model of the system is used to determine if the system is failing; knowledge-based FDI, which uses graphical models for diagnostic inference; data-driven FDI, implementing signal processing or machine learning (ML) models [2], [3]. Due to the increased complexity and automation degree of the monitored systems, the development of model-based diagnostic approaches capable of considering discrete and continuous system dynamics as well as the interactions between the large number of interconnected components included in the system, has become a challenging task [4]. Moreover, enhanced by the advancements involving internet of things, wireless communications, and development of mobile devices, the amount of available data that could be used for fault diagnosis purposes has grown in an exponential manner. In this context, as evidenced by literature [5], data-driven FDI has gained significant popularity, proving to be a promising, feasible, and cost-effective tool to detect and classify the occurrence of faults in complex dynamic systems. Data-driven ML approaches applied to fault diagnosis can be classified into three types: unsupervised learning methods, including Principal Component Analysis (PCA), Independent Component Analysis (ICA), and K-means clustering; supervised learning, including both classification algorithms and inference-based approaches; semi-supervised-learning, which is a combination of the two. As reviewed in [6], supervised learning is the most common approach used for fault diagnosis and it has been already explored within several different application fields, including photovoltaic systems [7], aircraft gas turbines [8], rotating machinery [9] (e.g. electric motors [10]–[12], bearings [13], automotive transmissions [14], etc.), industrial machinery [15], unmanned aerial systems [16], nuclear power plants [17] and shipboard systems [18], [19].

Focusing on the automotive sector, with the introduction of the new Euro 7 regulation, continuous ultra-low emissions will have to be ensured throughout a significantly extended vehicle lifetime and to be confirmed by real-world testing and on-board monitoring (OBM) of defined pollutant emissions [20], [21]. Thus, fast and reliable detection and identification of emission-relevant faults of engine sensors and components play an important role in ensuring the nominal expected emission behavior of the monitored vehicle. In this context, this work aims to study the potential of data-driven approaches in detecting emission-relevant engine faults, supporting standard On-Board Diagnostics (OBD) in pinpointing faulty components, which is

part of the main challenges introduced by Euro7 OBM requirements [22]. Moreover, this information could also be sent to a workshop, delivering valuable information for more accurate and efficient maintenance measures, thus minimizing vehicle downtime and unnecessary service effort. In this regard, different ML-based classification methods, including Classification Trees, Ensemble methods, and Neural Networks have been considered to develop an accurate and robust FDI model which takes as input available real-time on-board measurements and Engine Control Unit (ECU) signals.

Related works and novel contributions

This paragraph presents a selection of related works from the field of fault and anomaly detection applied to automotive engines. A comprehensive overview of condition monitoring and data processing techniques for FDI in internal combustion engines is given by [23]. In [24], a multi-layered artificial neural network (ANN) model for engine failure classification using the extracted features from sound intensity analysis is presented. With the proposed approach, different fault classes can be detected, but a dedicated setup for sound acquisition is required, which makes it not suitable for vehicle on-board implementation. In [25], the authors propose an alternative approach, where the input to the ANN-based classification model is engine vibration data. The developed fault diagnostic system is able to detect known engine faults with various degrees of severity, but the test is necessarily conducted in a semi-anechoic chamber to isolate the engine from any external noise that might affect the vibration response of the system. A similar approach is proposed in [26] to detect different faults including malfunction of manifold absolute pressure, knock sensor, and misfire by analyzing vibration signals. The designed multi-layered ANN can classify different fault situations with high accuracy, being trained on both selected single faults and combinations of them. A data-driven approach to detect knock phenomena using ECU signals instead of vibration data is presented in [27]. In this case, different unsupervised ML methods, including Autoencoder, Support Vector Machines (SVM), and Isolated Forest, have been investigated to perform anomaly detection through binary classification. Similarly, [28] presents a deep neural network (DNN)-based pre-ignition detection starting from all available ECU signals. In [29], a probabilistic neural network (PNN) is designed to detect different engine faults based on the information extracted from exhaust gases, namely HC, CO, NO_x, CO_2, and O_2 concentrations. High accuracy is reached, but measurements obtained with a dedicated exhaust gas analyzer are required. [30] presents a fault diagnosis system based on sparse Bayesian extreme learning machines (SBELM) which is able to detect both single and simultaneous engine faults while the system is trained by single-fault patterns only. However, an inductive pickup clamp, five gas analyzers, and a microphone are used to acquire the air-to-fuel ratio, ignition, and acoustic wave patterns respectively, which makes on-board implementation of such a system not feasible. In [31] the authors propose a fuel consumption classification system for heavy-duty vehicles (HDVs) based on bagged decision trees. The aim is to detect high fuel consumption patterns which are independent of vehicle load and slope, to identify anomalous driving behaviors or system failures, without focusing on the root cause of the anomaly. [32] presents a hybrid methodology utilizing both model-based residuals generation together with a long short-term memory neural network (LSTM NN) to predict selected faults. Only one residual is used and a very limited number of faults is considered if compared to other reviewed researches. [33] proposes a residual selection algorithm that combines model-based and data-driven methods to improve the effectiveness of anomaly detection and fault isolation properties using model-based diagnosis systems. In [34] an artificial neural network bank combined with available measurements is used to identify damaged engine sensors and replace them with an estimated signal. A data-driven approach based on two-class and one-class classifiers trained on a dataset of vehicle recordings is proposed in [35]. The aim is to classify data as either normal or anomalous, which means a potential fault, regardless of the cause of the anomaly. Based on strengths and weaknesses of previous related studies, the aim of this work is to propose a simple data-driven methodology to perform detection and classification of selected emission-relevant engine faults and combination of them, taking advantage of already available on-board signals, considering real-time implementability as well in view of on-board application.

The main contributions of this paper are summarized as follows:

- Different emission-relevant engine faults have been investigated.
- Several ML-based classification methods have been compared in terms of accuracy and computational effort to select the best trade-off.
- Input features for the FDI model are available ECU signals and standard on-board measurements.
- A classification model able to detect and identify different fault categories, involving sensors, actuators, or other engine components, has been developed.
- The developed model is able to handle both single faults and combinations of faults, being trained just on single faults.
- The real-time capability of the chosen classification approaches has been evaluated by implementing the FDI model on a real-time hardware.

Materials and Methods

The manuscript is structured as follows: firstly, an overview of the modeling approach is given, with a particular focus on the engine model, the modeling of specific engine faults and their impact on engine-out emissions. In the second part, the development and implementation of the proposed fault classification model is described. The approach that has been adopted to define the most suitable classification method is also introduced. Lastly, a summary of the considered test cases and conditions to evaluate the performance of the proposed FDI model is presented.

Simulation environment

Vehicle model

A comprehensive 0-D control-oriented vehicle model has been developed in MATLAB®/Simulink®, starting from already existing and previously validated simulation platforms made available by FEV. The modeled vehicle is a prototypal Diesel plug-in hybrid light commercial vehicle (LCV) property of FEV, which is currently used for research purposes. The considered hybrid architecture, represented in Figure 1, provides high flexibility in terms of possible operating modes and torque split strategies. The location of the electric machines, thanks to the separation clutch, allows to perform effective regenerative braking as well as pure electric drive. The combination of a 15 kWh high-voltage (HV) battery and a 50 kW/350 Nm electric motor (EM) ensures a significant all-electric range. The electric power unit is paired with a 2-liter compression ignition engine and an 8-speed manual transmission. All the main vehicle specifications are listed in Table 1.

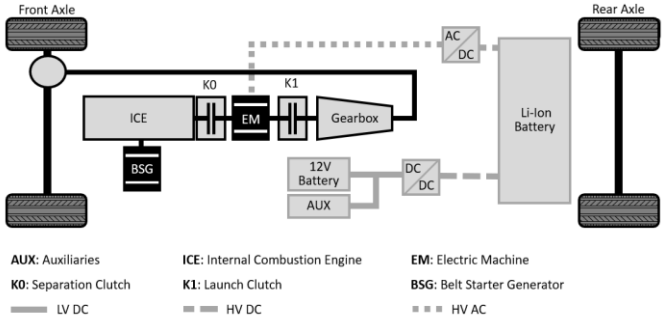


Figure 1. Schematic representation of vehicle hybrid powertrain architecture.

Table 1. Main powertrain and vehicle specifications.

Vehicle	Curb weight	3330 kg
	Configuration	P0P2, front-wheel drive
Internal combustion engine	Type	Diesel, turbocharged
	Displacement	2.0 l
	Rated power	120 kW @ 3600 rpm
	Rated torque	400 Nm @ 1800 rpm
	Gearbox	8-AT
Belt starter generator	Type	PM synchronous
	Peak power	9 kW
	Peak/continuous torque	50/25 Nm
	Maximum speed	15000 rpm
Electric machine	Type	PM synchronous
	Peak power	50 kW
	Peak/continuous torque	350/250 Nm
	Maximum speed	7000 rpm
Battery	Type	Lithium-ion polymer
	Capacity	15 kWh / 50 Ah
	Nominal voltage	300 V

An accurate analytical model has been adopted for both vehicle longitudinal dynamics and the electric power unit, including the EM and the HV battery. A detailed description and validation of the mentioned models are proposed in [36]. A mean value modeling approach has been adopted for internal combustion engine (ICE) modeling, which is briefly described in the next dedicated section.

