Artificial Intelligence Strategies for the Development of Robust Virtual Sensors: An Industrial Case for Transient Particle Emissions in a High-Performance Engine

Article ID: 03-17-02-0014 © 2024 SAE International doi:10.4271/03-17-02-0014

Artificial Intelligence Strategies for the Development of Robust Virtual Sensors: An Industrial Case for Transient Particle Emissions in a High-Performance Engine

Leonardo Pulga,1 Claudio Forte,1 Alfio Siliato,1 Emanuele Giovannardi, 2 Roberto Tonelli,3 Ioannis Kitsopanidis,3 and Gian Marco Bianchi,2

1 NAIS s.r.l., Italy

2 University of Bologna, Italy

3 Ferrari s.p.a., Italy

Abstract

The use of data-driven algorithms for the integration or substitution of current production sensors is becoming a consolidated trend in research and development in the automotive field. Due to the large number of variables and scenarios to consider; however, it is of paramount importance to define a consistent methodology accounting for uncertainty evaluations and preprocessing steps, that are often overlooked in naïve implementations. Among the potential applications, the use of virtual sensors for the analysis of solid emissions in transient cycles is particularly appealing for industrial applications, considering the new legislations scenario and the fact that, to our best knowledge, no robust models have been previously developed. In the present work, the authors present a detailed overview of the problematics arising in the development of a virtual sensor, with particular focus on the transient particulate number (diameter <10 nm) emissions, overcome by leveraging data-driven algorithms and a profound knowledge of the underlying physical limitations. The workflow has been tested and validated using a complete dataset composed of more than 30 full driving cycles obtained from industrial experimentations, underlying the importance of each step and its possible variations. The final results show that a reliable model for transient particulate number emissions is possible and the accuracy reached is compatible with the intrinsic cycle to cycle variability of the phenomenon, while ensuring control over the quality of the predicted values, in order to provide valuable insight for the actions to perform.

1. #Introduction

Incoming Euro VII legislation is expected to introduce the requirements for car manufacturers to increase the realtime monitoring and control of pollutant formations from each vehicle [1]. This scenario will translate into the need for developing reliable sensors for gaseous pollutants and particulate matter, currently not implemented in most vehicles [1, 2]. A viable and interesting alternative could be the definition of virtual sensors, capable of predicting pollutant emissions based on signals already available to the ECU or in combination with new hardware. In fact, the simultaneous increase in on-board electronics computing power is making it possible, in the near future, to deploy more sophisticated algorithms for tasks such as automatic driving functions, but also allowing the use of data-driven models in place of more traditional maps and correlations. Moreover, car manufacturers are collecting a large amount of data related to engine performance and emissions during virtual and physical testing, thanks to the increased number of sensors installed and the varieties of tests to be conducted. The combination of these two factors, makes it valuable to focus current research efforts on the applicability and optimization of data-driven models for the generation of virtual sensors either for real-time applications on edge devices [3] or from a cloud interface for fleet analytics tasks.

Among the potential targets for this technology, the prediction of pollutants is of particular interest, especially with respect to the particulate matter emissions, traditionally identified as a highly transient and variable phenomenon, more difficult to measure experimentally on-board or to predict with steady-state maps and models than HC, NOx, CO, and CO2 [4].

The study and modeling of particulate formation has been subject of many studies in recent years [5, 6, 7], from diesel and GDI engines, for which both particle mass (PM) and particle number (PN) are strictly regulated in Europe since 2011 for light-vehicle

diesel engines and since 2014 for petrol engines. The PN measure, in particular, is limited to 6 × 1011 #/km for detected particles with diameter larger than 23 nm. However, also smaller size particles are being actively investigated [8], considering the health concerns related to the high number present in the exhaust gases. In fact, particles with diameter between 10 nm and 23 nm are expected to represent a significant fraction of the overall number of solid particles in the nucleation stage (typically in the range 10–40 nm) [8] and their emission must be measured and controlled.

Some works, in recent years, have already been proposed with a focus on the use of artificial intelligence for the prediction of pollutant formation from internal combustion engines; in Küçük [9] for example, the authors have applied a different set of machine learning algorithms for the prediction of NOx, which is the same target proposed in Brusa et al. [4] and in other works, mainly related to compression ignition engines [10, 11, 12, 13]. In other articles, the authors have been successful in applying data-driven techniques to predict the combustion properties of different fuels and engines [14, 15] as well as predict the effect of parameter variations on the pollutant formation and properties during steady measurements. Only a limited number of articles have focused on the transient nature of pollutant formation phenomena [16], while, for example in [17], the authors have developed a combination of autoregressive and static NN models to predict the complex interactions occurring in a diesel engine aftertreatment system, in order to accurately compute its dynamic behavior, which is of paramount importance for the prediction of tailpipe emissions.

To the best knowledge of the authors, there is no report in literature of attempts at predicting the highly transient formation of solid particles from gasoline engines in industrial applications. Considering the novelty of the subject, there is no consolidated approach for all the modeling aspects involved; therefore, many potential choices need to be presented and discussed for taking informed decisions.

2. #Research Structure

In the present work, the authors introduce a robust machine learning pipeline for the prediction of engine-out solid emissions with real-time constraints. The methodology has been applied on the PN emissions prediction task, for which an accurate data analysis has also been conducted and the results, obtained on a validation dataset of experimental cycles, will be discussed.

In Sections 3–5 the experimental data will be presented and described accurately, in order to underline the requirements for developing a consistent modeling methodology and inform about the potential risks of considering the raw data as is.

Sections 6–9 are devoted to present the possible modeling approaches that need to be considered, leveraging the properties of different machine learning and deep learning algorithms. Particular focus will be given to the distinction between recurrent and nonrecurrent methods, as well as the necessary data preprocessing techniques.

Sections 10 and 11 will provide a synthetic overview of the results obtained with cross-validation and a deeper discussion about the performance of the models on a set of different validation cycles, focusing on different and complementary metrics.

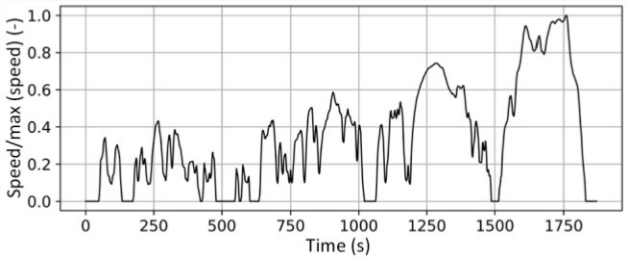
Observations regarding potential applicability, and performance measures will be outlined in the concluding Sections 12 and 13, with particular emphasis on the universality of most of the workflow for application to all other engine-out pollutants and vehicle signals in general.

3. #Experimental Dataset

The available experimental data, used for training and validation of the entire workflow, have been collected from a set of experiments conducted on a test bench from a vehicle with a V12 GDI SI aspirated engine, with 12 cylinders and overall displacement volume of 6495.6 cc. The test bench is a modification of a classic roll bench, with direct connection of the dyno to the front and rear axles, which is expected to have no consequences on the emission levels, but allows a better control on the vehicle dynamic behavior while avoiding wheel slip. All the data were recorded during transient cycles, with a virtual driver following two kinds of velocity profiles displayed in Figures 1 and 2:

*) Homologation cycles (WLTC),
*) RDE-like cycles, recorded in previous experimental tests on a real vehicle on streets in the neighborhood of Maranello containing urban, rural, and motorway conditions.

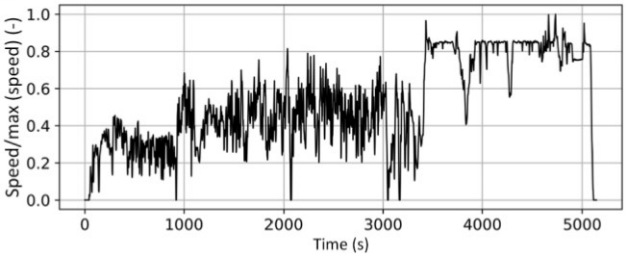
FIGURE 1 **Normalized vehicle speed profile for WLTC cycle.**



While the ambient temperature was controlled, and kept fixed at 25°C, the temperature of the engine (cooling system and components) at the beginning of each cycle was varied in order to consider different starting conditions and speed up the data collection process, not requiring long waiting times for cooling. It is expected [18] that external air temperature and humidity may play a significant role in modifying the relationship between input variables and PN formation, influencing mixing and combustion variability; therefore, these :factors will need to be further investigated in the future, to provide samples for training a model robust to this variations.

The measurements of the particulate matter at the engineout position was performed with an AVL particle counter (APC) [19], which measures the number of particles with diameter greater than 10 nm in the exhaust gas with a frequency of 10 Hz and a calibrated range of 0–10,000 #/cm^3.

FIGURE 2 **Normalized vehicle speed profile for RDE-type cycle.**



The instrument is composed of a pump, which draws the exhaust gas after the three-way catalysts right before the GPF, eliminating all particles >2.5 μm thanks to the special shape of the pump. The sampled gas is diluted with hot air at 150°C to stabilize and distribute the particles, in order to reduce the risk of agglomerations and deposits. After a further heating and dilution phase, where all the volatile particles are expected to be converted to the gaseous phase, the sample enters the condensation particulate counter at a temperature of 35°C where the light-scattering method is used to measure the particle concentration after they are enlarged thanks to the condensation of butanol [19]. Thanks to this measurement system, “particulate losses” and particle deposits that might spoil the transient results are expected to be entirely controlled.

