Data-Driven Open-Set Fault Classification of Residual Data Using Bayesian Filtering

Daniel Jung

***Abstract*— Data-driven fault classification in industrial applications is complicated by unknown fault classes and limited training data. In addition, different faults can have similar effects on sensor outputs resulting in fault classification ambiguities, i.e., multiple fault hypotheses can explain the data. One solution is to identify and rank all plausible fault classes that give useful information, for example, at a workshop when performing troubleshooting. A probabilistic fault classification algorithm is proposed for residual data classification combining the Weibull-calibrated one-class support vector machines for fault class modeling and Bayesian filtering for time-series analysis. The fault classifier ranks different fault classes and can identify sequences from unknown fault realizations, i.e., faults not represented in training data. Real residual data computed from sensor data and model analysis of an internal combustion engine are used as a case study illustrating the usefulness of the proposed method.**

***Index Terms*— Fault classification, hybrid fault diagnosis, machine learning, open-set classification, support vector machines.**

I. Introduction

IN THE automotive industry, onboard diagnosis (OBD) systems have been used for emission-related monitoring for decades. New applications, such as predictive maintenance and assisted troubleshooting at the workshop, are important to improve reliability and reduce system downtime in order to increase customer value. Connected vehicles and cloud computation capacities have put a focus on machine learning methods for fault diagnosis and prognostics.

Fault diagnosis of industrial systems is often conducted by analysis and classification of time-series data collected during system operation, for example, sensor data or computed residuals [1], [2]. When designing a fault diagnosis system, there are often many different types of faults that can occur in the system and should be detected. Even though there are tools to systematically identify all these fault classes early in the system development phase (see [3]), it is still a difficult task, especially for large-scale or complex systems. Therefore, there can be unknown faults that are not considered when training the diagnosis system [4].

Another complicating factor in data-driven fault diagnosis is collecting representative training data from all relevant faults. Data collection is an expensive and time-consuming process and not feasible in many applications [5], [6]. Especially, since many faults do not occur until after years of operation. Therefore, training data are not representative of all fault scenarios, meaning that a diagnosis system must be able to identify both known and unknown fault scenarios.

Different faults can have similar effects on system dynamics, resulting in fault classification ambiguities. Therefore, it is not desirable that a data-driven classifier only selects one fault class since the true fault could be missed, but it should instead identify and rank all plausible fault classes [4]. This type of information is useful, for example, at the workshop to support a technician during troubleshooting [7]. For reliable fault classification, it is also necessary to identify data sets with unknown faults, i.e., fault scenarios not represented in training data, since these cases need special attention to improve classification performance over time [8].

A. Problem Formulation

The objective of this brief is to develop a data-driven fault classification algorithm for time-series data, for example, sensor data or model-based residuals, that identify and rank fault hypotheses (fault classes). It is assumed that training data are limited and not representative of all fault realizations. Machine learning algorithms that assume all data classes are known, and representative training data are available, are not expected to give reliable outputs, especially in fault scenarios where data deviate too much from the training data. A fault classifier should, therefore, be able to identify when there are data sequences with unknown fault scenarios, i.e., sequences that do not resemble training data.

Fault diagnosis of an internal combustion engine is used as a case study. The fault scenarios cover different types of engine faults, including sensor faults, leakages, and air filter clogging. As input to the data-driven fault classification algorithm, a set of residual data is computed from a physically based model and real data from different fault scenarios collected from the engine test rig [9] (see Fig. 1).

B. Related Research

Data-driven monitoring and fault diagnosis of internal combustion engines are investigated in, for example, [10]. A data-driven classifier approach for fault diagnosis of an electric throttle control system is proposed in [5] where incremental learning is used to improve classification performance over time. In [11], an ensemble approach for automotive fault classification of both known and unknown faults in time-series data is developed by combining multiple machine learning

Manuscript received January 17, 2020; accepted May 22, 2020. Date of publication June 2, 2020; date of current version August 6, 2020. Manuscript received in final form May 23, 2020. Recommended by Associate Editor P. F. F. Odgaard.

The author is with the Department of Electrical Engineering, Linköping University, 581 83 Linköping, Sweden (e-mail: daniel.jung@liu.se).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2020.2997648

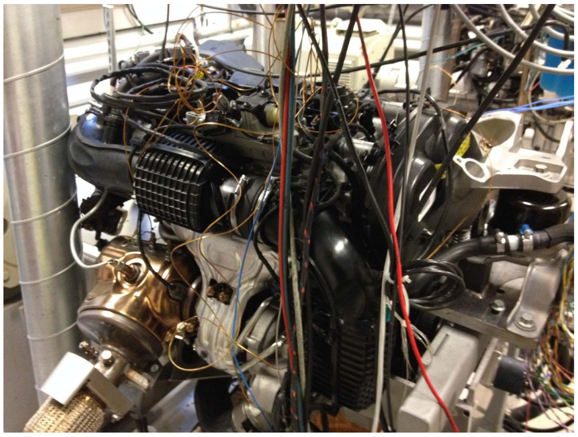


Fig. 1. Picture shows the engine test bench that is used for data collection.

methods for classification. A two-step fault classification approach to handle unknown faults in an electronic system using Gaussian mixture models and k-means is proposed in [12]. With respect to the mentioned work, an incremental probabilistic fault classification method is proposed that ranks different faults using model-based residuals as input.

One solution to limited training data is to use a physically based model of the system to generate features, for example, residuals [4]. Fault diagnosis methods for automotive applications, combining model-based and data-driven methods, are proposed in, for example, [13]–[16]. Research highlights the benefits of bridging and combining model-based and data-driven methods for fault diagnosis instead of only focusing on one of them [17].

In [18], both sensor data and residual data are used as an input to a tree augmented naive Bayes fault classifier. In [6], a conditional Gaussian network is proposed to handle both known and unknown fault classes. In [19], feature selection using neural networks is applied before training the fault classifiers. In [4], a hybrid diagnosis system design is proposed, which combines model-based fault isolation with support vector data description (SVDD) anomaly classifiers to rank the different fault hypotheses. In [20], model-based residuals and sensor data are used as inputs to a Bayesian network to perform fault classification, and in [21], model data features are extracted and fed into a neural network classifier. In [22], a hybrid approach combining model-based residuals with hidden Markov models and Bayesian methods is used to classify unknown faults.

Another related research topic is the open-set recognition problem in computer vision where data can belong to unknown classes not covered by the training data [8]. Unknown classes are further categorized into known unknowns and unknown unknowns, where the second case corresponds to the unknown faults considered in this brief. Different algorithms have been proposed to solve the open-set recognition problem, for example, Weibull-calibrated support vector machines [23] and extreme value machines [24].

