IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 9 June 2022, accepted 29 June 2022, date of publication 4 July 2022, date of current version 13 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3188102

RESEARCH ARTICLE

A Hybrid Machine-Learning Ensemble for Anomaly Detection in Real-Time Industry 4.0 Systems

DAVID VELÁSQUEZ[1,2,3], ENRIQUE PÉREZ[2], XABIER OREGUI[2], ARKAITZ ARTETXE[2], JORGE MANTECA[4], JORDI ESCAYOLA MANSILLA[5], MAURICIO TORO[1], MIKEL MAIZA[2], AND BASILIO SIERRA[3]

[1]RID on Information Technologies and Communications Research Group, Universidad EAFIT, Medellín 050022, Colombia
[2]Department of Data Intelligence for Energy and Industrial Processes, Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), 20014 Donostia-San Sebastián, Spain
[3]Department of Computer Science and Artificial Intelligence, University of Basque Country (UPV/EHU), 20018 Donostia-San Sebastián, Spain
[4]Technical Department Direction, Mapner, 20115 Astigarraga, Spain
[5]Department of Statistics and Operational Research, Universitat Oberta de Catalunya, Rambla del Poblenou, 08018 Barcelona, Spain

Corresponding author: David Velásquez (dvelas25@eafit.edu.co)

This work was supported in part by Vicomtech Foundation and in part by Universidad EAFIT.

ABSTRACT Detecting faults and anomalies in real-time industrial systems is a challenge due to the difficulty of sufficiently covering an industrial system's complexity. Today, Industry 4.0 makes it possible to tackle these problems through emerging technologies such as the Internet of Things and Machine Learning. This paper proposes a hybrid machine-learning ensemble real-time anomaly-detection pipeline that combines three Machine Learning models –Local Outlier Factor, One-Class Support Vector Machine, and Autoencoder–, through a weighted average to improve anomaly detection. The ensemble model was tested with three air-blowing machines obtaining a F_1-score value of 0.904, 0.890, and 0.887, respectively. The results of the ensemble model showed improved performance metrics concerning the individual metrics. A novelty of this model is that it consists of two stages inspired by a standard industrial system: i) a manufacturing stage and ii) an operation stage.

INDEX TERMS Anomaly detection, industry 4.0, machine learning, predictive maintenance, real-time.

I. INTRODUCTION

Thanks to the fourth industrial revolution (4IR), traditional industrial processes face new challenges: improving current or establishing new processes that efficiently use novel technologies and fully exploit their potential. 4IR or Industry 4.0 is viewed as a disruptive innovation in a highly competitive market that positively impacts several industrial sectors by incorporating new enabling technologies: 3D printing, the Internet of Things (IoT), Cyber-Physical Systems (CPS), Artificial Intelligence (AI), Big Data, Robotics, Nanotechnology, and Quantum Computing are examples of these technologies [1]. In industrial machines, high volumes of data are generated and acquired by data acquisition systems such as a Supervisory Control and Data Acquisition (SCADA) or an embedded system. AI algorithms can then process this data to generate new knowledge of the process and identify new machine conditions, which represents one of the advancements provided by Industry 4.0. Predictive maintenance is an industrial process that is the subject of the work presented in this article and highly benefits from the Industry 4.0 technologies mentioned above [2].

Nowadays, most industrial companies face problems arising from maintaining their systems. However, multiple techniques –involving predictive or *condition-based maintenance (CBM)*– allow predicting critical situations to reduce these problems. According to An *et al.* [3], in terms of diagnosis, predictive maintenance is divided into two categories:

The associate editor coordinating the review of this manuscript and approving it for publication was Mehul S. Raval.

i) Models that take into account physical principles and ii) models based on historical observations. One of the techniques used in the second group consists of the early detection of abnormal behavior in industrial equipment. This early detection can avoid possible breakdowns of equipment and reduce associated maintenance costs.

Anomaly detection is being researched in several application fields. Some of the associated research fields are disease detection, intrusion detection, fraud prediction, and fault detection in industrial equipment [4]. It is possible to identify anomalous states that do not match the normality data, which usually corresponds to the predominant states through anomaly detection.