The final dataset is composed of 36 cycles for a total time of more than 20 h of driving, for which transient particulate number emissions are recorded, as well as the ECU channels normally available on the vehicle, which are essential to describe engine behavior. In particular, for each of the two banks of the engine, the available signals could be divided into four main groups:

*) Operating conditions and control data: speed, acceleration, rpm, gear, pedal request, load, spark advance, valve phasing, and intake and exhaust mass flow rates (the former is directly measured while the latter is estimated from intake mass flow and injected mass);
*) Spray-related quantities: injection pressure, injected fuel mass, SOI, and duration of different injections;
*) Other sensors’ readings: lambda from HEGO and UEGO probes, temperature of oil and water-cooling circuits, temperatures from thermocouples on the exhaust line (not currently available in vehicle, therefore removed at a later stage in the development);
*) Switches activation information for control purposes (whether catalyst heating strategy or split injection are active, the lambda probe reliability, engine bank deactivation control, component protection strategy activation).

4. #Dataset Preparation

The first step in the handling of time-series data from different sources requires the verification of the synchronicity between the signals, which would otherwise impair the performance of data-driven algorithms. In order to verify and eventually correct the time shift, a classical approach is based on the cross-correlation calculation, starting from the assumption that the two time series are correlated and, therefore, the time delay that mostly increases the correlation coefficient between them is the desired value. Considering that time-series data can usually be composed of a large number of samples, the method applied in the current work is

the one proposed by [20], based on an efficient FFT-based approach, which reduces the complexity of the operation from O(n 2) to O(n log n).

FIGURE 3 Effect of time shift on relative behavior between PN10 and load.

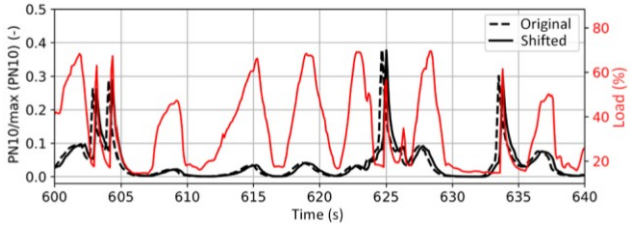
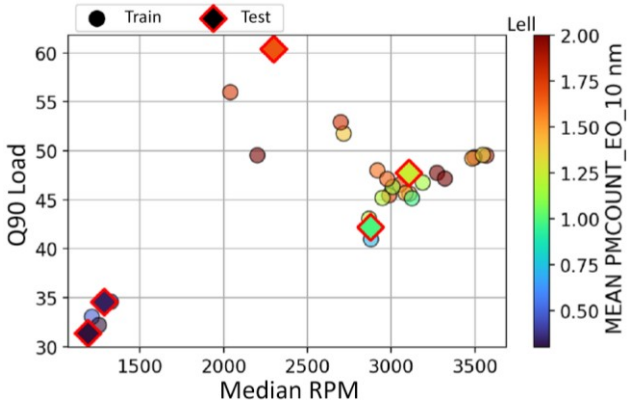


FIGURE 4 Distribution of train and test cycles in the rpm/load plane.



Starting from the work of [21], which will be further analyzed in Section 9, the parameter that best correlates with particulate number emission is the intake manifold pressure, which is not available in the current dataset, but can be physically considered an indicator of the engine load, which was therefore selected as the second variable. The method was applied after normalizing with standard scaling the load (transforming the distribution to reach mean value = 0 and standard deviation = 1) and logarithmic transform the target (in Section 6 more details will be provided) and, as can be seen from Figure 3, the shift obtained from this methodology was capable of identifying the correct variation.

At the same time, cycles with low Spearman correlation between the two variables, irrespective of the shift, should be carefully evaluated, due to concerns on the trustworthiness of the data. This process led to the identification of four cycles that, after further investigation were discarded for anomalies in the experimental measurements, leading to 32 actually available cycles.

Considering that no missing values were found in the dataset, the second step of the data preparation concerned the split between training and test data. In order to produce a fair final evaluation [22], five cycles selected heuristically (1 WLTC and 4 RDE cycles) were held out from the dataset used for all further evaluations and developments, leading to a training dataset composed of 27 cycles (15 h of driving).

The test set is considered to be representative of most real scenarios in which the model might operate after deployment, as can be seen from Figure 4, where the 0.9 quantile of the load and the median rpm value of each cycle (excluding the phases with engine off) are used as descriptive variables. At the same time, Figure 4 also shows that the train/test split performed for the final validation is consistent with the requirements of data-driven models of being trained and evaluated inside the same domain, which might not be, however, enforced in real-life applications, and other strategies, such as domain adaptation techniques should be taken into consideration for improving the generalization capabilities of the model.

This is why it is of paramount importance to define a pipeline capable of handling or, at least, identifying data drift from the input, to ensure that the ML model is working within valid ranges. Several techniques are suited for this task, based on machine learning models (specifically clustering or classification algorithms), statistical analysis, and eventually algorithms developed for addressing the anomaly detection task.

A combination of these methods was considered the most robust approach for our application, to detect several possible variations [23] and, in particular, the full pipeline was composed of:

1.) Isolation forest for anomaly detection, to check whether each new point lays within the training range.
2.) A combination of Wasserstain Distance metric (WD) and Population Stability Index (PSI) to check similarity between the distributions of the new points with respect to the training set.

The isolation forest is an unsupervised ensemble method, which uses as base learners extremely randomized trees (regression

trees employing only one feature for performing the split operations). After fitting the model on the training set, the prediction stage returns a measure of “normality” of the new sample, obtained by averaging the number of splitting from each tree, required to isolate the sample from the dataset.

For out-of-distribution points, the length of the path from root node to the terminating node is significantly lower than for in-distribution points, and this information is used to estimate the potential outlier, using a predefined threshold values calculated on the training set.

The WD is a statistical measure of the distance between two distributions, which is used to compare the n dimensional distributions of all relevant features in the training set w.r.t. new points.

The PSI calculation starts from the binning of the target variable (in the presented case using equal width bins):

$$PSI = \sum \left(N(b) - M(b) \right) \ln \left(\frac{N(b)}{M(b)} \right) \qquad \text{Eq. (1)}$$

where B is the total number of bins, N(b) is the fraction of points from the new distribution that belong to bin b, M(b) is the fraction of points from the reference distribution that belong to bin b. This implies that PSI value ranges from 0 to infinity, but in practice, the following ranges are accepted:

$$\begin{cases} PSI < 0.1 : \textit{No significant population change} \\ 0.1 \le PSI < 0.2 : \textit{moderate population change} \\ PSI \ge 0.2 : \textit{significant population change} \end{cases}$$

In a deployment scenario, the use of isolation forest will help in identifying points outside of the training domain, due to, for example potential sensors’ faults. At the same time, the use of the distribution metrics (WD and PSI) with a lower frequency will help in identifying potential variations in the relationship between variables due to uncontrolled components’ aging, damages, or other sources of drifts, which might compromise also the accuracy of the predictive algorithm.

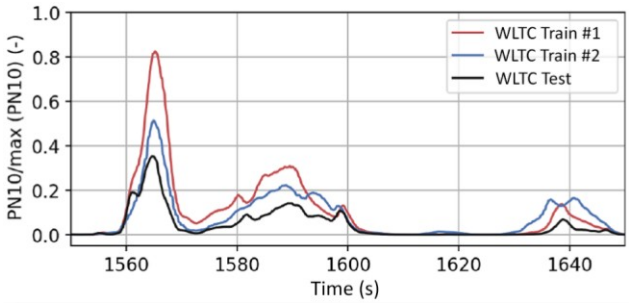
All three methods have been applied to the test set, in order to verify their applicability and, at the same time, ensure that the chosen validation cycles were not affected by data drift, which would require to reconsider the dataset split and data acquisition strategy. Detailed results on each cycle will be outlined in the next sections.

5. #Aleatoric Uncertainty Quantification

While data-drift detection addresses the concept of preventing the model to operate on unseen data after model deployment, in data-driven modeling for sensitive targets, uncertainty quantification is an essential aspect to consider. Traditionally, two sources of uncertainty are identified: aleatoric and epistemic. The former is also referred to as “data uncertainty” and it is related to the inherent complexity of the data due to intrinsic noise or variations related to external and unidentified features. In particular, engine emissions might be affected by an intrinsic variability of the air charge motion due to turbulence effects and combustion progress, or external parameters not registered, such as ambient humidity, cold start, frequent engine restart or emission history, which are expected to impact the solid emissions [24, 25, 26, 27, 28].

Section 7 will focus on the epistemic uncertainty, but as far as aleatoric uncertainty is considered, one of the most important aspects to consider is that, in practice, the addition of more training data cannot be used as a procedure for its reduction; therefore, it must be carefully investigated and known a priori. In literature, the data uncertainty related to the formation of PN in GDI engines has been experimentally investigated in [29, 30]. Their findings are particularly relevant, since the experiment is composed of a repetition of the same 14 s maneuver for 100 consecutive tests on a GDI engine, for which PN is stored. The results have been collected from the engine start, with coolant and engine oil at ambient temperature, but after the first 20 cycles, they reach a steady and constant value. Considering that, in GDI engines, liquid film formation due to spray impingement can be considered one of the driving factors for the formation of PN, it is not surprising that during the initial thermal transient phase the collected value of PN is up to 400% the average value of the last cycles. It is, however, noteworthy that, for the same maneuver, after the cooling liquids have reached a steady value, the measured PN shows a test-by-test variability distribution that can be represented with a Gaussian with standard deviation equal to 40% of its mean value.

FIGURE 5 Example of PN10 profiles on three WLTC cycles synchronized (all main actuations are coincident), underlying the aleatoric uncertainty of the results.



While finding the same level of reproducibility of the mentioned experiment in the available transient experimental dataset was not possible, some time periods have been isolated, and they present a comparable level of variability. In Figure 5, for example, the instantaneous PN profile of the same time window from three available WLTC cycles has been isolated. Operating conditions and actuations have been synchronized, and the resulting PN profile shows that, for the same maneuver, at steady coolant temperature, the aleatoric uncertainty can reach very significant values, that must be taken into account when defining performance metrics.

6. #Experiments

As it can be seen in Figure 6, the distribution of the target variable PN10 spans different orders of magnitude, and it is highly skewed toward lower values.