Mean Value Engine Model (MVEM)

The core of the simulation environment is the mean value engine model (MVEM), which includes the main engine sub-models, namely the air system that determines how much air is inducted into the cylinder; the fuel system that determines how much fuel is inducted into the cylinder; the torque generation system that determines how much torque is produced by the air and fuel in the cylinder as determined by the first two parts; the engine thermal system that determines the dynamic thermal behavior of the engine; the pollution formation system that models the engine-out emission. By definition, MVEMs are control-oriented models, meaning that they model the input-output behavior of the systems with reasonable precision but low computational complexity. The reciprocating behavior of the engine is not considered, neglecting the discrete engine cycles and assuming that all processes and effects are spread out over the engine cycle, still including all relevant transient effects [37]. The MVEM considered for this publication combines physics-based models, i.e., models that are based on physical principles and on a few experiments necessary to identify some key parameters, with a model-based ECU control structure, leading to a flexible simulation software that can be eventually adapted to be representative of different hardware architectures [38]. For the sake of brevity, a detailed description of the modeling approach and involved physical equations is not included in this paper, however it can be found in [39], as well as an overview of model validation on experimental data. The most relevant raw emissions, namely NO_x, HC, CO, and soot, are properly predicted within the engine model with a semi-physical approach, reaching accurate results also for dynamic responses thanks to a set of corrections based on oxygen concentration, temperatures, pressures, or

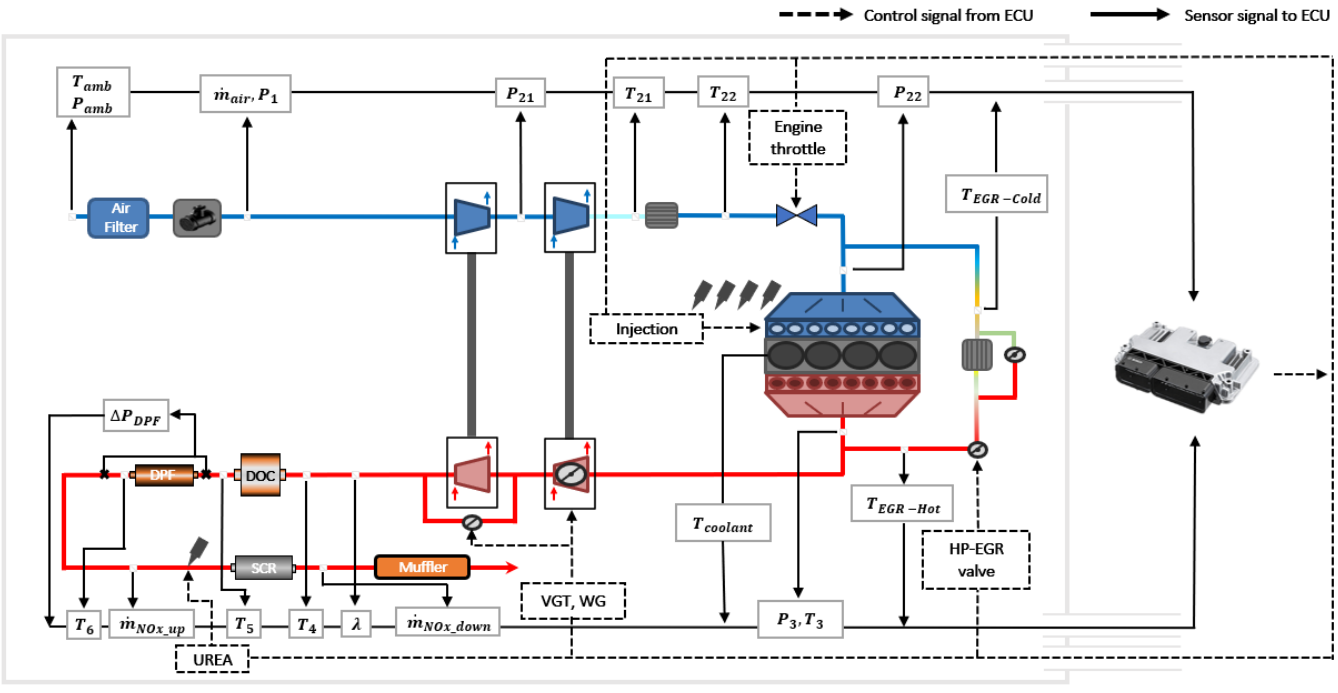


Figure 2. Engine and EATS architecture.

combustion properties (center of combustion, rail pressure, etc.) [40]. The modeled vehicle is equipped with a Diesel Euro 6 after-treatment system (ATS), which includes a Diesel oxidizing catalyst (DOC), a Diesel particulate filter (DPF), and a selective catalytic reduction (SCR) catalyst. The resulting pollutant emissions after oxidizing/reducing reactions that take place in the ATS are modeled with a map-based approach, by first calculating the conversion efficiency of each ATS component, as a function of gas average temperature, space velocity, and NO_2/NO_x ratio (in the case of NO_x conversion efficiency). Since out of the scope of the paper, for a comprehensive description of the ATS modeling approach, the reader is referred to [41]. All the mentioned maps for the engine and ATS models have been obtained from experimental tests previously carried out within FEV and the related data are confidential. Figure 2 shows the engine and ATS architecture of the considered vehicle, highlighting the most significant signals sent from the onboard sensors to the ECU and from the ECU to the actuators.

Emission-relevant faults investigation

Different types of engine faults have been simulated to evaluate their effect on engine-out pollutant emissions and the capability of the engine controller to react to these faults. The selected faults are summarized in Table 2: starting from the left, the simulated physical fault is described and a brief explanation of how it has been implemented into the engine plant model is given; then a qualitative summary effect on the main pollutant emissions, based on a deeper analysis involving several simulations in steady-state conditions for the different faults and degrees of severity, is expressed by the signs: plus (+) or minus (–), indicating whether an increase or decrease with respect to nominal conditions is observed, respectively; 1 to 3 signs depending on the magnitude of the variation. Besides the effect on actual pollutant emissions calculated by the physical engine model, the deviation between the emission level expected by the engine controller and the actual emissions calculated by the physical model is shown on the right side of the table. The analysis shows that almost all the considered faults have a very significant effect on NO_x and soot emissions (up to more than 50% increase compared to nominal) and that the physical-based models inside the ECU are not able to adapt themselves to correctly predict emissions in case of unexpected faults, despite the available measurements coming from the engine physical model. In fact, when a fault is considered, a variation with respect to nominal conditions is observed also for the physical quantities (mass flow, pressure, temperature, and oxygen concentration) of the intake and exhaust paths. This is reflected in a deviation between the expected values calculated by the ECU and the measured quantities that are sent from the sensors to the ECU. The magnitude of this deviation depends both on the type and extent of the considered faults and on the capability of the ECU models to adapt the calculations of the considered physical quantities, taking advantage of the other available measurements. For this reason, the set of residuals obtained from all the measured quantities, intended as the difference between the expected and actual values, can be used as an input to data-driven models to detect the presence of faults and identify them based on the characteristic trends of the considered residuals and their intercorrelations. As an example, Figure 3 shows the trend of the considered residuals in presence of two different intermittent faults, namely a low drift of the intake manifold absolute pressure (MAP) sensor and a high drift of the mass air flow (MAF) sensor.

Table 2. List of simulated engine faults and their related effects on resulting pollutant emission and controller emissions estimation.

Fault	Model implementation	Effect on physical model (fault VS nominal case)			Effect on controller model (controller VS physical model)		
		NO_x	HC, CO	soot	NO_x	HC, CO	soot
Clogged EGR valve	Upper saturation on EGR valve opening in the physical model	⊕⊕⊕	⊖	⊖⊖	⊖⊖	⊕	⊕⊕⊕
Reduced EGR cooler efficiency	Gain <1 on nominal EGR cooler efficiency	⊕⊕⊕	⊖	⊕⊕	⊖⊖	⊕	⊖
MAF sensor high drift	Gain >1 on air mass flow signal sent from the physical to the controller model	⊖⊖	⊕	⊕⊕⊕	⊕⊕⊕	⊖	⊖
MAF sensor low drift	Gain <1 on air mass flow signal sent from the physical to the controller model	⊕⊕⊕	⊖	⊖⊖	⊖	⊕	⊕⊕
Intake manifold pressure sensor high drift	Gain >1 on pressure signal sent from the physical to the controller model	⊖⊖	⊕⊕⊕	⊕⊕⊕	⊕⊕⊕	⊖⊖	⊖⊖
Intake manifold pressure sensor low drift	Gain <1 on pressure signal sent from the physical to the controller model	⊕⊕⊕	⊖	⊖⊖	⊖	⊕	⊕⊕⊕

○	Maximum observed variation between 10% and 50% for the considered steady-state engine operating points and fault conditions.
○○	Maximum observed variation between 50% and 100% for the considered steady-state engine operating points and fault conditions.
○○○	Maximum observed variation above 100% for the considered steady-state engine operating points and fault conditions.

Table 3. List of considered physical quantities for residuals calculation.

Abbreviation	Description
$\dot{m}_{air}$	Intake air mass flow
P_{21}	Pressure upstream high-pressure compressor
P_{22}	Intake manifold pressure
P_3	Exhaust manifold pressure
T_{21}	Temperature downstream high-pressure compressor
T_{22}	Temperature downstream water-charge air cooler
$T_{EGR-hot}$	Temperature upstream EGR valve
$T_{EGR-cold}$	Temperature downstream EGR cooler
T_3	Exhaust manifold temperature
T_4	Temperature downstream low-pressure turbine
λ	Oxygen concentration after combustion

Fault classification models

Investigation of supervised learning approaches

A selection of the most common supervised machine learning classification approaches that could be applied to FDI are studied in this section. A performance analysis has been carried out to select the most suitable approach for the specific application, considering model accuracy, training time, and prediction speed. For each group of models, a qualitative description of the underlying algorithm and the main pros and cons is provided [42].