The detection of anomalous states presents a challenging task. The detection becomes more complicated than usual if it is to be done in real-time due to the restrictive features of the streaming data. Unlike batch learning, where all the historical data are available, and no new information is added to the models already built, stream learning has five restrictions that must be taken into account [5]. i) Streaming data samples arrive online and can be read at most one time, which is a strong restriction for processing them since the system has to decide whether the current data sample is discarded or archived. ii) Past data samples can only be accessed if stored in memory. Otherwise, a forgetting mechanism in charge of discarding past samples is applied. iii) Since not all data samples can be stored, a decision made on past samples cannot be undone. iv) The data processing time of each data sample should be short and constant. v) The data processing algorithm must produce a model equivalent to what a batch algorithm would produce.

The former five restrictions are why most anomaly detection algorithms –for batch processing– do not apply to stream processing. Nonetheless, there are hybrid approaches that use batch-learning algorithms to build an initial model as the first step and then apply streaming anomaly-detection algorithms as the second step.

The contribution of this work is the evaluation and comparison of different methods to detect anomalies that, due to their performance-control metrics, establish the weight (or incidence) of each method in the final combined model, thus responding better and efficiently to the challenge of real-time anomaly detection. Specifically, the present work combines the predicted output of three Machine Learning (ML) models: Local Outlier Factor (LOF), One-Class Support Vector Machine (OCSVM), and Autoencoder employing a weighted average –using as weight the F_1-score value of each model. The goal of the combined model is the detection of anomalies in industrial systems in real-time. The proposed hybrid model was implemented using a data set from a real industrial system of air-blowing machines. Thus, it can be said that the proposed hybrid anomaly detection model applies to Industry 4.0 systems as well as other industrial frameworks where real-time data acquisition systems are available.

The following sections of the article are divided into four sections. The state-of-the-art section shows existing approaches and research for anomaly detection in real-time. Next, the third section shows a detailed explanation of the proposed hybrid anomaly detection. Finally, the results section describes the scores obtained by applying the hybrid anomaly detection methodology to a testing data set. A Conclusions section ends this paper, showing some concluding remarks and a future work proposal.

II. STATE OF THE ART

According to [6], [7], an anomaly can be defined as a point in time where the system's behavior is unusual and significantly different from previous, normal behavior. An anomaly may imply an adverse change in the system, for instance, a fluctuation in a jet engine's turbine rotation frequency, which possibly means an imminent failure. An anomaly may also mean positive behavior; for instance, many web clicks on a new product page imply higher demand. In both cases, anomalies in data provide an insight into abnormal behavior that can be translated into potentially useful information.

The challenge of detecting anomalies –in an industrial environment– can be twofold. Firstly, to propose a method to understand different data obtained from various sensors, often with excessive noise. Secondly, to obtain an overview of normal behavior to characterize such behavior from historical data. Therefore, to correctly detect anomalies in a data set, one must first characterize and define normal data behavior [8]. In addition, normal behavior can be characterized by the following three stages. (i) Consider data describing normal behavior through historical data (without considering anomalies) segmented into different classes according to the context in which they were recorded. (ii) Extract the most frequent behaviors, thus characterizing each class. (iii) Detect anomalies in newly recorded data based on previous knowledge.

In general, anomalies are classified into three types: specific, contextual, and collective [9]–[11]. It is considered a point anomaly when this single data point is recognized as anomalous concerning the rest of the data. According to [10], these anomalies must be identified before processing or analyzing the data.

- *Contextual anomalies* are those where the data are considered anomalous in a specific context (e.g., the same sample data are "normal" in a given scenario but anomalous in another context). These types of anomalies are more common in time-series data flows [10].
- *Collective anomalies* are those that occur when a collection of related data are considered anomalous to the total data. Collective anomalies can also be spatial if they are outside a typical range or temporal, where the value is not outside the typical range. However, the sequence in which it occurs is unusual.

Anomaly detection methods can be distinguished as supervised, semi-supervised and unsupervised. Using one method or another usually depends on the existence or not of descriptive labels of the anomaly. The labels can be categorical, e.g., we can have a case of binary or all/nothing labels such as "anomalous behavior" (1) and "non-anomalous / normal behaviour (0)", or numerical, e.g., a value of "anomaly score" ranging from 0 ("non-anomalous / normal") to 1 ("totally anomalous"). While anomaly detection could be posed as a supervised learning problem, this is –generally– not the case, as there is often no or little data labeled with the anomalous behavior [12].