Considering the fact that traditional data-driven algorithms aim at minimizing some flavors of the average errors, the immediate conclusion is that some preprocessing is required in order to handle the target distribution. To this aim, a few techniques can be directly employed, that is logarithmic transformation or binning, and after internal preevaluations, binning was chosen as target preprocessing methodology. The binning strategy implies the transformation of the target from a continuous distribution to a discrete variable with a limited number of values (called bins), as in Figure 7.

The number and reference value for each bin has been optimized through grid search optimization from an initial range of reasonable combinations, with the target of maintaining the cumulated error on the training dataset, due to the discretization strategy, within an acceptability range (3% error) and, at the same time, minimize the number of bins required, in order to reduce the potential modeling effort for the prediction, leading to a final value of 13 bins.

FIGURE 6 **Distribution of the normalized target variable PN10 in the training set (log scale to highlight the skewness).**

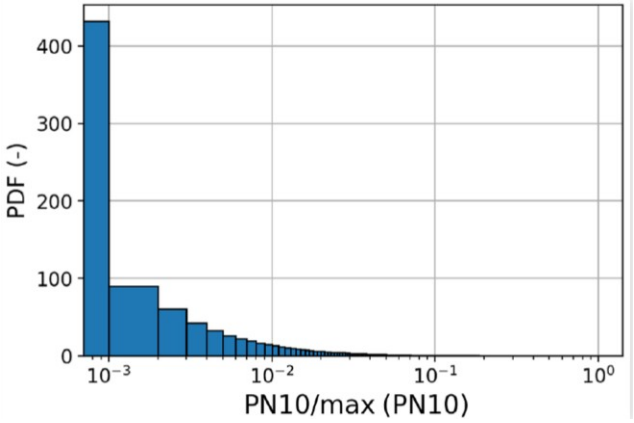
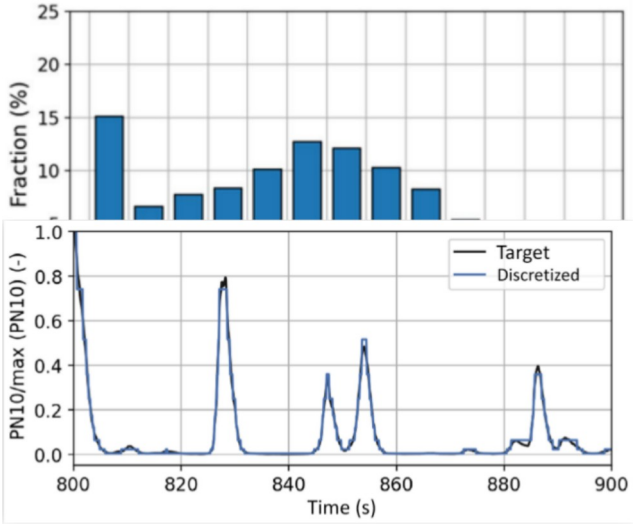


FIGURE 7 **Effect of the binning strategy on the original profile.**

FIGURE 8 **Relative fraction of each bin in the train dataset.**



The use of the binning strategy allows also to configure the prediction problem as a multiclass classification task, considering each bin as a different target, without keeping into account its nominal value. From the class distributions in the training dataset (Figure 8); however, it results clear that the class imbalancement is still in place and must be handled with appropriate techniques. In particular, it was chosen to compute a weight for each class equals to the inverse of its relative fraction (with maximum value set as 20), to be used for the computation of any loss function, in order to compensate for the not equal distribution of the data.

After transforming the target data, considering the fact that the input data is composed of time-dependent quantities, two modeling approaches are usually presented in literature: recurrent and nonrecurrent algorithms. Several algorithms have been developed to address the task of time-series prediction, from the traditional Auto Regressive algorithms or Box-Jenkins, to more sophisticated ones such as SARIMA [31].

However, after careful analysis regarding the target behavior, it was decided to not consider forecasting models whose prediction is strongly based on previous values. In fact, the behavior of the PN formation results strongly influenced by the exogenous variables (e.g., engine operating point and thermal state of the cooling system), rather than from its time history (also because the sampling frequency of 10 Hz does not allow to consider fast dynamics). Therefore, it was decided to focus on more “static” machine learning algorithms, considered appropriate for this kind of task, but still account for the dependency of the target on the transient behavior of the input variables thanks to different approaches that will be outlined in the following paragraphs. Therefore, in the present work, a set of classification models has been chosen to represent different approaches, while ensuring sufficient ability to reproduce the complexity of the problem (bagging and boosting of decision trees and different NN architectures). The set of considered models is composed by: recurrent NN, convolutional NN, deep NN leveraging the layers implemented in the TensorFlow library [32], random forest, and gradient boosting (in the implementations of XGBoost [33] and CatBoost [34] libraries).

As far as the recurrent and convolutional networks are concerned, the input of the network is expected to be a set of time series

with predefined window length, representing each channel up to the target time step. The starting point of the window has been optimized in the range (−10 s, −1 s) up to the current point (for which the target is stored). The optimal window size was found to be 5 s, identified with grid search optimization in combination with other hyperparameters (number of GRU layers and number of neurons of the inner layers for the RNN and number of 1D convolution layers, number of filters, and number and dimension of the dense layers for the CNN), and the following results will refer to that particular setup, reported in Appendix A.

As far as the nonrecurrent methods are concerned, all model require a feature engineering step in order to extract the most significant information from the time series, and convert the problem to a tabular classification task based on the extracted features. More details about the features extraction process will be provided in Section 9. After converting the input data, the hyperparameters of the chosen models have been optimized with Bayesian optimization in order to maximize the cross-entropy score of the multiclass classification task. In particular, for the deep NN, the number of hidden layers and neurons for each layer have been optimized, keeping constant learning setup and ReLU activation functions. For the random forest algorithm, the number of base estimators (classification trees) and minimum number of samples in each leaf of the trees have been optimized.

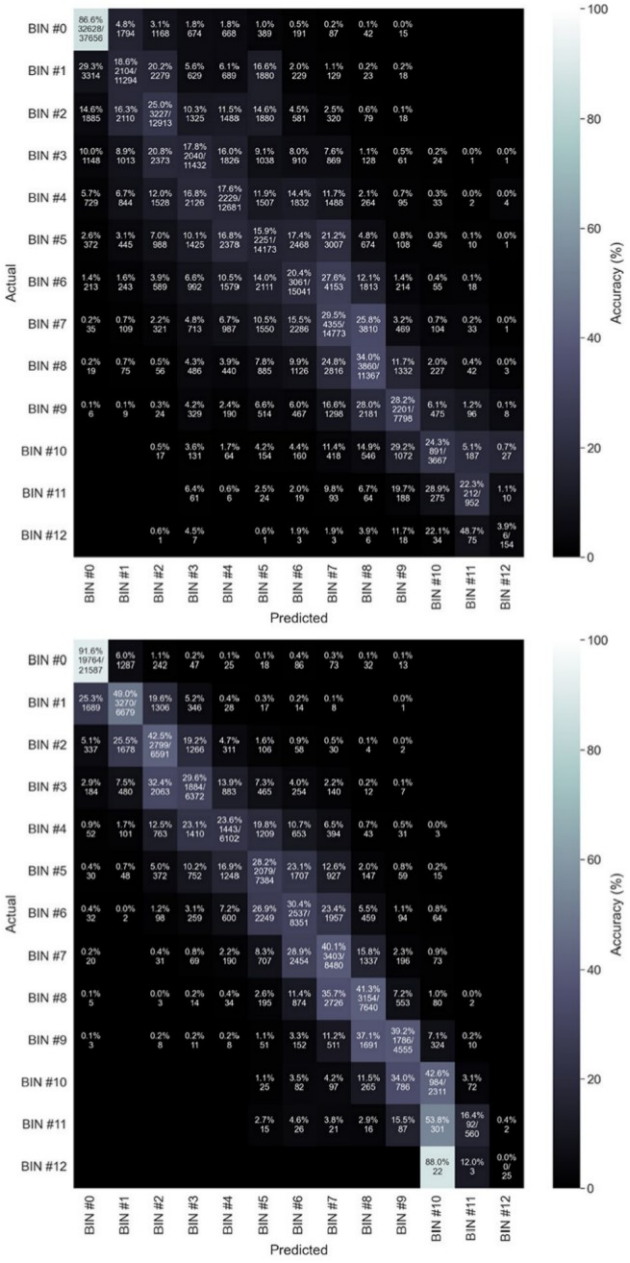
With respect to gradient boosting algorithms, XGBoost's hyperparameters have been optimized, while the use of histogram-based tree method has been kept constant to leverage the GPU implementation with single precision histogram, and the early stopping value was set to 100 iterations in order to stop the training if no improvement on a validation dataset is recorded.

For CatBoost, the optimized hyperparameters are collected in Appendix A, employing the virtual ensemble technique to allow the prediction of confidence intervals without the need to train more than one model. Besides the hyperparameters optimization, a different set of tests has been performed, focusing on the ordinal nature of the classes. In fact, differently from traditional multiclass classification tasks, the binning process implies an ordinal relationship between the classes, and the loss function could leverage this relationship. In order to include this observation, all the experiments have been re-ran considering a regression task, and using the RMSE as loss function for all models (maintaining the other optimized parameters unchanged). The L2 loss function was selected for its characteristics of penalizing large errors with more weight than small errors, which could further help in identifying correctly the large PN values, which are the most underrepresented class. The effect of this choice is immediately visible in the confusion matrix on the validation dataset of the XGBoost model, reported in Figure 9 after training with the traditional categorical cross-entropy loss function (a) and the RMSE loss function (b). Besides a small accuracy improvement, it must be noted that the mean distance between target and predicted classes has reduced, as well as the misprediction of points in the most extreme classes, which is a highly desirable behavior for our application.