This brief is based on previous research in [4] and [25]. The main contribution, with respect to mentioned works, is a data-driven probabilistic classification algorithm of time-series data combining Weibull-calibrated one-class support vector machines (OSVMs) [26] and Bayesian filtering and smoothing [27] to improve classification performance and ranking of fault hypotheses.

II. Fault Class Modeling Using Open-Set Classification

In real-life applications where training data are limited, it is important that a classifier can identify residual data that cannot be explained by any of the known fault classes, i.e., data that significantly deviate from training data.

A. Using One-Class Classifiers for Modeling Fault Classes

Let m be the number of available residuals and $\bar{r} = (r_1, r_2, \ldots, r_m)$ is a sample of all residuals outputs. The purpose of modeling different fault classes is to identify which fault hypotheses can explain the observed data $\bar{r}$. One-class classifiers are suitable for modeling fault classes since each class can be modeled individually.

There are multiple methods proposed for one-class classification, for example, probabilistic models, OSVMs, and isolation forests (iForests) [28]. Probabilistic models use probability distributions to model data from one class and detect outliers, with respect to that class, when the likelihood of a sample is small (see [6]). Nonprobabilistic models, such as OSVM and iForests, model a decision boundary that encapsulates training data to determine whether new data can be explained by that class or not.

Since training data are assumed to be limited, the distribution of data is not expected to be representative of each fault class. Training data might have been collected through experiments to cover different fault realizations but not to be representative of the actual distribution of fault realizations. The objective is to identify plausible fault hypotheses, regardless of how likely they are. Therefore, a nonprobabilistic approach is used to model which observations $\bar{r}$ can be explained by each fault class. Training data from each known class are modeled using a decision function representing the maximum distance from any training data where a new sample could be explained by that class, called a compact abating probability (CAP) model [23]. Unknown fault classes are identified when data are significantly deviating from training data. Fig. 2 shows a set of CAP models and the problem of classifying a set of new data when it significantly deviates from the known fault classes. It is shown in [23] that an OSVM classifier with a radial basis function (RBF) kernel yields a CAP model.

B. One-Class Support Vector Machines

There are two similar approaches for designing OSVM classifiers, referred to as ν-SVM [29] and SVDD [30], respectively, where ν-SVM is used in this brief. An OSVM classifier uses the kernel trick to model a decision boundary that encapsulates data from that class [31]. This is shown in Fig. 2 where the black lines represent the decision boundaries of two OSVM classifiers modeling data from Classes 1 and 2, respectively.

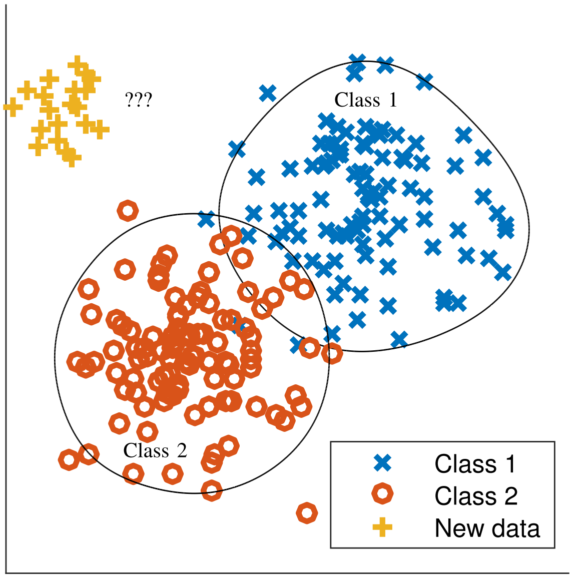


Fig. 2. Illustration of two CAP models using OSVM classifiers to model two known fault classes. The new data cannot be explained by any of the known fault classes and is considered to belong to an unknown fault class.

The OSVM classifier computes a score function, when evaluating each new sample, that is positive when belonging to the nominal class or negative if it is considered an outlier, i.e., not belonging to that class. The OSVM classifier evaluates each sample of residual data independently, meaning that time-series information of the residuals is ignored.

In previous work [4], a set of OSVM classifiers is used to model the CAP models from known fault classes. When classifying new data, each fault class is ranked based on how many samples that are associated with that fault class. Note that a sample can be explained by multiple fault classes. As new data are collected from different faults and correctly classified, the OSVM classifiers are updated accordingly to improve performance over time. Incremental training can be applied to reduce the computational cost when new data are collected (see [32]).

C. Weibull-Calibrated OSVM

Even though the OSVM classifier is a CAP model, its decision boundary depends on the distribution of the support vectors. Therefore, is it is relevant to have a measure of the probability that a sample $\bar{r}$ can be explained by fault class f^l, here denoted $P(\bar{r} \in f^l)$. There are some proposed methods to translate the score computed by an SVM into a probability, for example, Platt scaling [33] or Weibull-calibrated SVM [23]. The advantages of Weibull-calibrated SVM with respect to Platt scaling are discussed in, e.g., [23]. However, the Weibull-calibrated SVM classifier is a multiclass classifier that outputs one fault class, which could be the unknown class. Since the objective here is to model each fault class separately, to identify all plausible fault hypotheses, a Weibull-calibrated OSVM method, proposed in [26], is used called P_I-OSVM.

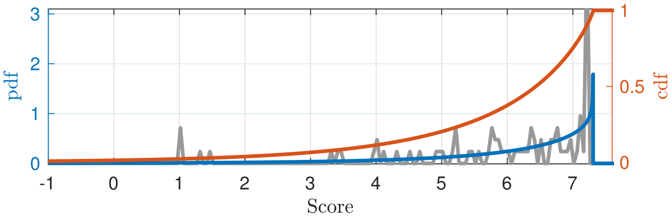


Fig. 3. PDF and cdf of the parameterized reverse Weibull distribution fit to the score values of the support vectors of a OSVM classifier.

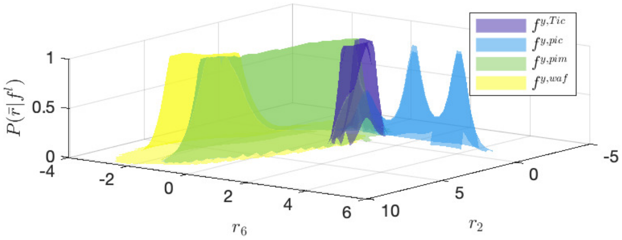


Fig. 4. Set of P_I-OSVM models are parameterized for two residuals. It shows the probability of inclusion for each fault class.