Decision Tree (DT)

Decision Trees are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. It consists of a rooted tree including one root node, internal or test nodes, and decision nodes called leaves. According to a certain function of the input attributes values, each test node divides the instance space into two or more subspaces. Each decision node is assigned to one class representing the most relevant target value. It has the great advantage of being easy to understand and interpret and perfect for visual representation. It is a non-parametric model, meaning that no assumption is required about the distribution of the data (differently from Naïve Bayes and Discriminant Analysis). Moreover, it requires little data preparation and feature selection happens automatically, so that unimportant features will not influence the result. On the other hand, DT learners can be unstable due to small variations of the data and also create overfitting, especially for deep structures, which lead to bad generalization capabilities. This issue can be mitigated by using DTs within an ensemble. Coarse Tree (CT), Medium Tree (MT) and Fine Tree (FT), characterized by increasing number of nodes, have been considered within the performance analysis.

k-Nearest Neighbors (KNN)

KNN is a non-parametric classifier, consisting of evaluating the similarity between data points based on a set of numerical features and a selected metric. A target point is assigned to the appropriate class by a majority vote of its neighbors. It is the simplest classification algorithm to implement with just one parameter to be set (i.e., the number of neighbors k). KNN does not explicitly build a model, but it simply tags the new data "memorizing" the training dataset. Hence, it takes longer time for inference than training and model size grows with the new data incorporated. KNN works well with a small number of input variables, but as the dataset grows efficiency or speed of the algorithm declines very fast. The different types of KNN considered for this analysis include Fine (FKNN), Medium (MKNN), Coarse

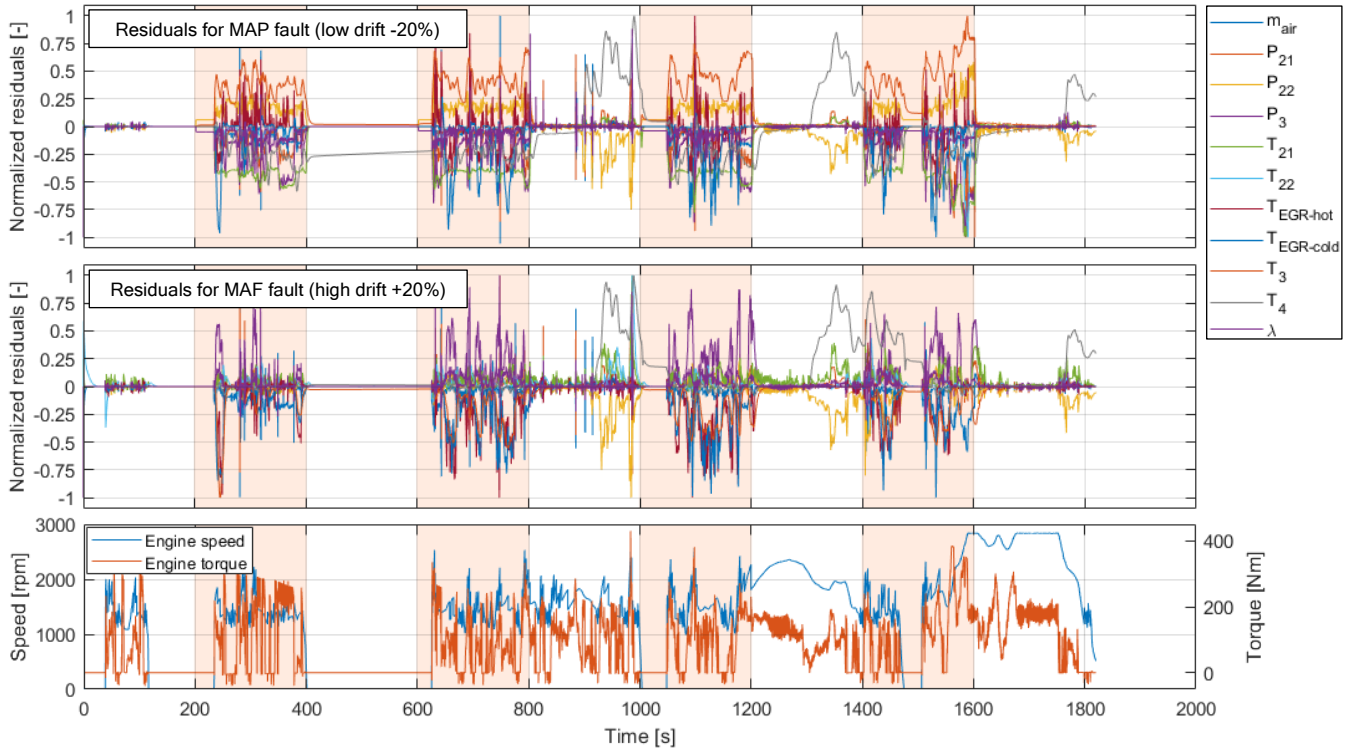


Figure 3. Residuals for different intermittent faults on WLTC driving cycle: MAP low drift at the top, MAF high drift at the bottom. The colored areas identify the intervals in which the fault is present.

(CKNN), Cosine (CosKNN), Cubic (CubKNN) and Weighted KNN (WKNN).

Discriminant Analysis (DA)

DA consists of finding a linear (or quadratic) combination of features that separates two or more classes of objects, assuming that different classes generate data based on different Gaussian distributions with the same covariance matrices. Furthermore, it is claimed that prior probabilities of class membership are known or can be estimated beforehand. The training phase requires significant computational effort to determine the discriminant functions and their parameters. Once completed, classifying any new data could be achieved simply by solving the corresponding discriminant function for each class and by applying the classification rule, so it is simple, fast, and easy-implementable. Accuracy is limited when its assumptions on predictors distribution are not met. Both Linear (LD) and Quadratic Discriminant (QD) have been considered in this analysis.

Support Vector Machines (SVM)

SVM classifies data by finding the best hyperplane that separates all data points of one class from those of the other class by the aid of a kernel function. The input data are plotted in a high-dimensional space (with as many dimensions as the number of features), and the SVM algorithm finds the best boundary that separates the classes This boundary is chosen in such a way that it maximizes the margin, which is the distance between the boundary and the closest data points from each class, also known as support vectors. SVM can treat nonlinear classification problems by applying nonlinear kernel functions (radial basis function (RBF), sigmoid, polynomial, etc.). SVM classifiers perform well in high-dimensional space and have excellent accuracy. Moreover, they require limited memory because they only use a portion of the training data. On the other hand, SVM requires a long training period, hence it is not practical for large datasets, and it is not suitable to handle overlapping classes (e.g., due to the presence of noise in the input signals). Linear (LSVM), Quadratic (QSVM), Cubic (CSVM), Fine Gaussian (FGSVM), Medium Gaussian (MGSVM), and Coarse Gaussian SVM (CGSVM) have been considered.

Naïve Bayes (NB)

Naïve Bayes is a classification algorithm that applies density estimation to the data. The algorithm leverages Bayes theorem, and (naively) assumes that the presence of one feature in a class doesn't affect the presence of another one. Another advantage is the ease of implementation, because it requires a small amount of training data, and it is also robust to isolated noise points and to irrelevant features. It is considered as a fast classification method, and it can handle high-dimensional data efficiently. However, it is not easily comprehensible for human readers, and the assumption that all predictors are independent, rarely happens in real life, decreasing the potentially high accuracy of this algorithm. Two types of NB algorithms have been considered within the present investigation, namely Gaussian (GNB) and Kernel Naïve Bayes (KNB).

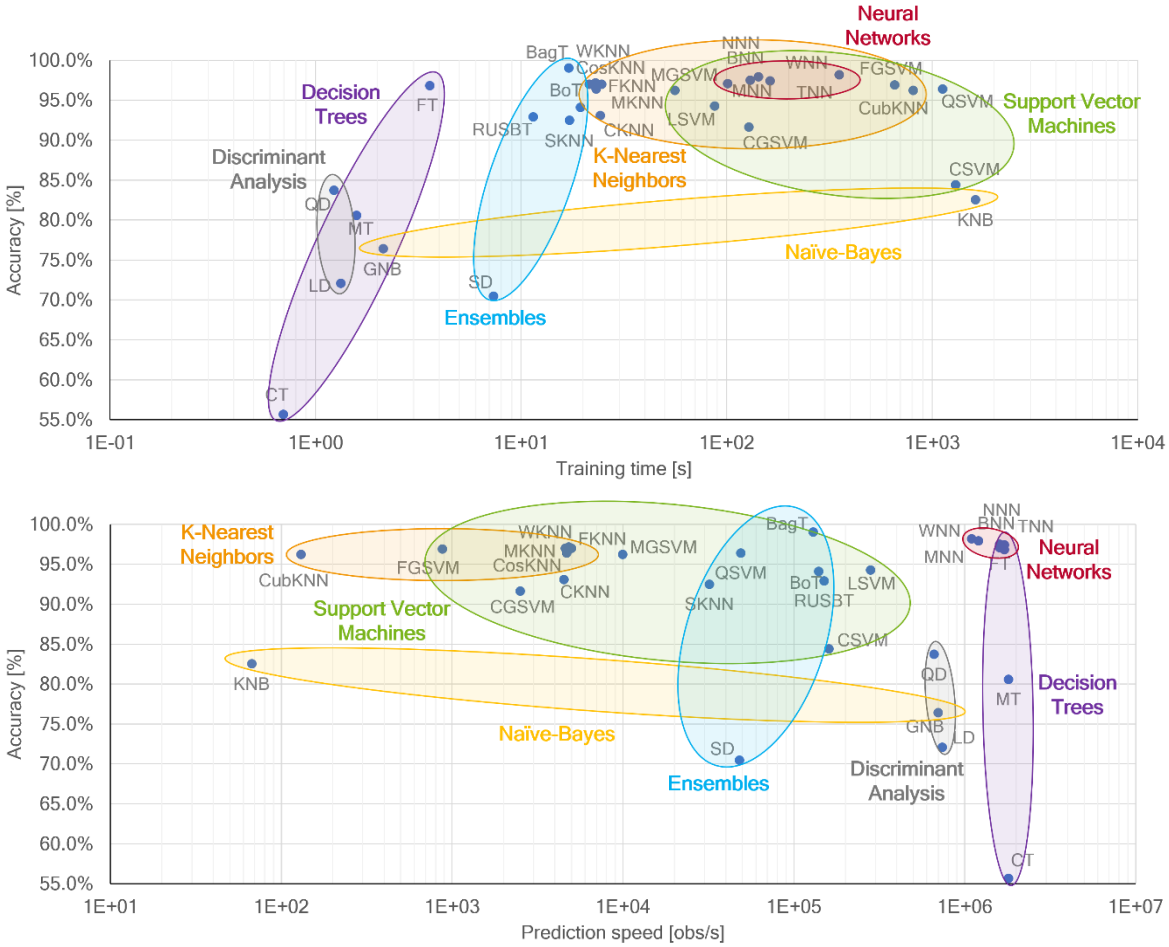


Figure 4. Accuracy, training time, and prediction speed of different families of classification algorithms.

Ensemble Methods

A classification ensemble is a predictive model composed of a weighted combination of multiple classification models. In general, combining multiple classification models increases prediction accuracy. At the same time, a deterioration of interpretability and computational cost is usually observed. Although an unlimited number of ensembles could be developed, the three main classes of ensemble learning methods are: bagging, based on fitting many decision trees on different samples of the same dataset and averaging the predictions; stacking, based on fitting many different model types on the same data and using another model to learn how to best combine the predictions; boosting, based on adding ensemble members sequentially that correct the predictions made by prior models and outputs a weighted average of the predictions. Ensemble methods including Boosted Trees (BoT), Bagged Trees (BagT), Subspace Discriminant (SD), Subspace KNN (SKNN), and RUSBoosted Trees (RUSBT) have been considered for this application.