7. #Epistemic Uncertainty Quantification

In literature, several techniques are proposed to assess the confidence interval in the prediction of machine learning and deep learning algorithms. Of particular interest for their straightforward implementation are bagging ensemble methods [35], conformal predictions [36], and the use of models trained to predict the coefficients of some prior assumed distribution. Since this work was not intended to provide an overview of different confidence prediction algorithms, but the applicability of a robust data-driven prediction pipeline, only the use of different flavors of the ensemble strategy have been employed. In particular, to limit model complexity for large deep learning algorithms, as RNN, CNN, and deep NN, the Monte Carlo dropout strategy [37] has been applied. The dropout strategy is a classical regularization technique used to increase the stability of the training process and reduce the risk of overfitting of NN by randomly setting to 0 a fraction of the inputs received by the next layer. When applied also in inference time, however, it results in a stochastic prediction, by virtually evaluating a different model each iteration. Based on this assumption, the Monte Carlo dropout algorithm requires to make the prediction a sufficient number of times (set up to 100), to allow for the computation of mean value and standard deviation of the distributions of the predictions.

FIGURE 9 Confusion matrices on validation set with reference multiclass classification task (upper) and leveraging the ordinal nature (lower).



As far as XGBoost and random forest are concerned, a simple ensemble strategy has been employed, by training five models initialized with different random seeds and on random 80% of the training set, in order to induce more variability in their training process.

With respect to the CatBoost implementation, it is possible to leverage a variation of the ensemble strategy for identifying the epistemic uncertainty, using an ensemble of models stored during training in different iterations (thus avoiding the need to fit different models) [38].

In all cases, by calculating the difference between total uncertainty of the predictions and expected data uncertainty, it is possible to quantify the expected epistemic uncertainty of the models. The use of ensembles is, in general, expected to provide an estimate more robust to bias [35] and, at the meantime, the knowledge of the model variability is a useful measure to account for while tuning the algorithms, considering that epistemic uncertainty can be reduced by optimizing the models or working on the training data (either by adding more points or including more relevant features).

8. #Custom Metrics

Another important aspect to deal with is related to the metrics that must be employed for the final predicted values, for which traditional regression metrics (MAE, R2, MSE, relative error) might not be appropriate. Considering, in fact, the distribution of the PN values and their physical values two approaches might be more appropriate and have been evaluated:

(a) Considering the relative error, which might still suffer from different dynamic predictions (i.e., predictions of the correct value but with a minimal temporal shift)

(b) Considering the integral value of the profile, which is, on the other hand, the real variable of interest for the final evaluation However, calculations based on the integral value can lead to misinterpretations of the results, considering the potential summation of errors with opposite signs, which might lead to good final values without providing robust predictability. For this reason, the results have been evaluated in terms of different metrics, constructed as the R2 and the relative error of the integral value in a limited time window of 5, 10, and 20 s recomputed every second (similarly to the moving average computation, for example computed for the 20 s relative error in Equation 2).

$$E_{20} = \frac{\sum_{i=1}^{n} |PN_{w20,i} - \hat{PN}_{w20,i}|}{\sum_{i=1}^{n} PN_{w20,i}} \% \quad \text{Eq.2)}$$

9. #Feature Engineering

Considering that the data acquired with a frequency of 10 Hz correspond to the average values of a different number of cycles depending on the engine speed (5 cycles at 6000 rpm against only 1 cycle at 1200 rpm), the feature engineering phase is expected to play a crucial role in the improvement of the performance of the model.

Since the experiments proposed are of various nature, two different pipelines have been developed depending on the use of recurrent or nonrecurrent methodologies. In the former case, the dataset has been designed in the form (#samples, #steps, #channels) for all features, where #steps corresponds to the dimension of the time window and #channels to the number of signals. With respect to the nonrecurrent feature generation phase, a three-step approach has been performed:

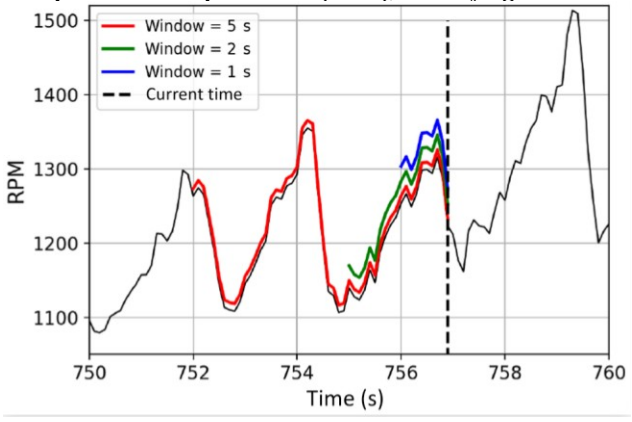
(a) Identification of a time window preceding the current step, with various sizes (1, 2, 5 s), to account for different “maneuvers” (as it can be seen from Figure 10).

(b) Calculation of relevant synthetic features from each time window and channel (e.g., final values, statistical values, positional values, derivatives, and others listed in Appendix B)

(c) Reduction of the number of inputs to help prevent overfitting, by following the principle of minimum redundancy maximum relevance (mRMR) [39].

Considering the fact that many features are expected to be highly correlated (e.g., the maximum values in overlapping windows of different sizes are intuitively expected to be coincident in many instants), the algorithm for identifying the most significant inputs has been carefully evaluated based on possible implementations suggested in literature [39]. For the chosen algorithm, two quantities needed to be computed: a correlation term and a feature importance term.

FIGURE 10 Time windows description on a sample feature (RPM), underlying the different types of dynamics.



For the correlation term, the Spearman correlation coefficient ρ(XS, Xi) between the ith and the sth feature was chosen, computed as in Equation 3 and indicating the monotonicity of the two sets of data (not limited to the linear hypothesis of the Pearson correlation coefficient).

$$\rho(X_s, X_i) = \frac{cov(R(X_i), R(X_s))}{\sigma(X_i)\sigma(X_s)} \quad \text{Eq. (3)}$$

where cov(R(Xi), R(Xs)) indicates the covariance of the ranks of the two variables and σ the standard deviation.

On the other hand, there are several techniques to calculate the importance of each feature for the final prediction.

Based on previous internal experimentation, a two-step approach has been employed:

(a) Identification of the mutual information, which is a statistical description of the relationship between two variables, calculated as the Kullback–Leibler divergence of the joint distribution of the two variables (p(Xi, Y)) from the product of their marginal distributions (pXi · pY):

$$MI_i = D_{KL}\left(p\left(X_i, Y\right) \middle\| p_{X_i} p_Y\right)$$

where Xi is the ith feature and Y represents the target variable. The value of MI is always positive, and reaches a value of 0 when the marginal distributions and joint distribution coincide (i.e., when the variable X and the target Y are independent). The potential of this metrics is mostly the ability to overcome some limitations of classical correlations index, and not relying on a trained model. The threshold value employed for a feature to be considered relevant was 0.1, which is a low value in general, but its main purpose was to discard the least relevant features before computing feature importance.

(b) Feature importance is, on the other hand, a metric derived from the analysis of the behavior of a trained model (in our case we have employed directly the XGBoost algorithm). There exist three major strategies to extract feature importance values from fitted tree-based algorithms: Shapely values, feature weights and permutation importance. Shap values (computed with the shap library [40]) represent the average of all the marginal contributions to all possible coalitions of features, borrowing the principles of cooperative game theory. While shap values are model agnostic, the computation of features weights is performed by computing the sum of all the occasions in which one feature has been used to create a new node, indicating its impact on the final prediction and, hence, its relevance.

Permutation importance is an extremely timeconsuming method, although very effective in identifying the sensitivity of the final prediction on each feature, by randomly shifting the values of the features one by one, and computing the final performance on the perturbated dataset (the features that have induced the highest performance drop are deemed more relevant). The feature importance methods employed, due to their limited time requirements, were shap value and feature weights, combined by averaging their normalized scores (normalization performed to have importance scores in the range [0,1]).

After removing the features with mutual information value below the threshold, the application of the mRMR method has followed the principle of the quotient, reported in Equation 4, where S is the feature space and |S| is the number of features.

$$rRMR_i = \frac{FI\left(Y, X_i\right)}{\frac{1}{s}\sum_{X_i \in S} \rho\left(X_s, X_i\right)} \qquad \text{Eq. (4)}$$

While the application of this approach on the single features for the nonrecurrent experiments is straightforward, the same is not immediately transferable to the recurrent formulation of the problem. However, thanks to the insights gained from the features chosen with the mRMR algorithm, only the channels present in the final dataset (independently from their transformations) have been selected.

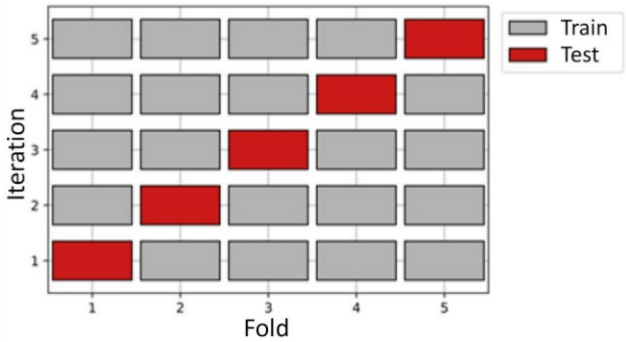
As a further verification, the most relevant features identified from the algorithm have a strong connection with the physical intuition and previous experimental results from literature [21], where the most relevant features identified were the inlet manifold pressure, the injection angle, throttle position, engine torque, and AFR.

The initial number of available channels was 53, from which over 700 features were calculated, but the reduction process has led to the final number of 228 employed inputs. The most relevant of these features refer to different dynamic properties of various properties of the engine operating condition, namely the intake air mass flow, the oil temperature, the injected mass, the engine load, and speed and the signal of engine stop.