Once the data is available, normally, a series of transformations of the data needs to be performed before starting the anomaly detection process [13].

- *Aggregation methods:* A set of consecutive values from a time-series data is replaced by a corresponding representative value. It provides benefits such as reducing dimensionality, although it can make detecting anomalies in subsequent steps difficult.
- *Discretisation methods:* Time-series data are converted into a discrete sequence of finite alphabets. Techniques such as symbolic sequence and editing distance can be applied to detect anomalies.
- *Digital Signal Processing (DSP) techniques* (such as Fourier transform, Gabor, and Wavelets filters): Time-series data are transformed into a lower-dimensional representation of the input data where anomaly detection can take place.

A common type of problem detected, which may be present in the data, is noise and outliers. Noise among normal data may cause the model not to obtain the desired optimal predictions. Outliers are data points that may be caused by noise or may have an irregular pattern of behavior. Therefore, this unusual behavior must first be identified and decided whether it should be considered an anomaly or an outlier.

Usually, data are created by one or more generation processes, representing system's activities. When the generation process behaves unusually, it creates anomalies. Therefore, an anomaly often contains valuable information about the abnormal characteristics of the systems and elements that impact the generation process [11].

A. CLASSIFICATION OF TECHNIQUES FOR ANOMALY DETECTION

There are currently six techniques to detect anomalies. These techniques are i) Statistics, ii) Classification, iii) Clustering, iv) Similarity-based, v) Soft Computing, and vi) Knowledge and Combined Techniques based, as explained in [13]. In Table 1, these techniques –and some examples of the algorithms– used can be seen in detail. The most relevant ones for this work will be detailed next.

1) STATISTICS BASED ANOMALY DETECTION TECHNIQUES

Statistical techniques adjust a predefined distribution to a given data and apply statistical inference to determine whether an instance belongs to that model. Instances with a low probability are reported as anomalies [14].

TABLE 1. Classification of the different techniques for anomaly detection [13].

Technique	Sub Techniques	Examples
Statistical	Parametric, Non-parametric	Box-plot, Grubbs test, Chi-square, PCA, Kernel methods
Classification	Multi-Class, One Class	Neural Networks, Bayesian Networks, SVM, Decision Trees
Clustering	Parametric, Non-parametric	DBSCAN, Rock, SNN, K-means, EM, LOF variants
Similarity based	Continuous and categorical data	k-NN variants, Relative Density
Soft Computing	GA, NN, Fuzzy and Rough Sets, Ant Colony	GANIDS, NN, DNN, CNN
Knowledge based	Rules and Expert Systems, Ontology and Logic-based techniques	Decision Trees
Combination Learners	Ensemble based, Fusion based, Hybrid	Bagging and Boosting

The two typologies covered by this technique are parametric and non-parametric. The first assumes an underlying data distribution. Although somewhat less efficient in finding anomalies, the second is preferred because, a priori, it does not define any model structure as this is determined from the data.

The most common parametric techniques are divided into those based on Gaussian models and those based on regression models. If a non-parametric approach is to be followed, such a classification can be made based on histograms or kernels.

Statistical techniques work well for simple structured data with small dimensions and volume. In such cases, several methods can be used [13], such as Box-plots, Blum Floyd Pratt Rivest Tarjan (BFPRT) algorithm, and similar central-value estimations on data streams; Medcouple and Grubbs test (for univariate data); Comparison of distributions (QQ charts, Kolmogorov-Smirnov test, Kruskal-Wallis test, and Wilcoxon signed range tests); Auto-regressive techniques (Auto-regressive Integrated Moving Average - ARIMA, Auto-regressive Moving Average - ARMA); ML-based methods; Bayesian networks. Principal Components Analysis (PCA) / Independent Component Analysis (ICA) (e.g., sequence micro-batch analysis).

2) CLASSIFICATION BASED ANOMALY DETECTION TECHNIQUES

Classification-based anomaly detection techniques perform two main stages called training and testing. *In the training phase*, the system learns from the available samples and generates a classifier. *In the testing phase*, samples that the classifier has not seen are tested to measure the model's performance. According to the labels available for training, classifiers can be grouped into two categories: i) one-class and ii) multi-class. Examples of single and multi-class classifiers are neural networks, Bayesian networks, Support Vector Machines (SVM), and decision trees. These, together with fuzzy logic, are also methods that present a good performance in the presence of strong noise [15]–[18].