In [23] and [26], a statistical extreme value theory is applied when proposing the P_I-OSVM classifier to model data from each class. The output scores from the support vectors of the OSVM are modeled to be reverse Weibull distributed. The corresponding cdf of the reverse Weibull distribution measures the probability that a new sample can be explained by that fault class, referred to as probability of inclusion in [26]. An example is shown in Fig. 3 showing the distribution of the OSVM score for a set of data, a reverse Weibull distribution fit to the score values, and the corresponding cdf.

The reverse Weibull cdf parameterized for the OSVM score value $g(\bar{r})$ is given by

$$P(\bar{r} \in f^l) = \begin{cases} e^{-\left(\frac{-g(\bar{r})+\nu_l}{\lambda_l}\right)^{\kappa_l}}, & \text{if } -g(\bar{r})+\nu_l \geq 0 \\ 1, & \text{otherwise} \end{cases} \tag{1}$$

where $\nu_l, \lambda_l, \kappa_l \geq 0$ are fit parameters for fault class f^l. For (1), denoted P_I-OSVM, to be a CAP model, the probability $P_{f^l}(\bar{r})$ is thresholded by a parameter δ, which represents when the Euclidean distance from a new sample to the training data is too large. An example of P_I-OSVM models $P(\bar{r} \in f^l)$ for a set of fault modes f^l is shown for a two residual output case in Fig. 4 where different fault scenarios in training data result in different residual outputs. The z-axis represents the conditional probability (1) that each fault class can explain the residual outputs.

III. Fault Classification of Time-Series Data Using Bayesian Filtering and Smoothing

One approach to classify each sample $\bar{r}_t$, where subscript t is used to denote time index, using the set of P_I-OSVM models is to select the class f^l with the highest probability, i.e., $\arg\max_{f^l} P(\bar{r} \in f^l)$. The probability of a sample to belong to an unknown fault class, denoted f^x, is difficult to model without prior information. In [6] and [23], there are no probability models of the unknown fault class f^x. Instead, f^x is selected when the probabilities of all known fault classes are below some threshold. Here, $P(\bar{r} \in f^x) = \delta$ is modeled, which is equal to the threshold of the corresponding CAP models, i.e., samples that do not belong to a known fault class are more likely to come from an unknown fault class.

Since a fault is often present during a longer time interval, Bayesian filtering and smoothing are applied here to improve classification performance by weighing in information from the consecutive samples [27]. This is relevant if there are multiple fault classes that can explain the same observations.

The probability that the system is changing from one fault mode to another at time t is modeled using a transition matrix $\Pi \in \mathbb{R}^{n+1\times n+1}$, where n is the number of known fault classes and plus one for the unknown fault class. Let $\Pi_{l,k}$ denote the element representing the probability that the system changes from mode f_{t-1}^l to f_t^k at time t. Faults are rare events and the probability that the system is changing mode is considered small compared with the system staying in the same mode.

The pdf $p(\bar{r}_t|f^l)$ of the residual output $\bar{r}_t$ given fault class f^l is unknown. However, to be able to use the P_I-OSVM models in a Bayesian framework, it is assumed here that $p(\bar{r}_t|f^l)$ is large when $P(\bar{r} \in f^l)$ is large. Then, the pdf is modeled as $p(\bar{r}_t|f^l) \propto P(\bar{r} \in f^l)$.

A Bayesian filter evaluating the probability of each fault class f_t^l at time t can be computed sequentially as

$$p\big(f_t^l|\bar{r}_{1:t}\big) \propto p\big(\bar{r}_t|f_t^l\big)\sum_{k=1}^{n+1}\Pi_{k,l}\,p\big(f_{t-1}^k|\bar{r}_{1:t-1}\big) \qquad (2)$$

where the prior distribution $p(f_0^l|\bar{r}_0) = p(f_0^l)$ and the probabilities of all modes are normalized, i.e., $\sum_{k=1}^{n+1} p(f_t^l|\bar{r}_{1:t}) = 1$.

The sequential formulation of Bayesian filtering is suitable for online computations where class probabilities are computed based on previous samples. A workshop would be able to download logged data and perform off-line computations on the whole data batch. Bayesian smoothing can be applied to a batch of T samples by performing an additional backward filtering after (2) as

$$p\big(f_t^l|\bar{r}_{1:T}\big) \propto p\big(f_t^l|\bar{r}_{1:t}\big)\sum_{k=1}^{n+1}\Pi_{l,k}\,p\big(f_{t+1}^k|\bar{r}_{1:T}\big) \qquad (3)$$

followed by a normalization to compute the class probabilities. Combining the P_I-OSVM classifiers and Bayesian filtering or smoothing gives a systematic method to identify and rank the different fault hypotheses based on how many samples in a data batch are classified as each fault class [4].

IV. Case Study

The case study in this brief is the same internal combustion engine system as considered in [9] and [25]. Sensor data have been collected from the engine test bed, including nominal system behavior (NF: No Fault) and seven different single-fault scenarios: air filter clogging f^{paf}, leakages at the air filter f^{Waf} and at the throttle f^{Wth}, and four different sensor faults $f^{y,\text{Tic}}$, $f^{y,\text{pic}}$, $f^{y,\text{pim}}$, and $f^{y,\text{Waf}}$. Table I summarizes the seven fault scenarios. The locations of the four sensors are shown in Fig. 5 where y^{Tic} and y^{pic} measure the temperature and pressure after the intercooler, y^{pim} measures the pressure at the intake manifold, and y^{Waf} measures the air flow through the air filter.

A mathematical model is available describing the air flow through an internal combustion engine. The model has been used in previous works for residual generation (see [4]), and the model structure is similar to the model described in [35], which is based on six control volumes and mass and energy flows given by restrictions (see Fig. 5).

TABLE I

Summary of Fault Scenarios Collected From Engine Test Rig

Fault	Description
f^{paf}	Air filter clogging
f^{Waf}	Leakage after air filter
f^{Wth}	Leakage before throttle
$f^{y,Tic}$	Intermittent fault in sensor measuring temperature at intercooler
$f^{y,pic}$	Intermittent fault in sensor measuring pressure at intercooler
$f^{y,pim}$	Intermittent fault in sensor measuring intake manifold pressure
$f^{y,Waf}$	Intermittent fault in sensor measuring air flow through air filter

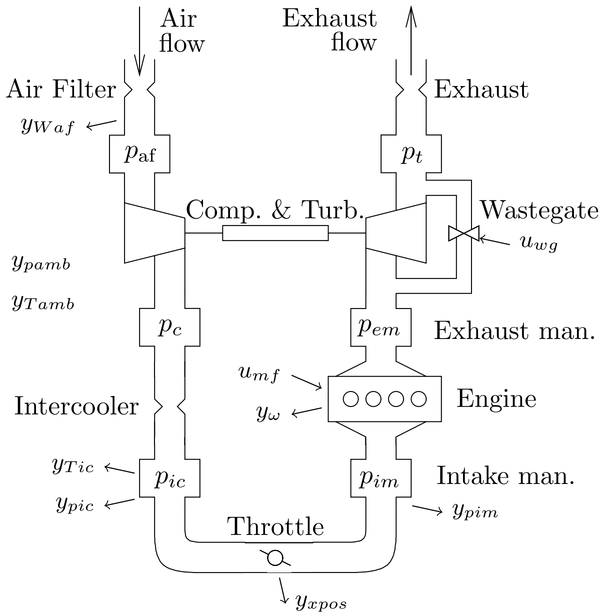


Fig. 5. Schematic of the model of the air flow through the model. This figure is used with permission from [34].