10. #Cross-Validation Results

As anticipated in Section 2, the final validation has been performed on five unseen driving cycles, while all optimization steps have been conducted using the k-fold cross-validation results (k = 5), repeating k-times the training and testing following the scheme in Figure 11. Considering that the problem is based on time-series, the split was not performed randomly, but each cycle in the training set was evenly divided into five fractions, and the train and test set were obtained combining, respectively, 4/5 and 1/5 of the fractions of all training cycles. The results that will be presented in this section refer to the cross-validation steps.

FIGURE 11 **Scheme of the five-fold cross-validation dataset split.**



As far as the multiclass classification view of the problem is concerned, the ROC-AUC calculated with a one-vs-one strategy and macro averaging has been employed, in order to reduce the impact of the class imbalancement in the calculation, without referring to the weights used during training. However, considering that the transformation of the problem in a classification task represents an artifact useful for handling the data, also the traditional regression metrics R2 and MAE have been reported on the transformed output. The tables with the results of the cross-validation report both the average ROC-AUC, R2 and MAE, as well as the standard deviation of the MAE with respect to the five folds. In the first row of all performance reports, the error induced by the discrete binning strategy is reported as a threshold value that the models are expected to tend to, but not improve on.

As reported in Table 1, the standard deviation of the MAE calculated on the instantaneous emission values is below 20% of the mean value, indicating a sufficient consistency of the predictions.

TABLE 1 **Results of the cross-validation process with multiclass classification loss function.**

Multiclass	ROC-AUC	R2 avg.	MAE avg.	MAE std.
DISCRETE	—	0.96	2.4E10	—
RNN	0.64	0.50	8.2E10	2.4E10
CNN	0.61	0.51	9.1E10	2.2E10
NN	0.69	0.59	6.8E10	1.8E10
RF	0.50	0.36	6.4E10	1.1E10
XGB	0.73	0.58	6.6E10	1.2E10
CTB	0.66	0.68	3.7E10	1.0E10

By leveraging the ordinal nature of the classes, the models show an improvement in performance with respect to the metrics applied to the transformed data (Table 2).

TABLE 2 **Results of the cross-validation process leveraging the ordinal nature of the classes.**

Ordinal	ROC-AUC	R2 avg.	MAE avg.	MAE std.
RNN	0.61	0.54	4.8E10	2.1E10
CNN	0.58	0.56	5.1E10	1.6E10
NN	0.74	0.61	3.5E10	1.3E10
RF	0.51	0.36	6.4E10	1.1E10
XGB	0.75	0.69	3.2E10	0.5E10
CTB	0.65	0.68	3.7E10	0.7E10

As can be seen from all metrics, both directly computed on the predicted classes and calculated on the transformed values, the models trained on engineered features perform on average more efficiently than RNN and CNN, which receive the same amount of information through the entire timeframe. Within the best performing models, it is noteworthy that the random forest algorithm is not capable of handling the complexity of the target. XGBoost, CatBoost, and the NN, on the other hand, all demonstrate a good predictive performance, especially in terms of the converted regression metrics.

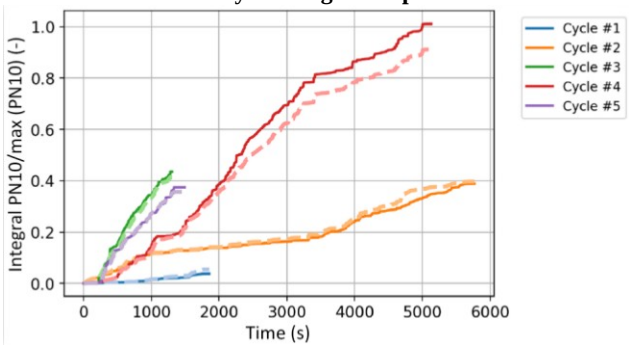
Considering the other possible target, PN23, the same analysis has been performed, with no significant variation in the conclusions; therefore, in the next section, only the values of PN10 will be discussed.

11. #Results on Validation Cycles

The results that will be presented refer to different cycles with increasing aggressivity levels, from WLTP to various RDE cycles.

The synthetic results confirm that the best performing approach is the ensemble of XGBoost models trained leveraging the ordinal nature of the classes before conversion.

FIGURE 12 **Cumulated emissions of the validation cycles target and predicted.**



From Figure 12, it can be seen that the relative behavior of the different cycles is well predicted, as well as the integral value, which is also evident from the results shown in Tables 3–5, considering the instantaneous metrics R2, as well as the newly introduced relative error in windows of 5 s and the relative error of the integral emissions (more examples of model's predictions are reported in Appendix C).

TABLE 3 **R2 results of the different models on validation cycles.**

R2	Cycle #1	Cycle #2	Cycle #3	Cycle #4	Cycle #5
DISCRETE	0.95	0.93	0.95	0.99	0.99
RNN	0.49	0.53	0.46	0.59	0.42
CNN	0.43	0.51	0.46	0.65	0.37
NN	0.56	0.57	0.66	0.82	0.62
RF	0.33	0.36	0.30	0.34	0.10
XGB	0.64	0.66	0.71	0.75	0.74
CTB	0.64	0.68	0.69	0.78	0.79

TABLE 4 **Relative error of the predictions cumulated over 5 s windows of the different models on validation cycles.**

W5s rel. error (%)	Cycle #1	Cycle #2	Cycle #3	Cycle #4	Cycle #5
DISCRETE	2.73	2.73	2.58	2.78	3.84
RNN	38.29	39.98	40.89	44.1	41.41
CNN	39.99	42.46	42.84	41.75	43.67
NN	34.37	38.64	33.23	38.91	29.57
RF	56.8	74.27	87.59	84.74	103.1
XGB	35.15	38.2	37.82	38.02	28.76
CTB	38.37	44.01	46.04	38.87	27.48

TABLE 5 **Cumulated relative error of the different models on validation cycles.**

Cum. rel. error (%)	Cycle #1	Cycle #2	Cycle #3	Cycle #4	Cycle #5
DISCRETE	−0.1	0.3	0.9	0.5	1.24
RNN	8.03	−0.94	10.59	12.1	28.04
CNN	14.85	1.28	19.66	37.2	35.2
NN	−5.34	−10.0	−5.6	−14.2	7.8
RF	1.93	11.42	22.71	−0.43	33.4
XGB	−4.07	−10.3	−3.62	3.53	5.7
CTB	11.07	3.24	14.6	6.66	12.4

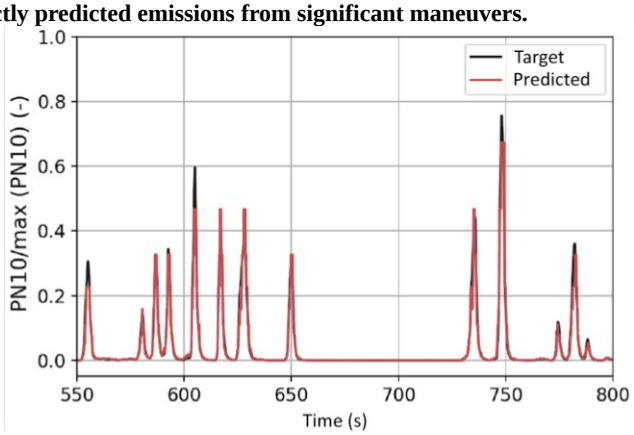
Whereas all the metrics show that the chosen algorithm (XGBoost) can outperform the alternatives, there are some observations that can be drawn from the results in Tables 3–5.

In particular:

*) There is a significant similarity between the R2 scores of the cross-validation phase and the results on the final validation cycles for all the models, suggesting that the optimization methodology was sufficient to provide a robust estimate of the final performance.
*) While some models display a significant variability in the results for different validation cycles, especially in terms of R2 and W5s (since the cumulated error may suffer from the compensation of underpredictions and overpredictions, as in the case of the RF results), the NN, XGB, and CTB models show a significant robustness against very different driving conditions.
*) Analyzing more in details the differences between NN, XGB, and CTB, it can be seen that they all display similar performances, superior to RNN and CNN models, indicating the importance of using engineered features instead of the raw input signals; this is possibly associated with the fact that, for this application, the time dependency of the output is less relevant than its dependency on exogenous variables, therefore the convolution or recurrent operations are mostly applied as a "feature extraction step," rather than leveraging all their potential, as assumed in Section 6.

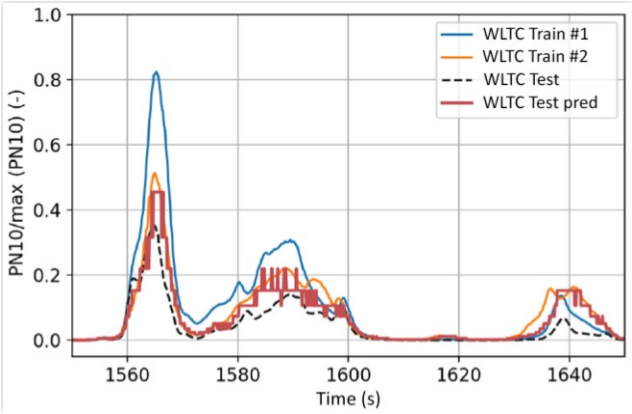
Focusing on the details of some maneuvers where the model predicts correctly the phenomena (e.g., gear down shift or acceleration), Figure 13 is an example of the capability of the model to accurately infer emission peaks position and absolute value.

FIGURE 13 Sample of correctly predicted emissions from significant maneuvers.



In some cases, however, as in the time window of Figure 14, the aleatoric uncertainty affects the instantaneous accuracy of the prediction w.r.t. the validation cycle. In the case of Figure 14, for example, the predictions of the model near 1570 s or 1640 s are not perfectly in accordance with the absolute value of the target; however, analyzing the behavior of the 2 WLTC cycles from the train set, after perfectly synchronizing most of the relevant features (which are almost coincident), it can be noted that there is an intrinsic variability in the absolute value of the peaks and the model's predictions lay within this range.

FIGURE 14 Prediction of XGBoost model compared with all WLTC cycles to present effect on prediction of aleatoric uncertainty.