Classification-based techniques have the advantage of being able to distinguish between observations that belong to different anomalies (instead of an overall class called "anomaly"), and their testing phase is quick, as the test instance is compared to the predefined model [19]. Although, classification techniques are based on the availability of assigning labels to various normal and abnormal classes, which is a difficult task. Also, these techniques assign labels to test data, which can be a disadvantage when an anomaly score is desired.

Classification-based techniques can also be categorized according to the type of anomaly. Radial-Base Functions (RBF), SVM, and derivates are commonly used for individual anomalies. RBFs are very accurate and fast, particularly for the supervised classification of individual anomalies. For multiple anomalies, Deep Neural Networks (DNN), induction rules, and decision trees are used. DNNs can provide exceptional recognition rates in static scenarios but can give data problems that vary over time.

3) CLUSTERING-BASED ANOMALY DETECTION TECHNIQUES

Clustering techniques are generally divided into two stages: first, the data are grouped with clustering algorithms, and then the degree of deviation is analyzed according to the results obtained by the clustering [4]. There are some prior considerations about the data instances in these unsupervised techniques. On the one hand, normal-data samples belong to global clusters. On the other hand, anomalies do not belong to any defined cluster. In addition, normal data samples are near the centroids of the closest cluster, while anomalous data are further away. Finally, normal-data samples belong to large, dense groups, but anomalies belong to local, small, disparate groups.

Cluster-based methods are applied in both supervised and unsupervised learning. Most techniques work well for complex, large-sized, and voluminous data and –optimally– if the anomalies do not form significant clusters in a short time series. Examples of this type of algorithm are k-Means, Shared Nearest Neighbour (SNN), Density-Based Spatial Clustering of Applications with Noise (DBScan), Self-Organizing Map (SOM), or Clustering-based Dynamic indexing Tree (CD-Tree) [4].

4) SIMILARITY BASED ANOMALY DETECTION TECHNIQUES

These techniques are the most widely used to detect anomalies. One of the techniques, based on similarity, is known as k Nearest Neighbours (k-NN). k-NN is a non-parametric method that requires a distance metric to measure the similarity between data observations. Although Euclidean distance is the most commonly used metric for data with continuous attributes, it is not usually employed on a practical level. The above is because the Euclidean distance does not work well in high-dimensional sets, and measurements such as Mahalanobis, Hamming, or Chebyshev distances are used instead. The k-NN algorithm is based on the data score given by the distance to most of the data around it. So, new data are classified according to this score. Although, there are some considerations to be taken into account in this type of technique [13]: i) A shortage of data can be seen as an anomaly in unsupervised techniques. ii) The performance is a function of the distance method chosen; therefore, the criteria must be clear when choosing a metric. iii) It is valid only in cases of low-dimensional data. Defining a measure of the distance between instances can be complicated when the data dimension is increased.

Another essential similarity-based anomaly detection technique is based on relative density rather than distance. This technique estimates the neighborhoods' density so that a data item in a low-density neighborhood will be anomalous while one in a high-density neighborhood will be considered normal. An existing method for the above is the Local-Outlier Factor (LOF), which introduces the concept of local outliers and is based on scoring a data sample according to the average ratio of the neighborhood's density to the instance's density [20].

B. RELATED WORKS

Many studies on anomaly detection in static data sets in the literature exist. Examples of supervised approaches are SVM and Decision Tree [12], or cluster-based methods such as the Distributed Matching-based Grouping Algorithm (DMGA) [21]. Other examples use self-adaptive and dynamic clustering to learn weights for anomaly detection [22] or statistical methods such as auto-regressive techniques (e.g., ARIMA models [23]).

The problem with these methods is that they are not designed to process streaming data as they need to have the data set previously stored in the main memory. Therefore, these traditional techniques have been adapted first and then applied to streaming-data environments in many cases.

In this sense, Tan *et al.* [24] propose a fast-anomaly detection of a class that uses only normal data and works well when anomalous data are rare. To do this, they use the Half-Space Trees (HS-Trees) algorithm. The HS-Trees algorithm presents a set of random HS trees. Each HS tree consists of a set of nodes, where each node captures the number of data elements (called mass) within a subspace of the data stream. The mass is used to profile the degree of an anomaly as it is quick and straightforward to calculate compared to other methods based on distance or density. The tree structure is constructed without any data, making it very efficient as it does not require restructuring the model once it is running on streaming data. HS-Trees only need normal data for training.