Nine residual generators $\bar{r} = (r_1, \ldots, r_9)$ have been generated in [25] from the model, using the Fault Diagnosis Toolbox in MATLAB [36]. A residual is a function comparing two different estimates of the same quantity to detect inconsistencies, for example, between a sensor value and a model prediction of the measured quantity. An illustrative example is shown in Fig. 6, where u represents control signals, f is faults, y is sensor data, $\hat{y}$ is model predictions, and $r = y - \hat{y}$ is the residual.

The internal combustion engine is an example of a system that operates at many different operating conditions, including transients. The residuals are designed to, ideally, filter out the system dynamics while being sensitive to faults. Even though both sensor data and residuals can be used as inputs to a classifier, only residual data will be used here.

The nine residuals are evaluated using data from different fault scenarios collected from the engine test rig.[1] The data set contains 20 276 samples, including nominal and faulty data.

[1] Residual data are available in the Fault Diagnosis Toolbox [36] that can be downloaded from https://faultdiagnosistoolbox.github.io. The selected residual subset used in this brief is described in [25].

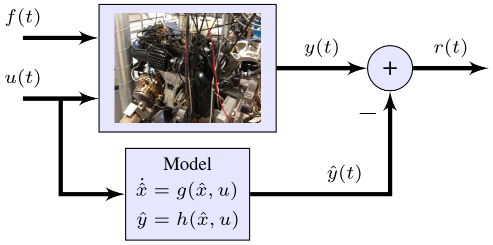


Fig. 6. Example of a residual $r(t)$ comparing measurements from the system $y(t)$ with model predictions $\hat{y}(t)$.

To evaluate the situation with limited training data, only 10% of the residual data, both nominal operation and from different fault scenarios, are used as training data and the remaining set is used for validation. Fig. 7 shows data from each residual from both nominal (NF) and seven fault classes (blue data) and the corresponding fault label (red data). The air filter clogging f^{paf} and leakages f^{Waf} and f^{Wth} have been collected from persistent fault scenarios, whereas sensor fault data are collected from intermittent sensor faults as shown in the figure.

V. Experimental Results

A P_I-OSVM model (1) is calibrated for each class in the training data and a decision threshold $\delta = 5\%$ is selected for each model. The OSVM classifier, used in the P_I-OSVM models, is implemented using the function `fitcsvm` in `MATLAB` and its kernel parameters are fit to training data using a subsampling heuristic [37]. In this analysis, fault detection and classification are performed simultaneously and the fault-free class NF is included as a fault class.

Validation data from each fault scenario in Fig. 7 are used to evaluate the similarity between the models by analyzing how many samples can be explained by each fault class. Fig. 8 shows the percentage of data from each fault scenario that can be explained by each fault class. Samples that are not associated with any known fault class are classified as the unknown fault class f^x. Note that the sum of each column in Fig. 8 can exceed 100% since each sample can belong to multiple fault classes. A significant number of samples can be explained by more than one fault class, e.g., {NF, f^{paf}} and {f^{waf}, f^{wth}}, showing that the CAP models for the different fault classes are overlapping. It is also visible that the overlap is not symmetric between fault pairs. For example, 81% of the samples from fault scenario f^{paf} can also be explained by $f^{y,\text{pim}}$, but only 31% of the samples from $f^{y,\text{pim}}$ can be explained by f^{paf}.

The CAP models are useful to identify fault hypotheses, i.e., which fault classes could explain the residual data. However, each sample is classified independently of the others ignoring information from the time-series data. To improve fault classification performance, the next step is to take time-series information into consideration.

A. Classification Using Bayesian Filtering and Smoothing

The next step is to evaluate the benefits of applying Bayesian filtering and smoothing with respect to only sample-by-sample classification of residual data. First, sample-by-sample classification is performed where each sample $\bar{r}_t$ is associated with the fault class f^l with the highest probability $p(f_t^l|\bar{r}_t)$ at time t. Each fault class f^l is ranked during a fault scenario by counting how many samples are associated with that fault class, similar to what is used in [4] and [25].