12. #Applicability

For most applications, the real-time constraint needs to be respected for the entire pipeline, therefore a comparison of the tested models has been performed for a single value prediction. While there are several acceleration technologies (use of parallel computing, GPUs, or modification of the runtime engine) that can help speed up the process, they are not directly applicable to all

experiments; therefore, to maintain a fair evaluation, all modeling strategies have been compared with respect to a single CPU setup (Intel i9-12900 K, 3.4 GHz) and the computing times for preprocessing, inference, and post-processing are reported in Table 6. The values reported are computed for a single timestep prediction, and are therefore the average computing time for loading data and generating the features (preprocessing for nonrecurrent methods) or reshaping (preprocessing for recurrent methods), make a single prediction (inference), and transform the predicted class into a continuous value of PN emitted, with very limited variability recorded from point to point.

TABLE 6 Mean computing times (ms) of different steps for the complete modeling workflows.

Time (ms)	Pre	Inference	Post	Tot.
RNN	1.5	3.2	0.9	5.6
CNN	1.5	4.5	0.9	6.9
NN	2.1	2.4	0.9	5.4
RF	2.1	2.5	0.9	5.5
XGB	2.1	1.6	0.9	4.6
CTB	2.1	2.2	0.9	5.2

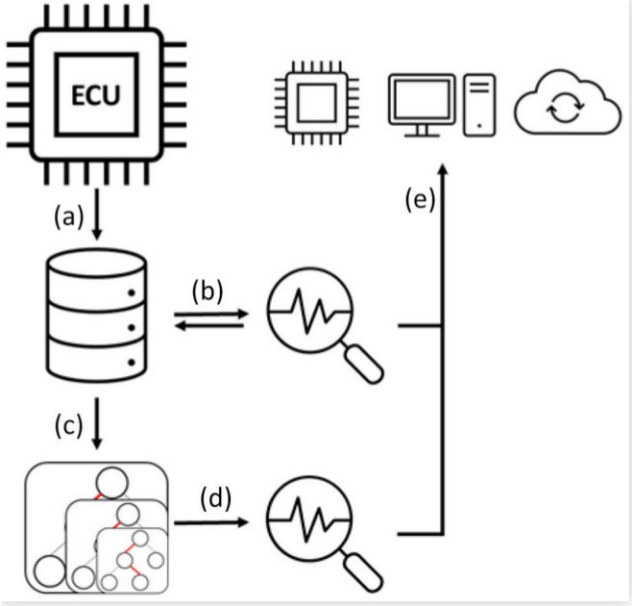
The results show that the total computing time for a single prediction, independently from the method used, would allow real-time compatibility, considering the current acquisition frequency of 10 Hz, with the fastest workflow being the one based on XGBoost, thanks to its inference speed, which, together with its accuracy make it the most suitable choice for this application.

Even after verifying the real-time compatibility, as it was discussed in the previous paragraphs, the applicability of datadriven algorithms to virtual sensor technologies is far from straightforward, especially when targeting complex variables such as the PN emission. For a reliable model deployment in both edge or remote scenarios, the expected complete pipeline must be capable of handling issues related but not limited to delays in the communication of different signal sources, variability in the noise of the data sources, occurrence of unseen conditions, or change in the behavior of the engine not related to the input variables (malfunctioning or aging leading to increased variability).

The proposed approach, summarized in Figure 15, is composed of five essential parts, for which details have been described in the previous paragraphs:

(a) Data ingestion/storage from ECU and feature generation (features that were selected with the mRMR approach using a combination of shap values and feature weights and the Spearman correlation coefficient).
(b) Check for data misalignment, anomaly detection with isolation forest, and every 60 s verify WD and PSI.
(b) Prediction with confidence interval of PN10/PN23 engine out (using an ensemble of XGBoost models trained on different subsets in order to assess the prediction robustness).
(d) Verify with statistical limits that the confidence of the model is below threshold.
(e) Transfer results and warnings to final application, either real-time (engine emission unit, OBD, ECU) or off-line (test bench processor, cloud storage).

FIGURE 15 Schematic of the complete process for the robust application of virtual sensors from ECU to deployment.



13. #Conclusion

The current work has demonstrated that it is possible to predict the transient engine-out particulate number with datadriven methods trained on real-world experiments. This will pave the way for the implementation of virtual sensors for the transient engine emissions control on vehicles. On the other hand, many issues must be addressed and, while most of them have been outlined and a potential strategy to handle them has been proposed, some alternative strategies can be proposed. For example, possible improvements might derive from considering approaches as active learning, to reduce the number of training points required and domain invariant models, in order to provide an increased robustness against data shift [41]. It is of paramount importance, however, to underline the impact of domain knowledge in this research, for making the problem tractable, both in terms of feature generation and target preprocessing, which would have been otherwise intractable with direct application of off-the-shelf AI models.

Contact Information

Leonardo Pulga, Ph.D.

corresponding author

Mail: leonardo.pulga@naisengineering.com

Address: Via Maria Callas, 4, 40131 Bologna BO, Italy

Abbreviations

CNN - Convolutional Neural Network
CO - Carbon Monoxide
CO2 - Carbon Dioxide
CPU - Central Processing Unit
CTB - CatBoost
ECU - Engine Control Unit
GDI - Gasoline Direct Injection
GPU - Graphics Processing Unit
HC - Unburned Hydrocarbon
HEGO - Heated Exhaust Gas Oxygen Sensor
MAE - Mean Absolute Error
mRMR - minimum Redundancy Maximum Relevance
NN - Neural Network
NOX - Nitrogen Oxides
OBD - On-Board Diagnostic Program
PM - Particulate Matter

PNxx - Particle Number (xx minimum diameter)
R2 - Coefficient of Determination
RDE - Real Driving Emission
RF - Random Forest
RMSE - Root Mean Squared Error
RNN - Recurrent Neural Network
ROC-AUC - Area Under the Receiver Operating Characteristic Curve
SI - Spark Ignited
UEGO - Universal Exhaust Gas Oxygen Sensor
WLTP - Worldwide Harmonized Light Vehicles
XGB - XGBoost

References

1 Samaras, Z., Kontses, A., Dimaratos, A., Kontses, D. et al., “A European Regulatory Perspective towards a Euro 7 Proposal,” SAE Int. J. Adv. & Curr. Prac. in Mobility 5, no. 3 (2023): 998-1011, doi:https://doi.org/10.4271/2022-37-0032.

2 Meier, F., Schrangl, P., and del Re, L., “Reduction of Transient Soot Emissions of a Production Diesel Engine Using a Fast Soot Sensor and Closed Loop Control,” IFACPap. 52, no. 5 (2019): 159-164, doi:https://doi.org/10.1016/j.ifacol.2019.09.026.

3 Moser, M., Kipping, S., Higuchi, K., and Hirayama, H., “Machine-Learned Emission Model for Diesel Exhaust OnBoard Diagnostics and Data Flow Processor as Enabler,” SAE Technical Paper 2021-01-5108 (2021), doi:https://doi.org/10.4271/2021-01-5108.

4 Brusa, A., Giovannardi, E., Barichello, M., and Cavina, N., “Comparative Evaluation of Data-Driven Approaches to Develop an Engine Surrogate Model for NOx Engine-Out Emissions under Steady-State and Transient Conditions,” Energies 15, 21 (2022): 8088, doi:https://doi.org/10.3390/en15218088.

5 Leach, F. et al., “Particulate Emissions from a Highly Boosted Gasoline Direct Injection Engine,” Int. J. Engine Res. 19, no. 3 (2018): 347-359, doi:https://doi.org/10.1177/1468087417710583.

6 Aikawa, K. and Jetter, J.J., “Impact of Gasoline Composition on Particulate Matter Emissions from a Direct-Injection Gasoline Engine: Applicability of the Particulate Matter Index,” Int. J. Engine Res. 15, no. 3 (2014): 298-306, doi:https://doi.org/10.1177/1468087413481216.

7 Zhang, S. and McMahon, W., “Particulate Emissions for LEV II Light-Duty Gasoline Direct Injection Vehicles,” SAE Int. J. Fuels Lubr. 5, no. 2 (2012): 637-646, doi:https://doi.org/10.4271/2012-01-0442.

8 Leach, F., Lewis, A., Akehurst, S., Turner, J. et al., “Sub23 nm Particulate Emissions from a Highly Boosted GDI Engine,” SAE Technical Paper 2019-24-0153 (2019), doi:https://doi.org/10.4271/2019-24-0153.

9 Altuğ, K.B. and Küçük, S.E., “Predicting Tailpipe NOx Emission Using Supervised Learning Algorithms,” accessed April 1, 2023, https://ieeexplore.ieee.org/abstract/document/8932775/.

10 Kakaee, A.-H., Rahnama, P., Paykani, A., and Mashadi, B., “Combining Artificial Neural Network and Multi-Objective Optimization to Reduce a Heavy-Duty Diesel Engine Emissions and Fuel Consumption,” J. Cent. South Univ. 22, no. 11 (2015): 4235-4245, doi:https://doi.org/10.1007/s11771015-2972-1.

11 Khurana, S., Saxena, S., Jain, S., and Dixit, A., “Predictive Modeling of Engine Emissions Using Machine Learning: A Review,” Mater. Today Proc. 38 (2021): 280-284.

12 Shahpouri, S., Norouzi, A., Hayduk, C., Rezaei, R. et al., “Hybrid Machine Learning Approaches and a Systematic Model Selection Process for Predicting Soot Emissions in Compression Ignition Engines,” Energies 14, no. 23 (2021): 7865, doi:https://doi.org/10.3390/en14237865.

13 Taglialatela, F., Lavorgna, M., Di Iorio, S., Mancaruso, E. et al., “Real Time Prediction of Particle Sizing at the Exhaust of a Diesel Engine by Using a Neural Network Model,” SAE Int. J. Engines 10, no. 4 (2017): 2202-2208, doi:https://doi.org/10.4271/2017-24-0051.