Another technique that is worth mentioning is the isolation-Forest Algorithm for Streaming Data (iForestASD) [25], based on the Isolation-Forest algorithm [26]. This method handles streaming data using sliding windows. In this case, the authors start from the "concept drift", which is a common occurrence handling the streaming of data in dynamic and non-stationary environments producing a change in the distribution of the data [27]. The "concept drift" is a problem that occurs when the statistical properties of the target variable change over time and the anomaly detection model is no longer compatible with the data the model handles, resulting in less accurate predictions. Therefore, to maintain the anomaly detection effectively, the model needs to be retrained and updated based on the new data the model receives [27].

Another research work on anomaly detection is proposed by [28], which is based on an HT (Hoeffding tree). It is an inductive-incremental decision-tree algorithm used for anomaly detection. A handicap of this algorithm is that it needs class labels to be available for training.

Another work to be highlighted would be that carried out by a group of Yahoo researchers [29]. Their system –called Extensible Generic Anomaly Detection System (EGADS)– allows precise, flexible, scalable, and extensible detection of anomalies, taking into account time series. The system makes it possible to separate forecasting, anomaly detection, and alerts into three separate components.

Finally, another interesting work is that contributed by [30] in which, through the integration of various technologies, the development of a disease in the leaf of a Colombian-coffee variety is evaluated and diagnosed. The project contribution relied on a model ensemble comprising four sub-models that received the data according to their nature. Once the prediction of each sub-model was made, its results were combined, calculating the weighted average. The weight of each sub-model was a value associated with its F_1-score value in the final model.

Most of the approaches to detect anomalies existing in the literature are based on models that first build a profile of what is "normal" and then point out those instances that do not fit that normal profile as anomalies (statistical methods, classification-based methods, or cluster-based methods use this approach).

A contribution of this work is to build an ensemble model that uses different algorithms that, by combining their results, will generate a new model to detect anomalies. Ensemble learning, either for classification or regression, refers to methods that generate multiple models that are combined to make a prediction [31]. Ensembles have been –extensively– used in the last decades as they are considered to provide greater accuracy and increased robustness [32]. Additionally, multiple ensemble approaches have been proposed, and several studies have reported that model diversity enhances the ensemble model's performance as different learners generalize in different ways [33].

III. PROPOSED METHODOLOGY

The proposed ML hybrid pipeline for real-time anomaly detection, as seen in Fig. 1, consists of two stages: i) the Manufacturing stage and ii) the Operation stage.

The manufacturing stage or pipeline of the Hybrid Anomaly Detection model construction process takes its name from the manufacturing process of an industrial machine. At this stage, an ML model is trained on machines' quality control process data to validate whether the machine meets its design standards or not [34]. Thus, the objective of completing this manufacturing stage model construction task is double: (i) to use the trained model for detecting machine design/manufacturing anomalies; (ii) to later deploy it in the operation stage of the machine when it is integrated into an industrial production process, for performing a machine operation anomaly detection task. This model construction manufacturing stage is equivalent to the design phase of a classical ML workflow. The metric chosen for measuring models' performance is the F_1-score of label L. The data set available is a slightly imbalanced (see Table 2 for class sizes percentage), where more machine's "normal data" than "anomalous data" exists, for which the F_1-score metric is considered appropriate. The F_1-score is a value in the [0, 1] range, and it's calculated as the harmonic mean of the estimator's precision and recall with respect to L (see Equation (1))

$$F_1\text{−}score_{\mathbf{L}} = \frac{2 \times \text{precision}_{\mathbf{L}} \times \text{recall}_{\mathbf{L}}}{\text{precision}_{\mathbf{L}} + \text{recall}_{\mathbf{L}}} \tag{1}$$

Finally, models' F_1-score ($F1_i$) performance ratio with respect to the sum of all F_1-scores ($\sum_j F1_j$) (see Equation 2) is calculated and used as the weight (w_i) for the weighted average of the prediction done by each model multiplied by the computed weights. This weighted average assembles the Hybrid Anomaly Detection model at the manufacturing stage.