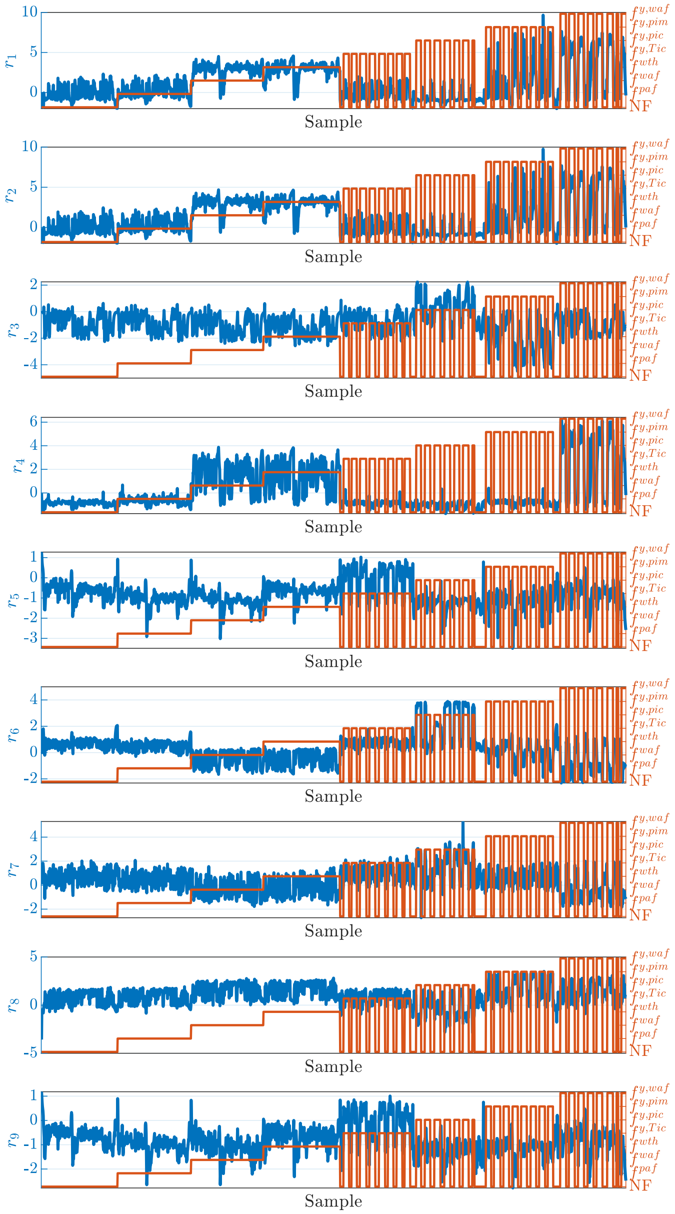
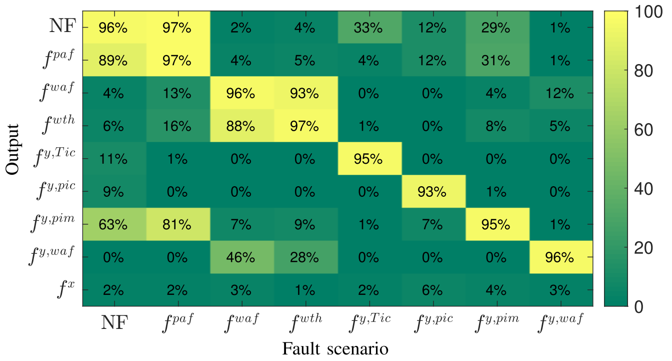


Fig. 7. Data from nine residuals collected from nominal system behavior (NF) and seven different faults. Each subplot shows one residual output where the blue curve is the residual output and the red curve is class label.

The distribution $p(f_t^l|\bar{r}_t)$ is evaluated using validation data where the a priori distribution $p(f_0^l)$ of all fault classes f^l are assumed equal and the results are shown in Fig. 9. It is highlighted in gray when each fault class is the true fault in the data set. Ideally, $p(f_t^l|\bar{r}_t) = 1$ when f^l is the true fault class and zero otherwise. Fig. 10 summarizes the classification performance when each sample $\bar{r}$ is classified as the fault class f^l with the highest probability $p(f_t^l|\bar{r}_t)$. Each element (k, l) in the matrix shows how many samples from a fault scenario with fault f^l are associated with fault class f^k. The evaluation in Fig. 8 shows that it is more difficult to correctly classify



Output	NF	f^{paf}	f^{waf}	f^{wth}	$f^{y,Tic}$	$f^{y,pic}$	$f^{y,pim}$	$f^{y,waf}$
NF	96%	97%	2%	4%	33%	12%	29%	1%
f^{paf}	89%	97%	4%	5%	4%	12%	31%	1%
f^{waf}	4%	13%	96%	93%	0%	0%	4%	12%
f^{wth}	6%	16%	88%	97%	1%	0%	8%	5%
$f^{y,Tic}$	11%	1%	0%	0%	95%	0%	0%	0%
$f^{y,pic}$	9%	0%	0%	0%	0%	93%	1%	0%
$f^{y,pim}$	63%	81%	7%	9%	1%	7%	95%	1%
$f^{y,waf}$	0%	0%	46%	28%	0%	0%	0%	96%
f^{x}	2%	2%	3%	1%	2%	6%	4%	3%

Fig. 8. Modeling fault classes using residual data from Fig. 7 and CAP models, here thresholded P_I-OSVM models. The overlap between fault classes is evaluated by counting the percentage of data that can be explained by each fault class. Samples not belonging to any known fault class belong to the unknown fault class f^x.

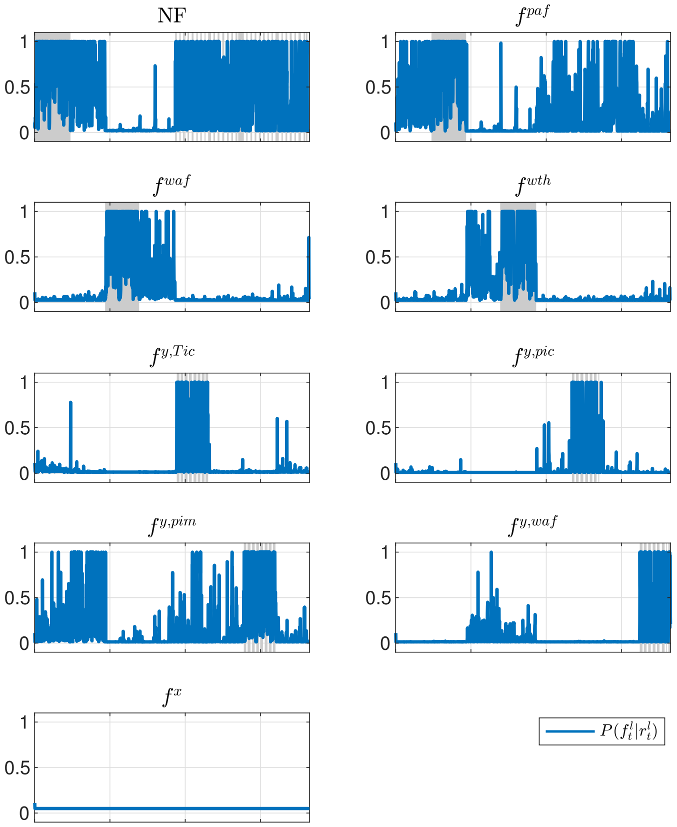
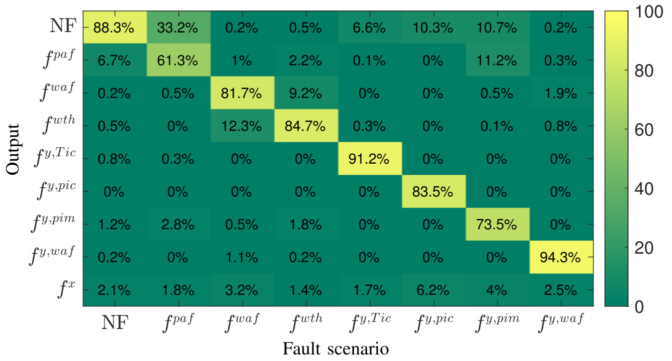


Fig. 9. Fault class probabilities $p(f_t^l|\bar{r}_t)$ from validation data in Fig. 7. The gray intervals represent when the corresponding fault class is the true one and the probability should be high and zero otherwise.

fault classes when the CAP-models are overlapping, in this case mainly {NF, f^{paf}} and {f^{waf}, f^{wth}}, respectively.