14 Pulga, L., Bianchi, G.M., Falfari, S., and Forte, C., “A Machine Learning Methodology for Improving the Accuracy of Laminar Flame Simulations with Reduced Chemical Kinetics Mechanisms,” Combust. Flame 216 (2020): 72-81, doi:https://doi.org/10.1016/j.combustflame.2020.02.021.

15 Larsson, T., Vermeire, F., and Verhelst, S., “Machine Learning for Fuel Property Predictions: A Multi-Task and Transfer Learning Approach,” SAE Technical Paper 2023-010337 (2023), doi:https://doi.org/10.4271/2023-01-0337.

16 Cruz-Peragón, F., Torres-Jiménez, E., Lešnik, L., and Armas, O., “Methodology Improvements to Simulate Performance and Emissions of Engine Transient Cycles from Stationary Operating Modes: A Case Study Applied to Biofuels,” Fuel 312 (2022): 122977, doi:https://doi.org/10.1016/j.fuel.2021.122977.

17 Sarkar, B., Gundlapally, S.R., Koutsivitis, P., and Wahiduzzaman, S., “Performance Evaluation of Neural Networks in Modeling Exhaust Gas Aftertreatment Reactors,” Chem. Eng. J. 433 (2022): 134366, doi:https://doi.org/10.1016/j.cej.2021.134366.

18 Raza, M., Chen, L., Leach, F., and Ding, S., “A Review of Particulate Number (PN) Emissions from Gasoline Direct Injection (GDI) Engines and Their Control Techniques,” Energies 11, no. 6 (2018): 1417, doi:https://doi.org/10.3390/en11061417.

19 Giechaskiel, B. et al., “Particle Number Measurements Directly from the Tailpipe for Type Approval of Heavy-Duty Engines,” Appl. Sci. 9, no. 20 (2019): 4418, doi:https://doi.org/10.3390/app9204418.

20 Fridman, L., Brown, D.E., Angell, W., Abdić, I. et al., “Automated Synchronization of Driving Data Using Vibration and Steering Events,” arXiv.org, October 21, 2015, accessed April 1, 2023, https://arxiv.org/abs/1510.06113v2.
21 Papaioannou, N., Fang, X., Leach, F., Lewis, A. et al., “A Random Forest Algorithmic Approach to Predicting Particulate Emissions from a Highly Boosted GDI Engine,” SAE Technical Paper 2021-24-0076 (2021), doi:https://doi.org/10.4271/2021-24-0076.
22 Hu, Z., Lu, Z., Song, B., and Quan, Y., “Impact of Test Cycle on Mass, Number and Particle Size Distribution of Particulates Emitted from Gasoline Direct Injection Vehicles,” Sci. Total Environ. 762 (2021): 143128, doi:https://doi.org/10.1016/j.scitotenv.2020.143128.
23 Zhao, Y. et al., “A Comparative Study on Unsupervised Anomaly Detection for Time Series: Experiments and Analysis,” arXiv.org, September 10, 2022, accessed April 1, 2023, https://arxiv.org/abs/2209.04635v1.
24 Monroe, R., Studzinski, W., Parsons, J.L., La, C. et al., “Engine Particulate Emissions as a Function of Gasoline Deposit Control Additive,” SAE Int. J. Fuels Lubr. 14, no. 1 (2021): 3-11, doi:https://doi.org/10.4271/04-14-01-0001.
25 Etikyala, S., Koopmans, L., and Dahlander, P., “History Effect on Particulate Emissions in a Gasoline Direct Injection Engine,” SAE Int. J. Engines 15, no. 3 (2021): 445455, doi:https://doi.org/10.4271/03-15-03-0999.
26 Choi, Y., Yi, H., Oh, Y., and Park, S., “Effects of Engine Restart Strategy on Particle Number Emissions from a Hybrid Electric Vehicle Equipped with a Gasoline Direct Injection Engine,” Atmos. Environ. 253 (2021): 118359, doi:https://doi.org/10.1016/j.atmosenv.2021.118359.
27 Oh, C. and Cheng, W.K., “Assessment of Gasoline Direct Injection Engine Cold Start Particulate Emission Sources,”SAE Int. J. Engines 10, no. 4 (2017): 1556-1565, doi:https://doi.org/10.4271/2017-01-0795.
28 Yusuf, A.A. and Inambao, F.L., “Effect of Cold Start Emissions from Gasoline-Fueled Engines of Light-Duty Vehicles at Low and High Ambient Temperatures: Recent Trends,” Case Stud. Therm. Eng. 14 (2019): 100417, doi:https://doi.org/10.1016/j.csite.2019.100417.
29 Berthome, V., Chalet, D., and Hetet, J.-F., “Characterization of Particle Emissions of Turbocharged Direct Injection Gasoline Engine in Transients and Hot Start Conditions,” J. Therm. Sci. 30, no. 6 (2021): 2056-2070, doi:https://doi.org/10.1007/s11630-021-1420-9.
30 Berthome, V., Chalet, D., and Hetet, J.-F., “Impact of BlowBy Gas and Endgap Ring Position on the Variations of Particle Emissions in Gasoline Engines,” Energies 14, no. 22 (2021): 7492, doi:https://doi.org/10.3390/en14227492.
31 Dama, F. and Sinoquet, C., “Time Series Analysis and Modeling to Forecast: A Survey,” arXiv.org, March 31, 2021, accessed June 15, 2023, https://arxiv.org/abs/2104.00164v2.
32 TensorFlow Developers, “TensorFlow (v2.14.0-rc1),” Zenodo, 2023, https://doi.org/10.5281/zenodo.8306789.
33 Chen, T. and Guestrin, C., “XGBoost: A Scalable Tree Boosting System,” in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ‘16 (New York: ACM, 2016), 785-794, doi: https://doi.org/10.1145/2939672.2939785.
34 Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A.V. et al., “CatBoost: Unbiased Boosting with Categorical Features,” arXiv.org, June 28, 2017, accessed April 1, 2023, https://arxiv.org/abs/1706.09516v5.
35 Rahaman, R. and Thiery, A.H., “Uncertainty Quantification and Deep Ensembles,” arXiv.org, July 17, 2020, accessed April 1, 2023, https://arxiv.org/abs/2007.08792v4.
36 Angelopoulos, A.N. and Bates, S., “A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification,” arXiv.org, July 15, 2021, accessed April 1, 2023, https://arxiv.org/abs/2107.07511v6.
37 Avci, M.Y., Li, Z., Fan, Q., Huang, S. et al., “Quantifying the Uncertainty of Neural Networks Using Monte Carlo Dropout for Deep Learning Based Quantitative MRI,” arXiv. org, December 2, 2021, accessed April 1, 2023, https://arxiv.org/abs/2112.01587v1.
38 Malinin, A., Prokhorenkova, L., and Ustimenko, A., “Uncertainty in Gradient Boosting via Ensembles,” arXiv.org, June 18, 2020, accessed April 1, 2023, https://arxiv.org/abs/2006.10562v4.
39 Zhao, Z., Anand, R., and Wang, M., “Maximum Relevance and Minimum Redundancy Feature Selection Methods for a Marketing Machine Learning Platform,” accessed April 1, 2023, https://ieeexplore.ieee.org/document/8964172.
40 Lundberg, S.M. and Lee, S.-I., “A Unified Approach to Interpreting Model Predictions,” Adv. Neural Inf. Process. Syst. 30 (2017): 4768-4777, accessed April 1, 2023, https://proceedings.neurips.cc/paper_files/paper/2017/hash/8a20a8621978632d76c43dfd28b67767-Abstract.html.
41 Strazzera, L., Gori, V., and Veneri, G., “DANNTe: A Case Study of a Turbo-Machinery Sensor Virtualization under Domain Shift,” arXiv.org, January 11, 2022, accessed April 1, 2023, https://arxiv.org/abs/2201.03850v1.

Appendices

Appendix A: Optimized

XGBoost	
Max_depth	15
Colsample_bytree	0.5
Min_child_weight	20
Gamma	1
Lambda	20
Alpha	0.01
Subsample	0.8
Eta	0.1
Tree_method	Gpu_hist
#iterations	10,000
Early stopping	100
CatBoost	
Iterations	1000
Learning_rate	0.1
Early_stopping	20
L2_leaf_reg	3

	RNN	CNN (1D)
Base layer	GRU	Conv1d
# layers	3	6
# dense layers	1	3
# params	0.15 M	0.25 M
Optimizer	Adam	Adam
Loss function	Sparse-categorical cross-entropy/MSE	Sparse-categorical cross-entropy/MSE
Batch size	420	420
Initial learning rate	1e−3	1e−3
Scheduler	Cosine annealing	Cosine annealing
Regularizer	L2 +BatchNorm +Dropout (rate = 0.1)	L2 +BatchNorm +Dropout (rate = 0.1)
Deep NN		
Base layer	Dense	
# layers	2	
# params	0.1 M	
Optimizer	Adam	
Loss function	Categorical cross_entropy/MSE	
Batch size	32	
Initial learning rate	0.001	
Scheduler	Cosine annealing	
Regularizer	L2 + BatchNorm + Dropout (rate = 0.2)	
Random forest		
N_estimators	300	
Min_samples_split	10	
min_samples_leaf	5	

Appendix B: Features Calculated from Time-windows

Synthetic features calculated for each channel in windows of 1, 2, 5 s:

1. Min value
2. Max value

3. Mean value
4. Quantiles (0.25, 0.5, 0.9)
5. Skewness
6. Kurtosis
7. Standard deviation
8. Integral
9. Difference between max and min value
10. Difference between index of max and min value
11. Last value
12. Min value of the derivative
13. Max value of the derivative
14. Index of the min value
15. Index of the max value
16. Sign of the difference between min value index and max value index

Appendix C: Detailed Results on Validation Cycles

FIGURE C.1 Cycle #1.