$$w_i = \frac{F_1 - score_i}{\sum_j F_1 - score_j} \tag{2}$$

The operation stage or pipeline refers to the phase when the machine is already running in production; in terms of a classical ML pipeline, it represents the deployment phase. Thus, this pipeline requires the machine to be able to measure the same variables taken at the manufacturing stage through industrial sensors. Once these sensors' data are captured in real-time, they are used as inputs for the Hybrid Anomaly Detector, already trained during the manufacturing stage. This detector will diagnose based on the data received to generate an alarm for the operator in case of an anomaly. This detector can also be tuned in operation through a supervised action of the operator. If this action is triggered, the data are captured during a time window and labeled as "normal" data. The models are retrained within the hybrid anomaly detector when the data capture is complete. Once the calibration is finished, the system will be able to continue detecting anomalies in real-time.

A. MANUFACTURING-STAGE PIPELINE

As previously mentioned, this stage is executed when the machine is in the factory. The proposed pipeline requires that the manufactured machine goes through a quality control process [34], where sensors can capture information about the

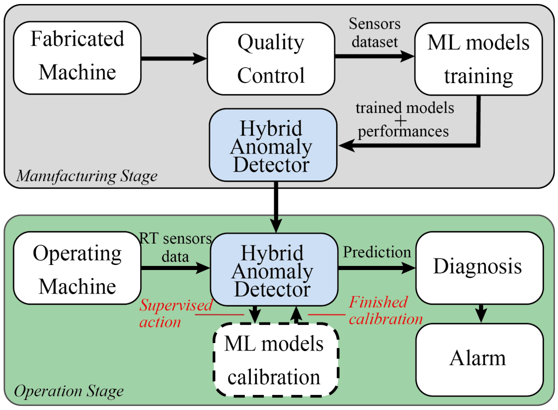


FIGURE 1. Higher-level representation of the proposed Hybrid-ML pipeline for Anomaly Detection in real-time.

manufactured machine's operation during a period of time. The data captured by the sensors during the quality control process will be called *sensor data set*.

Once sensors' data are stored, the data are pre-processed for data cleaning purposes, i.e., those features that the system cannot capture with sensors when the machine is in operation are removed.

The pre-processed data are then normalized so that all features are on the same scale and comparable in later stages of the pipeline. A feature selection is then carried out to extract those variables relevant to the study; this step includes as a first filter the expert in the domain knowledge, which can give an initial selection of what variables should be maintained or discarded. Then an automatic algorithm [35] to remove redundant features is applied. Following the above, a dimensionality reduction is performed using a Principal Components Analysis (PCA) to extract the data's most representative characteristics.

The next stage is to apply a clustering algorithm, the K-means algorithm, with $k = 2$, which allows a distinction between a group of data samples belonging to the transient state and another group of data belonging to the steady state. To correctly label the result of the groups generated by the clustering algorithm, the cluster assigned value is first identified to the sample with the lowest timestamp of the data set. This value will correspond to the *Transient Data Group* and, therefore, all the samples containing this same cluster value will correspond to this same state. The rest of the values will be labeled as *Steady-State Data Group*.

It is also proposed for the steady-state data group to apply an outlier detection algorithm. In this case, it is proposed to use a density-based algorithm called DBSCAN, which is useful to detect outliers in applications with noise, commonly found in industrial sensor data [36].

Once the data group belonging to the transient state, stable state, and outliers (in the stable state) have been identified, a data set with new labels is generated. Furthermore, a depuration stage is carried out to obtain the final label for the data set. The transient state and outliers are labeled with a value of -1, and the normal stable data is labeled with a value of 1. The previous data set is then divided at random and stratified into three sets: training, validation, and test. The training set corresponds to 60% of all the data, where only the normal data are used to build each ML model with cross-validation, which allows for testing its intermediate performance and tuning model hyper-parameters.

For this pipeline, the following three ML algorithms were used, selected as a result of the authors' research work on state of the art relating one-class anomaly detection for real-time systems, as they present an optimum balance of computation cost, implementation complexity, and performance [6]–[8], [12], [19]: i) LOF, which finds anomalous data points using the local deviation of a given data point to its neighbors [20]; ii) One-Class SVM (OCSVM), which finds a frontier that encloses the vast majority of data (normal data) and new upcoming data that lay outside the frontier are considered abnormal [37], [38]; and iii) Autoencoder, which reduces the input data's dimensionality by encoding the information to a smaller space. From this compressed space, it is decoded to the same dimensions as the original input. The reconstruction error in this process determines a possible anomaly [39].