Artificial Neural Network (ANN)

The standard ANN consists of a set of connected neurons which are organized in an input layer, an output layer, and one or more hidden layers in between. An increase in the number of layers allows for nonlinear calculations, improving the weakness of the perceptron (namely a single-layer NN) linear nature. In FDI applications, multi-layer perceptron (MLP) networks have shown good accuracy and robustness to noise and errors [43]. However, high-capacity networks with complex structures lead to high computational complexity and thereby to slow convergence and potential overfitting during training. To overcome this issue, a large, diversified dataset is required for training [16]. For the analysis presented in this paper, Narrow (NNN), Medium (MNN), Wide NN (WNN), characterized by a single layer with an increasing number of neurons, have been considered, as well as Bi-layered (BNN) and Tri-layered NN (TNN).

Performance analysis

The *Classification Learner App* by Matlab® has been used to train the classification models introduced in the previous paragraphs, in order to choose the best approach for the specific application. The training dataset given as input to the models includes 13 features, namely the 11 residuals previously introduced in addition to engine speed and load request. The data have been generated simulating real driving cycles in both nominal and faulty conditions. Specifically, two different driving cycles in nominal condition and the same driving cycle with five different degrees of severities for each fault (see Table 4), have been simulated, in order to obtain a well-balanced dataset. Depending on the considered fault, the degree of severity is intended as:

- the percentage error between the measurement and the actual physical quantity for MAP and MAF sensors drifts;
- the imposed maximum EGR valve opening, expressed in percentage with respect to the nominal condition, for EGR valve clogging;
- the actual cooling efficiency of the EGR cooler, expressed in percentage compared to the nominal case, for reduced EGR cooler efficiency.

The data from the simulations have been sampled with a frequency of 1 Hz, for a total of around 200000 observations (which would corresponds to 55 hours of real driving) available for the offline training and validation of the models. Each observation is associated with a label identifying a specific fault class. A list of the labels can be found in Table 4, together with the fault cases included in the training dataset. The known responses to the input data are provided to the model as well. As a preliminary step, all the different models have been trained and validated to investigate their performance in terms of prediction speed, expressed in observations per second (obs/s), training time (s), and accuracy, measuring the percentage of correctly predicted instances and calculated as:

Table 4. List of fault classes, related labels, and fault cases included in the training dataset.

Class label	Fault	Severity
NOM	None	-
MAP LOW	Intake manifold pressure sensor low drift	-5%, -10%, -15%, -20%, -25%
MAP HIGH	Intake manifold pressure sensor high drift	+5%, +10%, +15%, +20%, +25%
MAF LOW	MAF sensor low drift	-5%, -10%, -15%, -20%, -25%
MAF HIGH	MAF sensor high drift	+5%, +10%, +15%, +20%, +25%
EGR	Clogged EGR valve	40%, 30%, 20%, 10%, 0%
EGC	Reduced EGR cooler efficiency	90%, 70%, 50%, 20%, 0%

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (1)$$

where TP and TN are the number of true positives and true negatives, respectively (i.e., correct predictions), while FP and FN are the number of false positives and false negatives, respectively (i.e., misclassified observations). Holdout validation, recommended for large datasets, has been applied with 70% of the dataset used for training and 30% held out for validation. The default set of hyperparameters proposed by the *Classification Learner* App has been considered for each model [44]. The results of this analysis are graphically represented in Figure 4. The Bagged Trees (BagT) method, from the Ensemble family, reaches the highest accuracy on validation (98.9%), while the Fine Tree (FT) and the Wide Neural Network (WNN) methods show the highest prediction speed, which makes them interesting in view of real-time application, together with a high prediction accuracy of 96.8% and 98.2%, respectively. On the other hand, Neural Networks require a long training time, which is significantly reduced for the Ensemble methods and even lower for the Decision Trees. Since each of the mentioned approaches shows different advantages/disadvantages in terms of accuracy, training time, or prediction speed, all three methods have been considered for further investigations to evaluate their performance on different testing cases (see section *Results and discussion*).

Models optimization

To maximize the accuracy of the selected classification models, the main hyperparameters have been optimized using Bayesian optimization with a maximum of 100 iterations, which is far above the number of iterations after which an asymptote in the evaluated model accuracy is observed. The optimizable parameters for each model and their range of optimization are summarized in Table 5, together with the optimal hyperparameter configurations resulting from the Bayesian optimization. Figure 5 shows the confusion matrices evaluated on the validation dataset for the three models. The confusion matrix is a useful tool to understand how the currently selected classifier performed in each class, eventually identifying the areas where the

Table 5. Set of optimizable parameters, optimization range, and final configuration resulting from Bayesian optimization of the selected models.

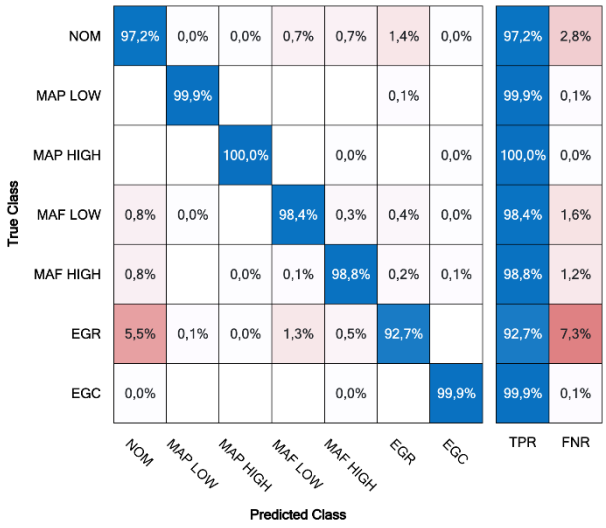
Model	Optimizable hyperparameters	Optimization range	Optimal configuration
Optimizable Tree	Maximum number of splits	[1, max (2, n–1)]	858
	Split criterion	Gini's diversity index, Twoing rule, Maximum deviance reduction	Maximum deviance reduction
Optimizable Ensemble	Ensemble method	AdaBoost, RUSBoost, Bag	AdaBoost
	Maximum number of splits	[1, max (2, n–1)]	655
	Number of learners	[10, 500]	455
	Learning rate	[0.001, 1]	0.948
	Number of predictors to sample	[1, max (2, p)]	All
Optimizable Neural Network	Number of fully connected layers	[1, 3]	2
	Layers size	[1, 300]	43, 173
	Regularization strength	[0.00001/n,100000/n]	4.8454e-11
	Activation function	ReLU, Tanh, None, and Sigmoid.	Tanh

n = number of observations (=217740)
p = number of predictor variables (=13)

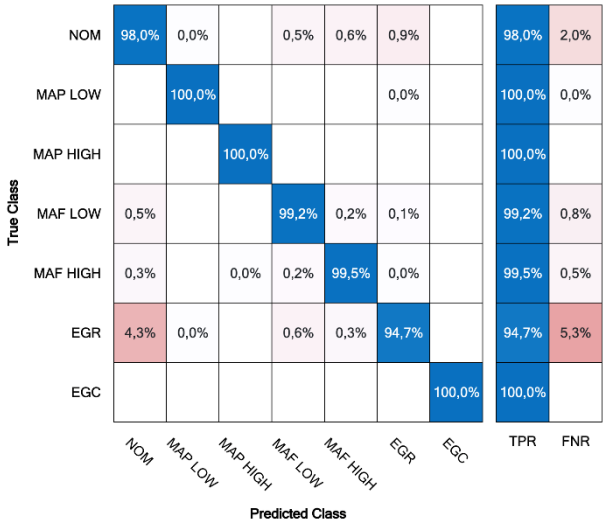
classifier performed poorly. The rows show the true class, and the columns show the predicted class, so that diagonal cells show where the true class and predicted class match. On the right of each confusion matrix, the true positive rates (TPR) column summarizes the proportion of correctly classified observations per each true class, while the false negative rates (FNR) column identifies the proportion of incorrectly classified observations per each true class. The results, expressed in percentage for each fault class, show that the three optimized models can successfully classify the different fault conditions included in the validation dataset. The overall accuracy, calculated as in (1), reaches 99% for the Ensemble model, while it is slightly lower for the Tree and Neural Network models, namely 98.5% and 98.7% respectively. More in detail, all the models can perfectly recognize the faults regarding MAP sensor and EGR cooler, with a TPR very close to or equal to 100%. The most significant misclassification error involves the EGR valve clogging, with a maximum FNR above 7% for the Tree model, which is mainly misclassified as a nominal condition. For each model, no significant number of cases in which a fault is incorrectly classified as another fault (excluding nominal condition) is observed, thus minimizing the risk of incorrect fault detection. Given the very good overall performance, all the classification models have been considered for further investigations to assess their accuracy and robustness on different test cases and to evaluate their real-time capability.

Simulink model implementation and test cases

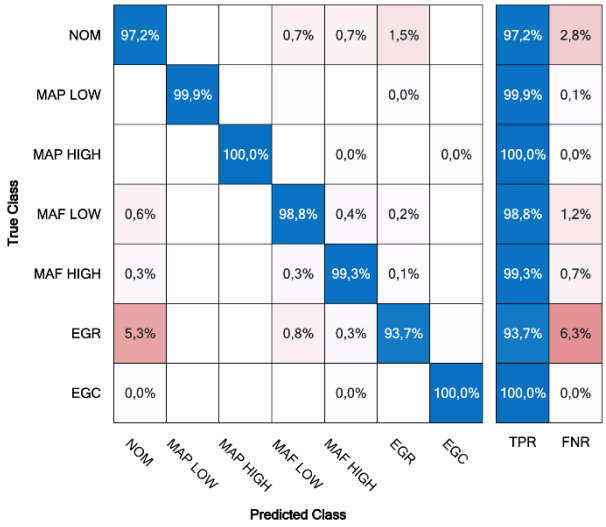
The three validated models have been integrated into the Simulink vehicle model by means of the dedicated blocks from the *Statistics and Machine Learning Toolbox*, to be directly tested on different simulated driving cycles and fault conditions. The considered driving cycles are known homologating driving cycles, including WLTC (Worldwide Harmonized Light Vehicles Test Cycle, [45]), SFTP-US06 (Supplemental Federal Test Procedure, [46]), and FTP-72 (Federal Test Procedure,[47]), which differ from each other in terms of length, maximum speed, and harsh acceleration phases. An RDE-compliant



(a)



(b)



(c)

Figure 5. Confusion matrices of the optimized models evaluated on the validation dataset: Optimized Tree (a), Optimized Ensemble (b), Optimized Neural Network (c).

driving cycle, different from the one considered for the training phase, has been included as well to complete the testing framework. As a first step, the models have been tested on single-fault scenarios, also including degrees of fault severity that were not included in the training dataset, in order to evaluate the interpolation and extrapolation capabilities of the developed models. Then, combinations of two simultaneous faults have been considered, to evaluate the robustness of the models in such testing conditions, despite being trained only on single-fault scenarios. All the driving cycles have been run considering low battery initial state of charge, in order to force the engine start within the first seconds of the driving cycle. The same ambient temperature, set to 25°C, has been considered for all the test cases, since the effect of variable environmental conditions on fault detection accuracy has not been investigated in this work.