A comparison of filtered (2) and smoothed (3) estimates of class probabilities is shown in Fig. 11. Here, the transition probability between two different classes in Π is chosen experimentally as 1% and staying at the same class as $100-(n+1)\%$. Experiments show that a higher transition probability between fault classes results in bigger fluctuations in $p(f_t^l|\bar{r}_t)$, whereas a lower transition probability reduces the fluctuations but, sometimes, also requires more samples after a fault occurs before $p(f_t^l|\bar{r}_t)$ changes significantly. The different subplots in Fig. 11 show the computed probability of each fault class where the highlighted gray areas show when the fault is present and the ranking should be high and zero otherwise.



Output	NF	f^{paf}	f^{waf}	f^{wth}	$f^{y,Tic}$	$f^{y,pic}$	$f^{y,pim}$	$f^{y,waf}$
NF	88.3%	33.2%	0.2%	0.5%	6.6%	10.3%	10.7%	0.2%
f^{paf}	6.7%	61.3%	1%	2.2%	0.1%	0%	11.2%	0.3%
f^{waf}	0.2%	0.5%	81.7%	9.2%	0%	0%	0.5%	1.9%
f^{wth}	0.5%	0%	12.3%	84.7%	0.3%	0%	0.1%	0.8%
$f^{y,Tic}$	0.8%	0.3%	0%	0%	91.2%	0%	0%	0%
$f^{y,pic}$	0%	0%	0%	0%	0%	83.5%	0%	0%
$f^{y,pim}$	1.2%	2.8%	0.5%	1.8%	0%	0%	73.5%	0%
$f^{y,waf}$	0.2%	0%	1.1%	0.2%	0%	0%	0%	94.3%
f^{x}	2.1%	1.8%	3.2%	1.4%	1.7%	6.2%	4%	2.5%

Fig. 10. Evaluation of a set of P_I-OSVM classifiers where the output of the ensemble classifier for each sample is the class with the highest probability (see Fig.. 9). Each column represents a fault scenario and each row represents the ranking of each fault class.

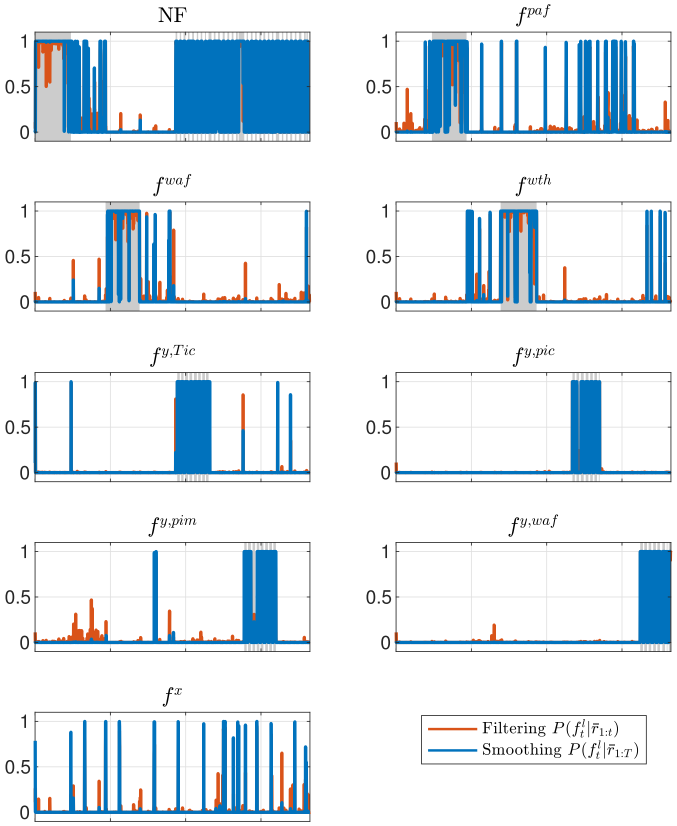


Fig. 11. Fault class probabilities using Bayesian filtering $p(f_t^l|\bar{r}_{1:t})$ and smoothing $p(f_t^l|\bar{r}_{1:T})$. The gray intervals represent when the corresponding fault class is present. Smoothing makes the probability of one fault class more dominating with respect to the other classes compared to filtering.

Each sample is associated with the fault class with the highest probability after applying Bayesian filtering and smoothing. Compared with the sample-by sample classification in Fig. 9, the filtered estimates significantly improve fault classification performance, especially between fault classes that are overlapping in Fig. 8. The smoothed probability often seems to dominate for one class at each sample time compared with using the Bayesian filter only. In the figure, only a few samples are classified to belong to the unknown fault case.

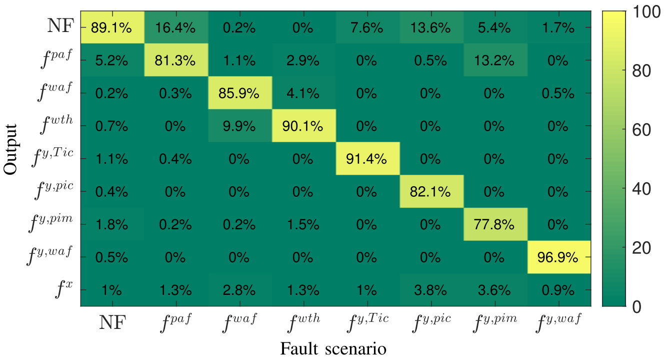


Fig. 12. Classification and ranking of a set of fault scenarios using a set of P_I-OSVM classifiers and Bayesian filtering (see Fig. 11). The rows represent the ranking of the different fault hypotheses for each fault scenario in the different columns.

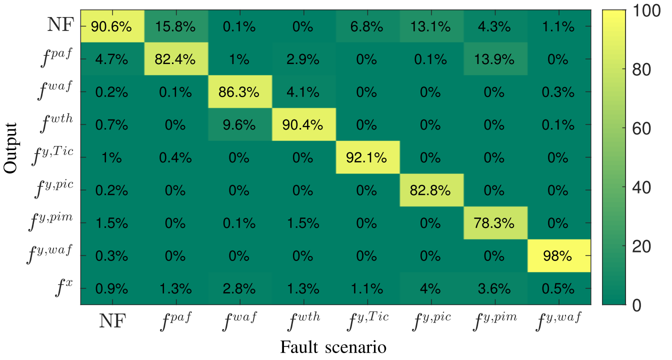


Fig. 13. Classification and ranking of a set of fault scenarios using a set of P_I-OSVM classifiers and Bayesian smoothing (see Fig. 11). There is a slight improvement compared with only using Bayesian filtering in Fig. 12.