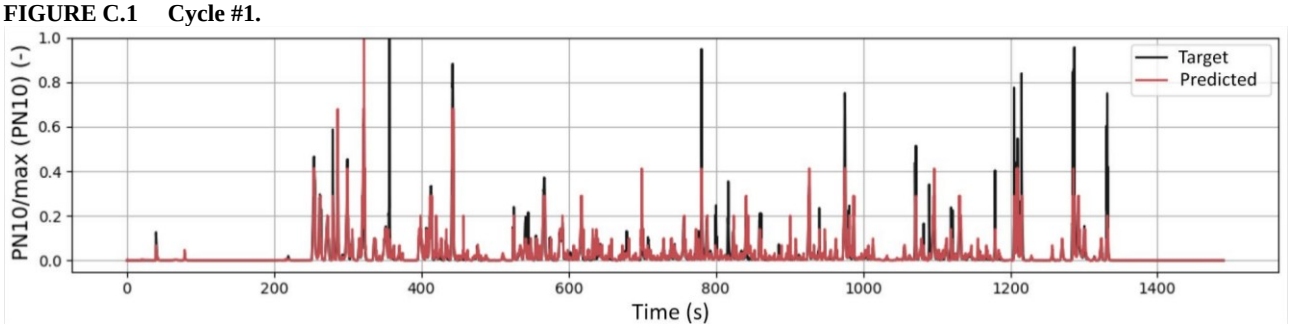


FIGURE C.2 Cycle #2.

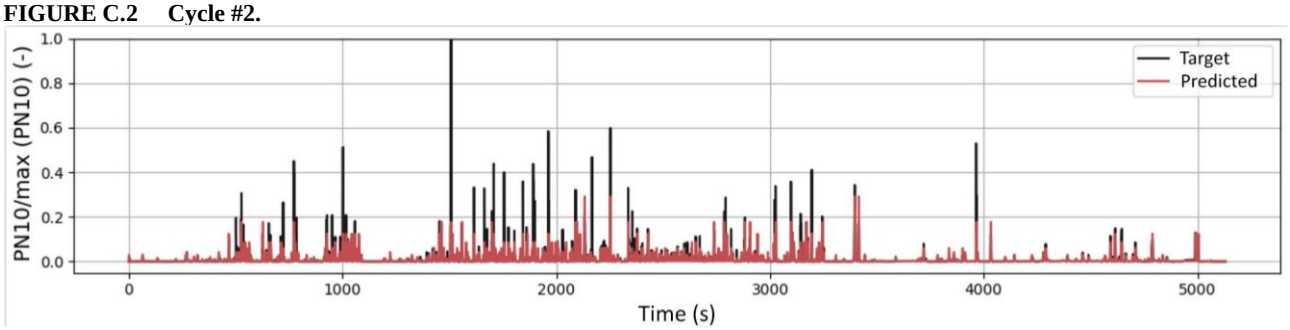


FIGURE C.3 Cycle #3.

FIGURE C.4 Cycle #4.

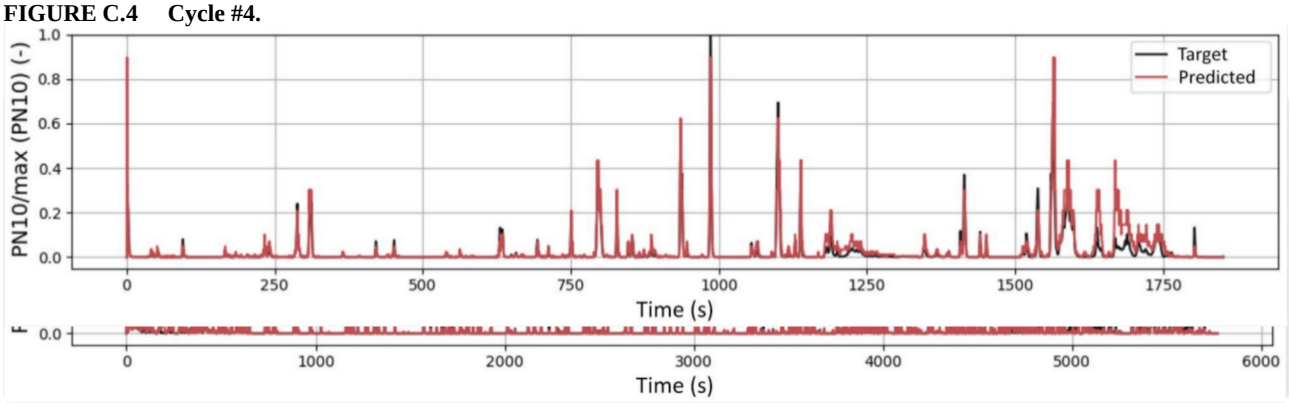


FIGURE C.5 Cycle #5.

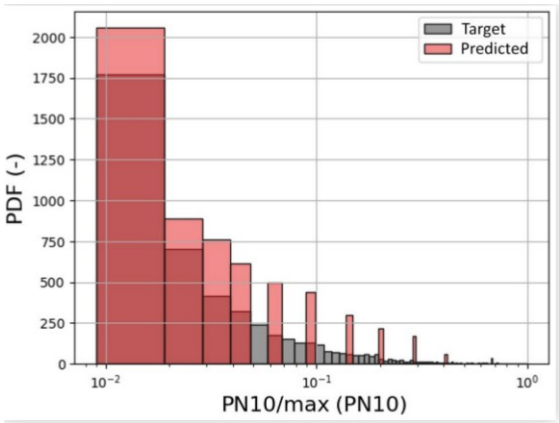


FIGURE C.6 Cycle #1 distribution (evident effect of binning strategy).

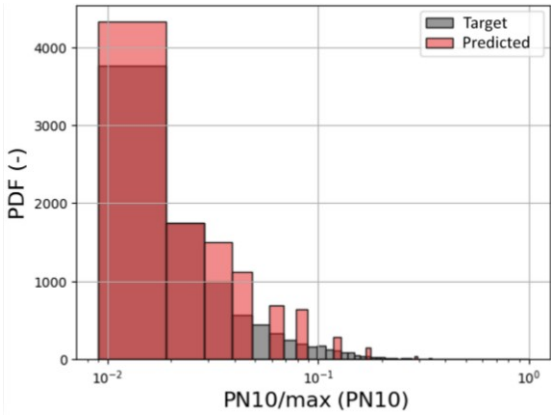


FIGURE C.7 Cycle #2 distribution (evident effect of binning strategy).

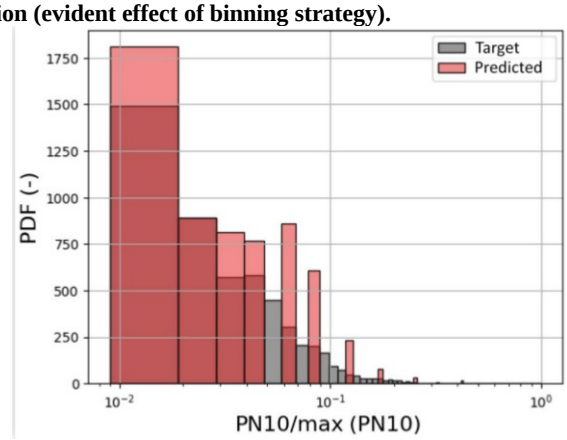


FIGURE C.8 Cycle #3 distribution (evident effect of binning strategy).

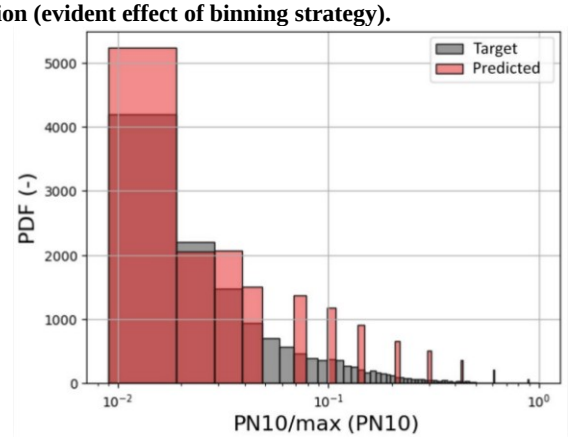


FIGURE C.9 Cycle #4 distribution (evident effect of binning strategy).

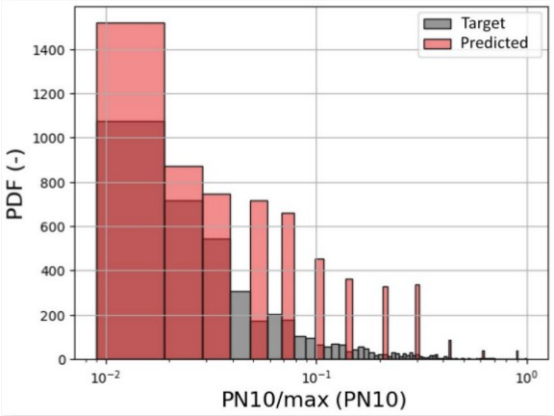
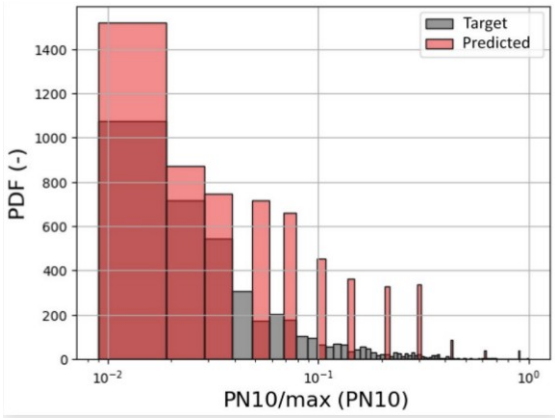


FIGURE C.10 Cycle #5 distribution (evident effect of binning strategy).


Target
Predicted
PDF (-)
0
200
400
600
800
1000
1200
1400
10^{-2}
10^{-1}
10^{0}
PN10/max (PN10)