Real-time implementation

As a final step, the developed classification models have been implemented on a real-time hardware to evaluate the feasibility of a real on-board implementation in terms of the requested computational load. The hardware chosen for this application is a Raspberry Pi 4 computer [48], coupled with a PiCAN2 board [49] to manage the automotive CAN-bus standard communication. The software has been developed in Simulink by means of the MATLAB Support Package for Raspberry Pi Hardware [50]. Thanks to this architecture, the Raspberry will be able to communicate via CAN with vehicle control units, thus receiving all the signals needed by the classification model. Taking advantage of information that is easily available on a standard vehicle, the proposed solution based on a Raspberry Pi 4 ensures high modularity at low cost, without the need of modifying the already existing vehicle software.

Results and discussion

The results in terms of classification accuracy on different driving cycles and fault conditions are summarized in Table 6 and Table 7. More in detail, Table 6 shows several tests where a single fault has been considered for each simulated driving cycle, while combinations of two faults are shown in Table 7. In view of a real-time on-board application, the raw prediction given by the classification model has been filtered by means of a moving average. The length of the sliding window has been set to 10 seconds, a good trade-off between a too large sliding window, which could lose some relevant information, and a too small one, which could be too sensitive to instantaneous model inaccuracies. This means that, for each time instant, the fault detection model outputs the label of the fault class which has been most frequently observed in the past 10 seconds, i.e. the one with the highest probability. The detected classes shown in Table 6 consider just the averaged prediction signal, so that negligible, but still unavoidable, misclassification errors are not considered. The related accuracy corresponds to the TPR calculated on the raw prediction signal, which is a useful parameter to evaluate the effective performance of each model for the different test cases. Based on the results, all the models are able to detect the true fault class, with no misclassification errors if the mentioned moving average is applied to the raw prediction signal.

Table 6. Detected classes (after prediction averaging) and related accuracy (calculated on raw prediction) for the three classification models on different single-fault test cases.

Test Case			Detected Classes Accuracy [%]					
Driving Cycle	True fault class	Fault severity	Optimized Tree		Optimized Ensemble		Optimized Neural Network	
WLTC	NOM	-	NOM 94.1%		NOM 97.0%		NOM 96.7%	
	MAF HIGH	15%	MAF HIGH 99.2%		MAF HIGH 99.7%		MAF HIGH 99.5%	
	EGR	30%	EGR 50.2%	NOM 47.1%	EGR 49.3%	NOM 49.2%	EGR 49.9%	NOM 48.3%
	EGC	80%	EGC 99.9%		EGC 100%		EGC 100%	
	MAP LOW	-30%	MAP LOW 100%		MAP LOW 100%		MAP LOW 100%	
US06	NOM	-	NOM 87.8%		NOM 93.1%		NOM 92.7%	
	MAF LOW	-10%	MAF LOW 95.1%		MAF LOW 97.7%		MAF LOW 95.2%	
	EGC	70%	EGC 99.8%		EGC 99.9%		EGC 99.9%	
	MAP LOW	-20%	MAP LOW 99.9%		MAP LOW 100%		MAP LOW 99.9%	
FTP-72	MAF HIGH	+10%	MAF HIGH 98.7%		MAF HIGH 99.2%		MAF HIGH 99.0%	
	MAP HIGH	+20%	MAP HIGH 100%		MAP HIGH 100%		MAP HIGH 100%	
	EGR	40%	EGR 13.9%	NOM 79.9%	EGR 15.5%	NOM 81.9%	EGR 15.4%	NOM 81.5%
	EGR	10%	EGR 87.4%	NOM 12.2%	EGR 82.2%	NOM 11.8%	EGR 88.6%	NOM 11.3%
RDE	EGC	40%	EGC 100%		EGC 100%		EGC 100%	
	MAP LOW	-40%	MAP LOW 100%		MAP LOW 100%		MAP LOW 99.5%	
	MAF HIGH	+40%	MAF HIGH 98.7%		MAF HIGH 99.6%		MAF HIGH 99.6%	

Table 7. Detected classes (after prediction averaging) and related accuracy (calculated on raw prediction) for the three classification models on different double-fault test cases.

Test Case					Detected Classes Accuracy [%]										
Driving Cycle	True fault classes		Fault severity		Optimized Tree			Optimized Ensemble				Optimized Neural Network			
WLTC	MAF LOW	MAP LOW	-15%	-15%	MAP LOW 99.9%			MAP LOW 99.9%				MAP LOW 99.9%			
	MAF LOW	MAP LOW	-25%	-5%	MAF LOW 14.2%	MAP LOW 70.4 %	EGR 15.3%	NOM 1.8%	MAF LOW 12.1%	MAP LOW 80.1%	EGR 80.1%	MAF LOW 10.4%	MAP LOW 89.6%		
	MAP LOW	EGC	-10%	50%	MAP LOW 19.1%	EGC 78.8%		MAP LOW 21.9%		EGC 78.1%		MAP LOW 13.2%	EGC 86.8%		
	MAF LOW	EGR	-15%	30%	MAF LOW 86.8%	EGR 11.3%		MAF LOW 87.5%		EGR 12.3%		MAF LOW 97.6%	EGR 2.4%		
FTP-72	MAF HIGH	MAP HIGH	+15%	+15%	MAP HIGH 99.9%			MAP HIGH 99.9%				MAP HIGH 100%			
	MAF LOW	MAP HIGH	-15	+15%	MAP HIGH 99.9%			MAP HIGH 99.9%				MAP HIGH 99.9%			
	MAP HIGH	EGC	+10%	70%	MAP HIGH 21.3%	EGC 78.7%		MAP HIGH 8.0%		EGC 92.0%		MAP HIGH 21.2%	EGC 78.6%		
	MAF HIGH	EGR	+15%	20%	NOM 4.7%	MAF HIGH 6.0 %	EGR 87.8%	NOM 4.9%	MAF HIGH 5.1%	EGR 90.3%		NOM 8.2%	MAF HIGH 10.7%	EGR 79.1%	

The very high accuracy, close to or equal to 100%, on the test cases involving EGC and MAP sensor faults, confirms what already stated in the validation phase. Compared to the Neural Network and Ensemble models, a slightly lower accuracy is observed for the Tree model for both nominal and MAF sensor fault cases, which is still far above 90%, except for the US06 nominal case. A brief consideration must be added with regard to the cases involving EGR valve clogging: this type of fault imposes a limitation on the EGR valve opening which depends on the fault severity; thus, differently from the other considered faults, which affect all the engine operating points, only in the operating points where the requested EGR valve position would have been higher than the threshold imposed by the clogging there will be an effect due to the fault. For this reason, the fault can be effectively detected just when the requested EGR valve position is above the maximum opening imposed by the fault, which, on the same driving cycle, happens more or less frequently depending on the fault severity. As a result, the classification models correctly recognize both EGR valve fault and nominal conditions when EGR valve clogging is considered, and the percentage of observations classified as EGR valve fault increases if a lower maximum opening of the EGR valve is allowed. Overall, all the developed models show a robust and reliable behavior on single fault classification on different driving cycles, including interpolation and extrapolation capabilities on selected fault conditions which were not included in the training dataset (i.e. EGC 40%, EGC 80%, MAP LOW -40%, MAP LOW -30%, MAF HIGH +40%). Moreover, the Ensemble model is confirmed to be the most accurate, followed by the Neural Network model, which shows similar performance in most of the test cases.

Focusing on Table 7, the detected fault classes in case of combinations of two faults, extracted from the averaged signal, are shown. Each detected class corresponds to a percentage of observations, calculated on the raw prediction, classified as the considered class. In the cases in which a combination of MAF and MAP sensors drift with the same absolute value is considered (e.g. MAF LOW -15% + MAP LOW -15%), the models are not able to detect both the faults, but just the MAP sensor fault is detected, for almost 100% of the observations. On the other hand, if a lower severity is considered for the MAP sensor fault with respect to the MAF sensor fault (e.g. MAF LOW -25% + MAP LOW -5%), the models are able to detect also the MAF sensor fault. However, only the Neural Network is capable of correctly detecting the considered fault combination, while both Ensemble and Tree models show significant false negatives corresponding to EGR valve fault. In the remaining test cases, combining MAP or MAF sensors faults with EGC or EGR valve faults, all three models show good behavior: they are able to correctly detect both the fault classes, with no false class detection, except for one case in which also nominal class is detected with a low percentage, which anyway could be accepted on a real application since not leading to wrong fault detection. Based on these results, when a combination of two faults is considered, the Neural Network classification model shows the preferred behavior: in most of the considered tests it is able to correctly detect both faults, and no false fault detections are observed.

In the following paragraphs, three of the considered test cases are deeply analyzed to show how the models behave.