Classification performance using Bayesian filtering and smoothing is shown in Figs 12 and 13, respectively. The output percentages show the ranking of each fault class in each scenario. The most significant improvement, with respect to the sample-to-sample classification in Fig. 8, is classification of fault f^{paf} where the ranking of the true fault increases from 61.3% to 81.3%. When comparing the results in Figs. 12 and 13, Bayesian smoothing has only a slight improvement in classification accuracy with respect to Bayesian filtering.

B. Classification of Unknown Faults

Unknown fault scenarios are simulated by training a set of P_I-OSVM models without including training data from the fault class that is considered unknown in the scenario. Seven unknown fault scenarios are evaluated where data from one fault class in Table I are excluded during each training phase and a set of P_I-OSVM models is trained based on the remaining known fault classes. Then, validation data from the unknown fault class is classified using the P_I-OSVM models and Bayesian smoothing to rank the different fault classes in each scenario. Ideally, in each fault scenario, the unknown fault class f^x should have the highest rank since the model of the true fault is not available.

The results of the unknown fault scenarios are shown in Fig. 14. Note that NF is not evaluated as an unknown fault scenario, and therefore, the first column is marked with X, but it is ranked in the other fault scenarios. The unknown fault class in each fault scenario is marked with "-" since there is no P_I-OSVM model to rank that fault. In all sensor fault scenarios, i.e., $f^{y,\mathrm{Tic}}$, $f^{y,\mathrm{pic}}$, $f^{y,\mathrm{pim}}$, and $f^{y,\mathrm{waf}}$, the unknown fault class has the highest rank. The two faults f^{waf} and f^{wth} are classified as each other and f^{paf} is classified as NF, which are expected since the CAP models are overlapping (see Fig. 8). The situation when NF gets a high rank, when a fault is present in the system, is likely when it is difficult to distinguish faults from model uncertainties and sensor noise.

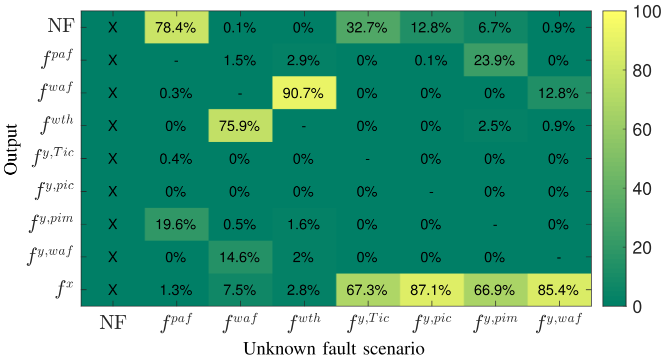


Fig. 14. Evaluation of classification of unknown fault scenarios. Training data from the selected unknown fault class are not included when training the set of P_I-OSVM models. All known fault classes are ranked using Bayesian smoothing and the evaluated fault class is marked with "-" in each scenario. Ideally, the unknown fault class f^x should have the highest rank in each column. However, some unknown faults are identified as another known fault class when the CAP models are overlapping.

One solution is to perform fault diagnosis in two steps, starting with a fault detection step followed by a fault classification step when a fault is detected. In situations where false alarms should be avoided, change detection algorithms, such as cumulative sum (CUSUM) [38], can be used to reduce the false alarm rate and improve the detection performance of small faults by allowing a longer time before detection. If a fault is detected with a low risk of false alarms, the following fault classification step can be performed by only considering faults without including the nominal class NF. For example, if a fault is detected in the unknown fault scenario with fault f^{paf} (see Fig. 14), and the NF fault class is removed during the Bayesian smoothing step, the ranking of f^{pim} increases from 19.6% to 82%, the unknown fault class f^x increases from 1.3% to 13%, and all the remaining fault classes remain below 2.4%. The higher ranking of f^{pim} is explained by the overlapping CAP models of the two fault classes (see Fig. 8).

The results show that the fault classification algorithm is able to handle unknown faults, but if residual data from a new type of fault are similar to a known fault, the previously known fault class will have a higher rank. When the root cause of a detected unknown fault has been correctly identified, for example, by a technician at the workshop, the fault models can be updated accordingly with the new training data, using, for example, incremental learning of the existing fault model or creating a new model for a newly identified fault class.

VI. Conclusion

Data-driven fault classification is complicated by unknown fault modes and limited training data. If multiple fault classes can explain residual data it is relevant to identify and rank the different faults instead of only selecting the most likely one, for example, when supporting a technician at a workshop. The solution proposed here is to apply the principles of open-set recognition, which considers the problem of data classification when there are unknown fault classes and limited training data. Modeling each fault class using a P_I-OSVM classifier is used to measure the probability of inclusion that can be combined with Bayesian filtering or smoothing to improve the classification performance of time-series data. An advantage of the proposed method is that it is straightforward to update and include new fault classes over time as new data are collected and labeled. Experiments using real engine data from different fault scenarios show that the proposed fault classification algorithm can identify unknown faults and that including temporal information significantly improves classification performance with respect to sample-to-sample classification.

REFERENCES

[1] M. Blanke, M. Kinnaert, J. Lunze, M. Staroswiecki, and J. Schröder, *Diagnosis and Fault-Tolerant Control*, vol. 2. Berlin, Germany: Springer-Verlag, 2006.

[2] I. Hwang, S. Kim, Y. Kim, and C. E. Seah, "A survey of fault detection, isolation, and reconfiguration methods," *IEEE Trans. Control Syst. Technol.*, vol. 18, no. 3, pp. 636–653, May 2010.

[3] D. H. Stamatis, *Failure Mode and Effect Analysis: FMEA From Theory to Execution*. Milwaukee, WI, USA: ASQ Quality Press, 2003.

[4] D. Jung, K. Y. Ng, E. Frisk, and M. Krysander, "Combining model-based diagnosis and data-driven anomaly classifiers for fault isolation," *Control Eng. Pract.*, vol. 80, pp. 146–156, Nov. 2018.

[5] 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.

[6] M. A. Atoui, A. Cohen, S. Verron, and A. Kobi, "A single Bayesian network classifier for monitoring with unknown classes," *Eng. Appl. Artif. Intell.*, vol. 85, pp. 681–690, Oct. 2019.