Case 1

A WLTC driving cycle with clogged EGR valve, leading to 30% maximum valve opening, is considered in Figure 6. Starting from the top to the bottom of the figure, engine speed and load are shown, followed by the cumulated engine-out NO_x emission: this plot highlights the effect of the considered fault in terms of resulting pollutant emissions (blue line) with respect to the nominal case (red line); the dotted line corresponds to the prediction made by the physics-based model inside the engine controller, which deviates from the actual emission calculated by the engine physical model when a fault is considered. The second plot from the bottom compares the requested EGR valve position (red line) and the actual one (blue line), which is affected by the induced fault imposing a maximum valve opening (dotted line). The bottom plot shows the instantaneous fault class predicted in each simulation time step, i.e. every 10 ms, by the optimized Tree model (grey line), together with the fault classes effectively detected when a moving average with a window length of 10 seconds is applied (red line). The phases in which the engine is switched off have been excluded from the prediction and identified

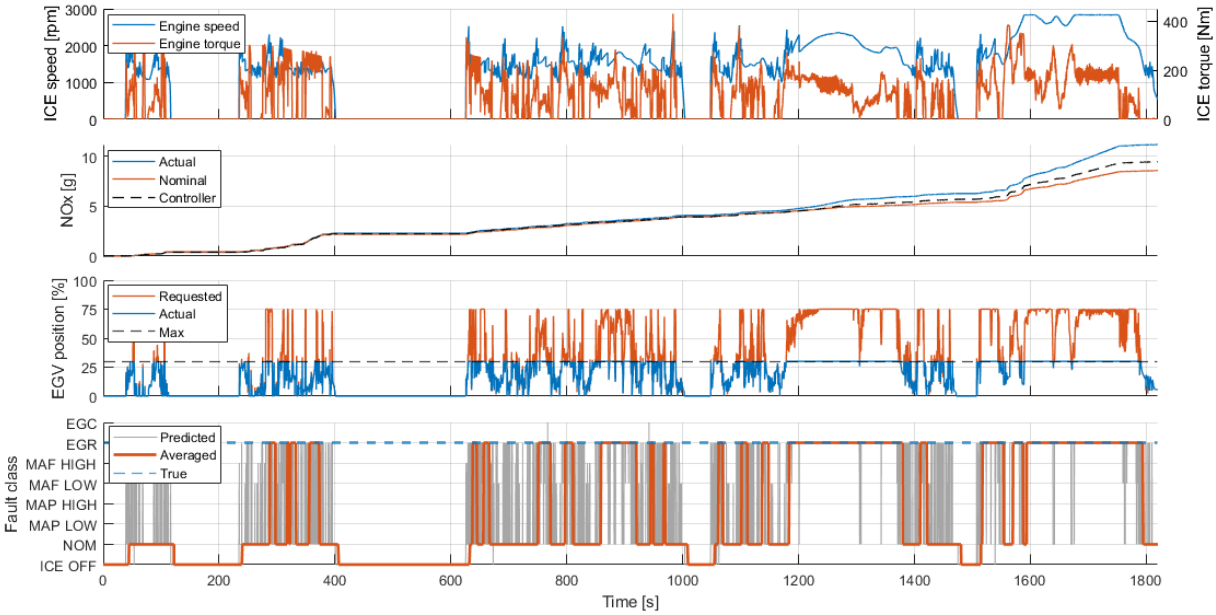


Figure 6. Case 1: test of optimized Tree model on WLTC driving cycle with clogged EGR valve (30% maximum valve opening).

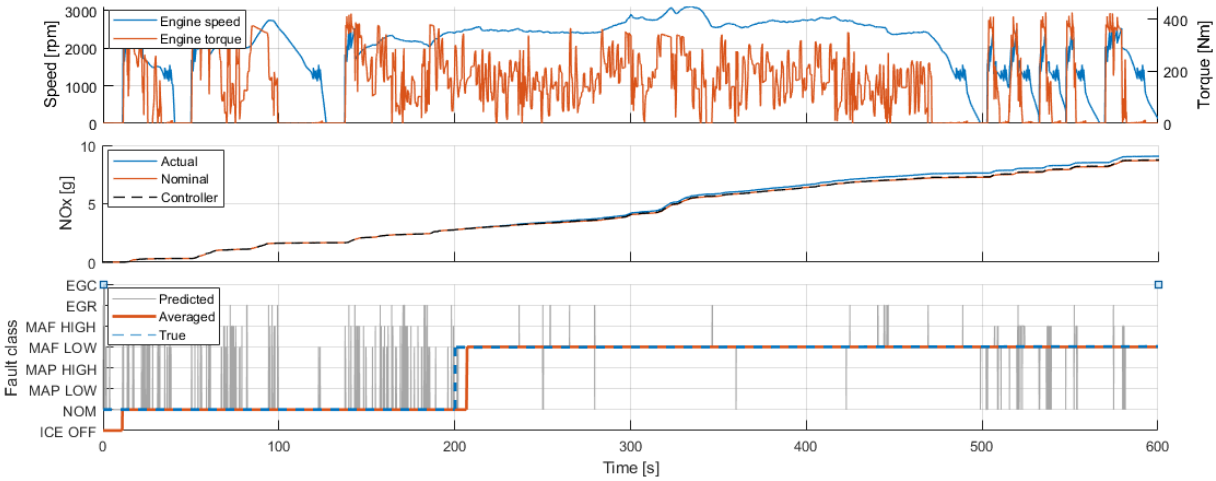


Figure 7. Case 2: test of optimized Ensemble model on FTP-72 driving cycle with 10% MAF sensor low drift starting from 200 s.

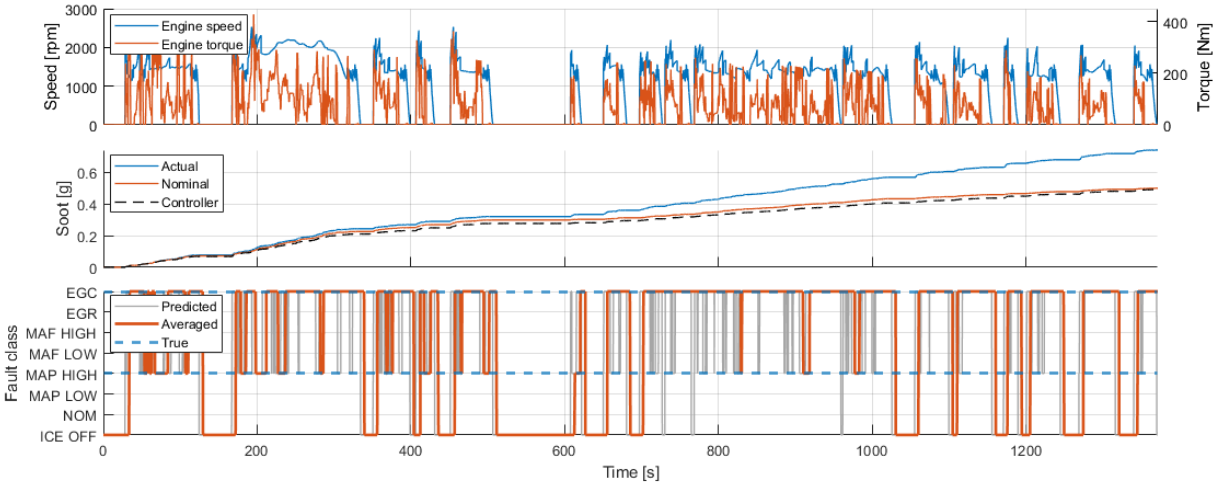


Figure 8. Case 3: test of optimized Neural Network on FTP-72 driving cycle with combined faults, i.e. 10% MAP sensor high drift and 70% EGR cooler efficiency.

under the label “ICE OFF”. The true fault class, according to the considered test case, is highlighted by the blue dotted line. As already introduced in the previous paragraphs, the classification model is able to detect the fault when actually present: focusing on the two plots at the bottom of Figure 6, it can be observed that the model recognizes the presence of the fault each time that the actual EGR valve opening is limited with respect to the requested one, namely when the engine operating condition differs from the nominal case.

Case 2

A US06 driving cycle with a MAF sensor low drift of -10%, is considered in Figure 7. The fault has been introduced in the model after 200 s from the beginning of the cycle, to better highlight the reaction of the classification model to the fault introduction. As in the previous case, engine speed and torque are shown, together with the resulting NO_x emissions, which start deviating from the nominal condition when the fault is introduced, even if only a slight increase of emission is observed due to the limited length of the cycle. In the bottom plot, the instantaneous fault class predicted by the optimized Ensemble model (grey line), together with the fault classes effectively detected applying the moving average (red line), is shown. The true fault class, corresponding to the blue dotted line, highlights the introduction of the fault. As expected, the classification model is able to detect the fault when first introduced, with a slight delay depending on the moving average, which however proves to be effective in filtering eventual signal spikes due to negligible misclassification errors.

Case 3

The results on FTP-72 driving cycle with a combination of two faults, namely 10% MAP sensor high drift and 70% EGR cooler efficiency, are shown in Figure 8. The two considered faults have opposite effects on NO_x emissions, as previously analyzed (Table 2), while overlapping effects on soot emissions. In this case, being the most relevant result, the cumulated soot emission is shown, which significantly deviates from the nominal condition due to this fault combination. As in the previous cases, in the bottom plot, the instantaneous fault class predicted by the optimized Neural Network model (grey line), together with the fault classes effectively detected applying the moving average (red line), is shown. The true fault classes, corresponding to the blue dotted lines, highlight the introduction of two different faults. Even in the case of fault combination, the classification model is able to correctly detect both faults, even if the EGC class is predominant, as confirmed by the results in Table 7.

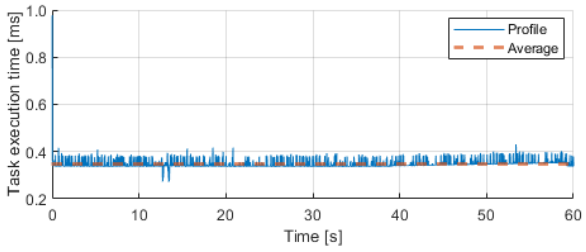
Evaluation of real-time capability

As already introduced in the section *Materials and Methods,* the developed classification models have been deployed on a Raspberry Pi to verify their real-time capability and related computational load.

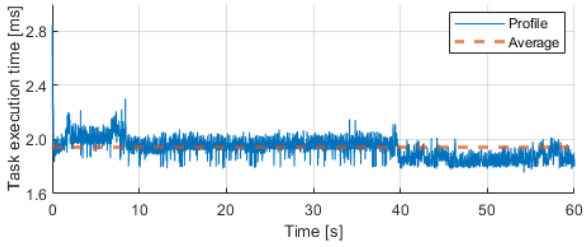
Table 8. Maximum and average execution time and CPU utilization for the three classification models running on Raspberry Pi.

Model	Task execution time [ms]		CPU utilization [%]	
	Max	Avg	Max	Avg
Tree	1.0	0.3	4.8	1.7
Ensemble	2.8	1.9	14.2	9.7
Neural Network	1.2	0.5	5.9	2.3

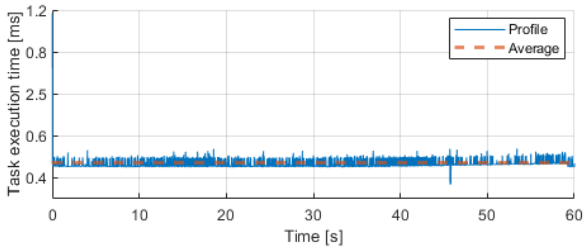
The *Code Profile Analyzer* tool by MATLAB® has been used to monitor and analyze the execution time profile of the real-time applications running on the Raspberry (Figure 9). Considering a time-step of 10 ms, which is a commonly set value for standard control units, the results in Table 8 show that all three models are real-time capable, with a limited computational load that reaches a maximum value of 14.2% of CPU utilization in the case of the Ensemble model. As expected from the preliminary performance analysis introduced in the section *Materials and Methods,* the Tree model shows the shortest task execution time, with an average of 0.3 ms and a maximum of 1 ms, followed by the Neural Network model, which shows a similar behavior with 0.5 ms and 1.2 ms of average and maximum task execution time respectively. The Ensemble model shows a significantly higher task execution time compared to the other models, with an average value of 1.9 ms and a maximum of 2.8 ms, which however is far within the considered application time-step of 10 ms.



(a)



(b)



(c)

Figure 9. Task execution time profile for Tree (a), Ensemble (b), and Neural Network (c) real-time applications.

Conclusions and future developments

The present activity has been focused on the investigation of supervised machine learning classification approaches to detect and identify emission-relevant engine faults based on available on-board signals. For this purpose, a comprehensive 0-D control-oriented vehicle model has been developed in Simulink® and used to generate data to train and test the classification models. As a first step, several simulations have been performed introducing different faults, involving EGR valve, EGR cooler, MAF and MAP sensors, to evaluate their effect on pollutant emissions. Then, a full dataset, obtained simulating real driving cycles in both nominal and selected faulty conditions, has been used to train and validate different classifiers, in

order to evaluate their performance in terms of accuracy, prediction speed, and requested training time. As a result of this preliminary analysis, three different types of classifiers have been selected, namely Decision Tree, Ensemble, and Neural Network-based classifiers, which have been further improved through Bayesian optimization of the main hyperparameters. The optimized models have been integrated into the Simulink® vehicle model and tested on different driving cycles and fault conditions to evaluate their accuracy, robustness, and interpolation/extrapolation capabilities. As a last step, the developed models have been flashed on a real-time hardware to verify their real-time capability and related computational load in view of real-world application. The results discussed in section *Results and discussion,* highlighted that:

- All three optimized classification models are able to correctly detect and identify single faults with high overall accuracy, regardless of the considered driving cycle or fault severity.
- The Ensemble model has proven to be the most accurate, followed by the Neural Network model, which shows similar performance in most of the test cases.
- No false fault detections are observed for the considered test cases, provided that a moving average is applied to the raw prediction signal to neglect instantaneous misclassification errors.
- When a combination of two faults is considered, the Neural Network classification model shows the best behavior: in most of the considered tests it is able to correctly detect both the induced faults without false fault detections, even if trained just on single-fault conditions.
- All three optimized models are real-time capable, with a maximum task execution time of 2.8 ms for the Ensemble application.
- The Neural Network-based classifier provides the best overall performance if both accuracy on single and multiple fault detection and computational load are considered.

Despite these promising results obtained in simulation, which show the potential of supervised machine learning approaches applied to on-board engine fault detection, the robustness and reliability of the proposed approach must be confirmed by real-world application, where random errors, for example, due to noise and sensor inaccuracies, may strongly affect the final result. Moreover, additional known engine faults should be included in the analysis, and a parallel approach handling unknown fault conditions, simply classifying them as non-nominal, should be developed in view of real-world application. Nonetheless, the proposed methodology can be definitely considered a successful proof of concept to be used as a starting point for further developments. Moreover, the proposed real-time application offers a ready-to-use and flexible tool to perform HiL (Hardware-in-the-Loop) and on-board testing of the developed fault classification models. In this respect, future research will be focused on the improvement of the presented fault classification approach to overcome its actual limits, followed by on-board testing to assess its performance in real-world application.

References

[1] I. Samy and D.-W. Gu, “Fault Detection and Isolation (FDI),” in *Fault Detection and Flight Data Measurement: Demonstrated on Unmanned Air Vehicles Using Neural Networks*, I. Samy and D.-W. Gu, Eds., in Lecture Notes in Control and Information Sciences. , Berlin, Heidelberg: Springer, 2011, pp. 5–17. doi: 10.1007/978-3-642-24052-2_2.

[2] S. X. Ding, *Model-Based Fault Diagnosis Techniques: Design Schemes, Algorithms and Tools*. in Advances in Industrial Control. London: Springer, 2013. doi: 10.1007/978-1-4471-4799-2.

[3] C. Sankavaram, A. Kodali, K. R. Pattipati, and S. Singh, “Incremental Classifiers for Data-Driven Fault Diagnosis Applied to Automotive Systems,” *IEEE Access*, vol. 3, pp. 407–419, 2015, doi: 10.1109/ACCESS.2015.2422833.

[4] M. Sayed-Mouchaweh, Ed., *Fault Diagnosis of Hybrid Dynamic and Complex Systems*. Cham: Springer International Publishing, 2018. doi: 10.1007/978-3-319-74014-0.

[5] K. T. P. Nguyen, K. Medjaher, and D. T. Tran, “A review of artificial intelligence methods for engineering prognostics and health management with implementation guidelines,” *Artif Intell Rev*, vol. 56, no. 4, pp. 3659–3709, Apr. 2023, doi: 10.1007/s10462-022-10260-y.

[6] A. R. Sahu, S. K. Palei, and A. Mishra, “Data-driven fault diagnosis approaches for industrial equipment: A review,” *Expert Systems*, vol. n/a, no. n/a, p. e13360, May 2023, doi: 10.1111/exsy.13360.

[7] M. Hajji *et al.*, “Multivariate feature extraction based supervised machine learning for fault detection and diagnosis in photovoltaic systems,” *European Journal of Control*, vol. 59, pp. 313–321, May 2021, doi: 10.1016/j.ejcon.2020.03.004.

[8] B. Li and Y.-P. Zhao, “Multi-label learning using label-specific features for simultaneous fault diagnosis of aircraft engine,” *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 236, no. 10, pp. 2057–2073, 2022, doi: 10.1177/09544100211049935.

[9] R. Liu, B. Yang, E. Zio, and X. Chen, “Artificial intelligence for fault diagnosis of rotating machinery: A review,” *Mechanical Systems and Signal Processing*, vol. 108, pp. 33–47, Aug. 2018, doi: 10.1016/j.ymssp.2018.02.016.

[10] H. El hadraoui, M. Zegrari, A. El Maghraoui, O. Laayati, E. Sabani, and A. Chebak, “Data-driven Diagnostics for Electric Traction Systems: A Study of Induction Motor,” in *IEEE EUROCON 2023 - 20th International Conference on Smart Technologies*, Jul. 2023, pp. 626–631. doi: 10.1109/EUROCON56442.2023.10199047.

[11] R. Argawal, D. Kalel, M. Harshit, A. D. Domnic, and R. R. Singh, “Sensor Fault Detection using Machine Learning Technique for Automobile Drive Applications,” in *2021 National Power Electronics Conference (NPEC)*, Dec. 2021, pp. 1–6. doi: 10.1109/NPEC52100.2021.9672546.

[12] Andre A. Silva, Ali M. Bazzi, and Shalabh Gupta, “Fault diagnosis in electric drives using machine learning approaches,” presented at the 2013 International Electric Machines & Drives Conference, IEEE, 2013. Accessed: Sep. 26, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/6556173

[13] C. Hu, J. Wu, C. Sun, R. Yan, and X. Chen, “Interinstance and Intratemporal Self-Supervised Learning With Few Labeled Data for Fault Diagnosis,” *IEEE Transactions on Industrial Informatics*, vol. 19, no. 5, pp. 6502–6512, May 2023, doi: 10.1109/TII.2022.3183601.

[14] Y. Ji *et al.*, “A Study on the Anomaly Detection of Engine Clutch Engagement/Disengagement Using Machine Learning for Transmission Mounted Electric Drive Type Hybrid Electric Vehicles,” *Applied Sciences*, vol. 11, no. 21, Art. no. 21, Jan. 2021, doi: 10.3390/app112110187.