[7] A. Pernestål, M. Nyberg, and H. Warnquist, "Modeling and inference for troubleshooting with interventions applied to a heavy truck auxiliary braking system," *Eng. Appl. Artif. Intell.*, vol. 25, no. 4, pp. 705–719, Jun. 2012.

[8] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 7, pp. 1757–1772, Jul. 2013.

[9] E. Frisk and M. Krysander, "Residual selection for consistency based diagnosis using machine learning models," in *Proc. IFAC SafeProcess*, Warszaw, Poland, Aug. 2018, pp. 139–146.

[10] A. Haghani, T. Jeinsch, M. Roepke, S. X. Ding, and N. Weinhold, "Data-driven monitoring and validation of experiments on automotive engine test beds," *Control Eng. Pract.*, vol. 54, pp. 27–33, Sep. 2016.

[11] A. Theissler, "Detecting known and unknown faults in automotive systems using ensemble-based anomaly detection," *Knowl.-Based Syst.*, vol. 123, pp. 163–173, May 2017.

[12] H.-C. Yan, J.-H. Zhou, and C. K. Pang, "New types of faults detection and diagnosis using a mixed soft & hard clustering framework," in *Proc. IEEE 21st Int. Conf. Emerg. Technol. Factory Automat. (ETFA)*, Sep. 2016, pp. 1–6.

[13] C. Sankavaram *et al.*, "Model-based and data-driven prognosis of automotive and electronic systems," in *Proc. IEEE Int. Conf. Automat. Sci. Eng.*, Aug. 2009, pp. 96–101.

[14] C. Svärd, M. Nyberg, E. Frisk, and M. Krysander, "Automotive engine FDI by application of an automated model-based and data-driven design methodology," *Control Eng. Pract.*, vol. 21, no. 4, pp. 455–472, Apr. 2013.

[15] J. Luo, M. Namburu, K. R. Pattipati, L. Qiao, and S. Chigusa, "Integrated model-based and data-driven diagnosis of automotive antilock braking systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 40, no. 2, pp. 321–336, Mar. 2010.

[16] D. Jung and C. Sundström, "A combined data-driven and model-based residual selection algorithm for fault detection and isolation," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 2, pp. 616–630, Mar. 2019.

[17] K. Tidriri, N. Chatti, S. Verron, and T. Tiplica, "Bridging data-driven and model-based approaches for process fault diagnosis and health monitoring: A review of researches and future challenges," *Annu. Rev. Control*, vol. 42, pp. 63–81, Jan. 2016.

[18] H. Khorasgani and G. Biswas, "A methodology for monitoring smart buildings with incomplete models," *Appl. Soft Comput.*, vol. 71, pp. 396–406, Oct. 2018.

[19] W. Zhang, G. Biswas, Q. Zhao, H. Zhao, and W. Feng, "Knowledge distilling based model compression and feature learning in fault diagnosis," *Appl. Soft Comput.*, vol. 88, Mar. 2020, Art. no. 105958.

[20] K. Tidriri, T. Tiplica, N. Chatti, and S. Verron, "A generic framework for decision fusion in fault detection and diagnosis," *Eng. Appl. Artif. Intell.*, vol. 71, pp. 73–86, May 2018.

[21] I. Matei, M. Zhenirovskyy, J. de Kleer, and A. Feldman, "Classification-based diagnosis using synthetic data from uncertain models," in *Proc. PHM Soc. Conf.*, vol. 10, no. 1, 2018, pp. 1–8.

[22] Y. Yan, P. B. Luh, and K. R. Pattipati, "Fault diagnosis of components and sensors in HVAC air handling systems with new types of faults," *IEEE Access*, vol. 6, pp. 21682–21696, 2018.

[23] W. J. Scheirer, L. P. Jain, and T. E. Boult, "Probability models for open set recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2317–2324, Nov. 2014.

[24] E. M. Rudd, L. P. Jain, W. J. Scheirer, and T. E. Boult, "The extreme value machine," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 762–768, Mar. 2018.

[25] D. Jung, "Engine fault diagnosis combining model-based residuals and data-driven classifiers," in *Proc. IFAC Int. Symp. Adv. Automot. Control*, 2019, pp. 285–290.

[26] L. Jain, W. Scheirer, and T. Boult, "Multi-class open set recognition using probability of inclusion," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 393–409.

[27] G. Kitagawa, "Non-Gaussian state—space modeling of nonstationary time series," *J. Amer. Stat. Assoc.*, vol. 82, no. 400, pp. 1032–1041, 1987.

[28] R. Domingues, M. Filippone, P. Michiardi, and J. Zouaoui, "A comparative evaluation of outlier detection algorithms: Experiments and analyses," *Pattern Recognit.*, vol. 74, pp. 406–421, Feb. 2018.

[29] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, and J. C. Platt, "Support vector method for novelty detection," in *Proc. NIPS*, vol. 12, 1999, pp. 582–588.

[30] D. M. J. Tax and R. P. W. Duin, "Support vector data description," *Mach. Learn.*, vol. 54, no. 1, pp. 45–66, Jan. 2004.

[31] T. Hastie, R. Tibshirani, J. Friedman, and J. Franklin, "The elements of statistical learning: Data mining, inference and prediction," *Math. Intell.*, vol. 27, no. 2, pp. 83–85, 2005.

[32] D. M. J. Tax, "Ddtools, the data description toolbox for MATLAB, version 2.1.2," Dept. EWI, Delft Univ. Technol., Delft, The Netherlands, Tech. Rep., Jun. 2015. [Online]. Available: https://www.tudelft.nl/ewi/over-de-faculteit/afdelingen/intelligent-systems/pattern-recognition-bioinformatics/pattern-recognition-laboratory/data-and-software/dd-tools/

[33] J. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," *Adv. Large Margin Classifiers*, vol. 10, no. 3, pp. 61–74, 1999.

[34] L. Eriksson, S. Frei, C. Onder, and L. Guzzella, "Control and optimization of turbo charged spark ignited engines," in *Proc. IFAC World Congr.*, 2002, pp. 1–6.

[35] L. Eriksson, "Modeling and control of turbocharged SI and DI engines," *Oil Gas Sci. Technol.-Revue l'IFP*, vol. 62, no. 4, pp. 523–538, 2007.

[36] E. Frisk, M. Krysander, and D. Jung, "A toolbox for analysis and design of model based diagnosis systems for large scale models," in *Proc. IFAC World Congr.*, Toulouse, France, 2017, pp. 3287–3293.

[37] *MATLAB 2018b Statistics and Machine Learning Toolbox*, MathWorks, Natick, MA, USA, 2018.

[38] E. Page, "Continuous inspection schemes," *Biometrika*, vol. 41, nos. 1–2, pp. 100–115, 1954.