[15] J. A. Carino *et al.*, “Fault Detection and Identification Methodology Under an Incremental Learning Framework Applied to Industrial Machinery,” *IEEE Access*, vol. 6, pp. 49755–49766, 2018, doi: 10.1109/ACCESS.2018.2868430.
[16] M. Saied, H. Attieh, H. Mazeh, H. Shraim, and C. Francis, “Supervised Learning Classification Applications in Fault Detection and Diagnosis: An Overview of Implementations in Unmanned Aerial Systems,” *SAE Int. J. Aerosp.*, vol. 16, no. 1, pp. 01-16-01–0004, Aug. 2022, doi: 10.4271/01-16-01-0004.
[17] G. Hu, T. Zhou, and Q. Liu, “Data-Driven Machine Learning for Fault Detection and Diagnosis in Nuclear Power Plants: A Review,” *Frontiers in Energy Research*, vol. 9, 2021, Accessed: Sep. 26, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fenrg.2021.663296
[18] M. Cheliotis, I. Lazakis, and G. Theotokatos, “Machine learning and data-driven fault detection for ship systems operations,” *Ocean Engineering*, vol. 216, p. 107968, Nov. 2020, doi: 10.1016/j.oceaneng.2020.107968.
[19] G. Tsaganos, N. Nikitakos, D. Dalaklis, A. I. Ölcer, and D. Papachristos, “Machine learning algorithms in shipping: improving engine fault detection and diagnosis via ensemble methods,” *WMU J Marit Affairs*, vol. 19, no. 1, pp. 51–72, Mar. 2020, doi: 10.1007/s13437-019-00192-w.
[20] Z. Samaras, S. Hausberger, and D. G. Mellios, “Preliminary findings on possible Euro 7 emission limits for LD and HD vehicles,” presented at the Online AGVES Meeting, Oct. 27, 2020. [Online]. Available: https://circabc.europa.eu/
[21] *Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on type-approval of motor vehicles and engines and of systems, components and separate technical units intended for such vehicles, with respect to their emissions and battery durability (Euro 7) and repealing Regulations (EC) No 715/2007 and (EC) No 595/2009*. 2022. Accessed: Sep. 28, 2023. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52022PC0586
[22] V. Müller, H. Pieta, J. Schaub, M. Ehrly, and T. Körfer, “On-Board Monitoring to meet upcoming EU-7 emission standards – Squaring the circle between effectiveness and robust realization,” *Transportation Engineering*, vol. 10, p. 100138, Dec. 2022, doi: 10.1016/j.treng.2022.100138.
[23] A. Bingamil, I. Alsyouf, and A. Cheaitou, “Condition monitoring technologies, parameters and data processing techniques for fault detection of internal combustion engines: A literature review,” in *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, Nov. 2017, pp. 1–5. doi: 10.1109/ICECTA.2017.8252038.
[24] Y. S. Wang, N. N. Liu, H. Guo, and X. L. Wang, “An engine-fault-diagnosis system based on sound intensity analysis and wavelet packet pre-processing neural network,” *Engineering Applications of Artificial Intelligence*, vol. 94, p. 103765, Sep. 2020, doi: 10.1016/j.engappai.2020.103765.
[25] R. Ahmed, M. El Sayed, S. A. Gadsden, J. Tjong, and S. Habibi, “Automotive Internal-Combustion-Engine Fault Detection and Classification Using Artificial Neural Network Techniques,” *IEEE Transactions on Vehicular Technology*, vol. 64, no. 1, pp. 21–33, Jan. 2015, doi: 10.1109/TVT.2014.2317736.
[26] M. Khajavi, S. Nasiri, and A. Eslami, “Combined fault detection and classification of internal combustion engine using neural network,” *Journal of Vibroengineering*, vol. 16, pp. 3912–3921, Dec. 2014.
[27] L. T. Francis, V. E. Pierozan, G. Gracioli, and G. M. de Araujo, “Data-driven Anomaly Detection of Engine Knock based on Automotive ECU,” in *2022 XII Brazilian Symposium on Computing Systems Engineering (SBESC)*, Nov. 2022, pp. 1–8. doi: 10.1109/SBESC56799.2022.9965059.
[28] P. Wolf, A. Mrowca, T. T. Nguyen, B. Bäker, and S. Günnemann, “Pre-ignition Detection Using Deep Neural Networks: A Step Towards Data-driven Automotive Diagnostics,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Nov. 2018, pp. 176–183. doi: 10.1109/ITSC.2018.8569908.
[29] S. Zhu, M. K. Tan, R. K. Y. Chin, B. L. Chua, X. Hao, and K. T. K. Teo, “Engine Fault Diagnosis using Probabilistic Neural Network,” in *2021 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, Sep. 2021, pp. 1–6. doi: 10.1109/IICAIET51634.2021.9573654.
[30] P. K. Wong, J. Zhong, Z. Yang, and C. M. Vong, “Sparse Bayesian extreme learning committee machine for engine simultaneous fault diagnosis,” *Neurocomputing*, vol. 174, pp. 331–343, Jan. 2016, doi: 10.1016/j.neucom.2015.02.097.
[31] M. E. Mumcuoglu *et al.*, “Fuel consumption classification for heavy-duty vehicles: a novel approach to identifying driver behavior and system anomalies,” in *2023 AEIT International Conference on Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, Jul. 2023, pp. 1–6. doi: 10.23919/AEITAUTOMOTIVE58986.2023.10217234.
[32] M. Youssef and H. Ibrahim, “Automotive engine fault detection and isolation using LSTM for model-based residual sequence classification,” in *2021 18th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)*, Nov. 2021, pp. 1–6. doi: 10.1109/CCE53527.2021.9633053.
[33] D. Jung and C. Sundström, “A Combined Data-Driven and Model-Based Residual Selection Algorithm for Fault Detection and Isolation,” *IEEE Transactions on Control Systems Technology*, vol. 27, no. 2, pp. 616–630, Mar. 2019, doi: 10.1109/TCST.2017.2773514.
[34] M. Cervantes-Bobadilla *et al.*, “Multiple fault detection and isolation using artificial neural networks in sensors of an internal combustion engine,” *Engineering Applications of Artificial Intelligence*, vol. 117, p. 105524, Jan. 2023, doi: 10.1016/j.engappai.2022.105524.
[35] A. Theissler, “Detecting known and unknown faults in automotive systems using ensemble-based anomaly detection,” *Knowledge-Based Systems*, vol. 123, pp. 163–173, May 2017, doi: 10.1016/j.knosys.2017.02.023.
[36] G. Di Pierro, “Development of an integrated experimental and numerical methodology for the performance analysis of multiple hybrid electric architectures over different driving cycles,” Doctoral Dissertation, Politecnico di Torino, Turin, Italy, 2020. [Online]. Available: https://iris.polito.it/handle/11583/2843980?mode=full.13517#.YsXJeHZBxaQ
[37] L. Guzzella and C. H. Onder, *Introduction to Modeling and Control of Internal Combustion Engine Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-10775-7.
[38] S.-Y. Lee *et al.*, *X-in-the-Loop-basierte Kalibrierung: HiL-Simulation eines virtuellen Dieselantriebsstrangs*. 2017.
[39] S.-Y. Lee *et al.*, “Scalable Mean Value Modeling for Real-Time Engine Simulations with Improved Consistency and Adaptability,” SAE International, Warrendale, PA, SAE Technical Paper 2019-01–0195, Apr. 2019. doi: 10.4271/2019-01-0195.

[40] D. Blanco-Rodriguez, G. Vagnoni, S. Aktas, and J. Schaub, “Model-based Tool for the Efficient Calibration of Modern Diesel Powertrains,” *MTZ Worldw*, vol. 77, no. 10, pp. 54–59, Oct. 2016, doi: 10.1007/s38313-016-0103-5.
[41] S. Canè, L. Brunelli, S. Gallian, A. Perazzo, A. Brusa, and N. Cavina, “Performance assessment of a predictive pre-heating strategy for a hybrid electric vehicle equipped with an electrically heated catalyst,” *Applied Thermal Engineering*, vol. 219, p. 119341, Jan. 2023, doi: 10.1016/j.applthermaleng.2022.119341.
[42] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. in Springer Series in Statistics. New York, NY: Springer, 2009. doi: 10.1007/978-0-387-84858-7.
[43] A. A. A. Mohd Amiruddin, H. Zabiri, S. A. A. Taqvi, and L. D. Tufa, “Neural network applications in fault diagnosis and detection: an overview of implementations in engineering-related systems,” *Neural Comput & Applic*, vol. 32, no. 2, pp. 447–472, Jan. 2020, doi: 10.1007/s00521-018-3911-5.
[44] “Classification Learner App - MATLAB & Simulink - MathWorks Italia.” Accessed: Oct. 13, 2023. [Online]. Available: https://it.mathworks.com/help/stats/classification-learner-app.html?s_tid=CRUX_lftnav
[45] “Emission Test Cycles: WLTC.” Accessed: Oct. 16, 2023. [Online]. Available: https://dieselnet.com/standards/cycles/wltp.php
[46] “Emission Test Cycles: SFTP-US06.” Accessed: Oct. 16, 2023. [Online]. Available: https://dieselnet.com/standards/cycles/ftp_us06.php
[47] “Emission Test Cycles: FTP-72 (UDDS).” Accessed: Oct. 16, 2023. [Online]. Available: https://dieselnet.com/standards/cycles/ftp72.php
[48] R. P. Ltd, “Buy a Raspberry Pi 4 Model B,” Raspberry Pi. Accessed: Oct. 17, 2023. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/
[49] “PiCAN2 CAN-Bus Board for Raspberry Pi 2-3,” SK Pang Electronics Ltd. Accessed: Oct. 17, 2023. [Online]. Available: https://www.skpang.co.uk/products/pican2-can-bus-board-for-raspberry-pi-2-3
[50] “MATLAB Support Package for Raspberry Pi Hardware Documentation - MathWorks Italia.” Accessed: Oct. 17, 2023. [Online]. Available: https://it.mathworks.com/help/supportpkg/raspberrypiio/

Contact Information

Mailing address: University of Bologna, via Umberto Terracini 28, 40131, Bologna, Italy.

Email address: stella.cane2@unibo.it

Definitions/Abbreviations

ANN	artificial neural network
ATS	after-treatment system
BagT	bagged tree
BSG	belt starter generator
CAN	controller area network
CO	carbon monoxide
CO_2	carbon dioxide
DA	discriminant analysis
DNN	deep neural network
DOC	Diesel oxidizing catalyst
DPF	Diesel particulate filter
DT	decision tree
ECU	engine control unit
EGR	exhaust gas recirculation
EM	electric motor
FDI	fault detection and isolation
FN	false negative
FNR	false negative rate
FP	false positive
FT	fine tree
FTP	federal test procedure
HC	hydrocarbons
HDV	heavy-duty vehicle
HiL	hardware-in-the-loop
HV	high-voltage
ICA	independent component analysis
ICE	internal combustion engine
KNN	k-nearest neighbors
LCV	light commercial vehicle
LSTM	long short-term memory
ML	machine learning
MVEM	mean value engine model
NB	Naïve Bayes
NO_x	nitrogen oxides
O_2	molecular oxygen
OBD	on-board diagnostic
OBM	on-board monitoring
PCA	principal component analysis
PHEV	plug-in hybrid electric vehicle
PNN	probabilistic neural network
RDE	real-driving emission
SBELM	sparse Bayesian extreme learning machines
SCR	selective catalytic reduction
SFTP	supplemental federal test procedure
SVM	support vector machine
TN	true negative
TP	true positive
TPR	true positive rate
WLTC	worldwide harmonized light vehicles test cycle
WNN	wide neural network