Normal data are used for the training because the proposed pipeline is designed to identify anomalies based on a single class for novelty detection, and individual ML models use unsupervised algorithms.

The validation set, which corresponds to 20% of the data set, is used to obtain the definitive performance (in this case, the F_1-score value) of each trained model. The weights for the predictions of each model are then determined as the ratio of each F_1-score value (obtained using the validation set). The weights are stored to be later used for the rounded weighted average of the Hybrid Anomaly Detector component. The test set corresponds to the final 20% of the data set and is reserved for measuring the performance of the hybrid anomaly detector. The manufacturing stage pipeline is shown in Fig. 2.

B. OPERATION-STAGE PIPELINE

This stage is executed when the machine is in operation. The operating machine generates real-time data from previously installed sensors during this process, corresponding to the same sensors used in the manufacturing stage. Each execution cycle is pre-processed and delivered to the previously obtained hybrid model, giving a diagnosis if the machine is in normal condition or if any anomalies should be reported through an alarm.

The operation stage also allows for calibrating the Hybrid Anomaly Detection models required in industrial systems that degrade over time and can be planned (e.g., every time maintenance is carried out). The operator must verify that the machine is in a stable state and under optimal conditions of normality and activate the ML models' calibration routine to carry out this process. Once this process is activated, the system will collect data during a period of time, which will depend on each system's dynamics. Each data will be stored with the normality label in the data set. This data set with normal data is then used to retrain each ML algorithm

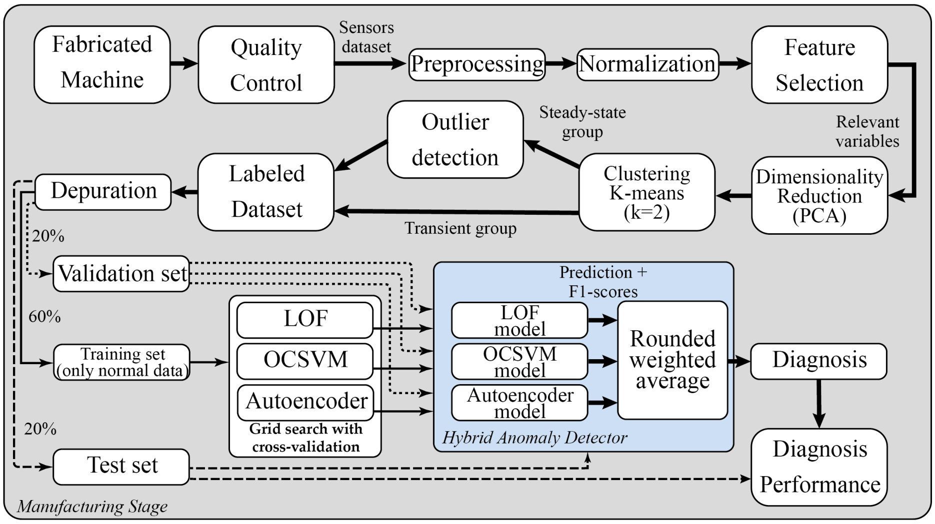


FIGURE 2. ML manufacturing stage pipeline.

TABLE 2. Air-Blowing machines' data set characteristics.

Model version		
A	Date control	1 June 2020
	Start-End time	08:30 - 09:36
	Total Samples	1990
	Normal Samples	67.789%
	Anomaly Samples	32.211%
	Sample period	2 sec.
B	Date control	15 June 2020
	Start-End time	08:13 - 09:20
	Total Samples	2009
	Normal Samples	55.351%
	Anomaly Samples	44.649%
	Sample period	2 sec.
C	Date control	14 July 2020
	Start-End time	09:40 - 10:51
	Total Samples	2132
	Normal Samples	70.779%
	Anomaly Samples	29.221%
	Sample period	2 sec.

TABLE 3. Variables pre-processing at manufacturing stage.

Variable	Available at Manufacturing	Available at Operation
Flow Rate	✓	×
Nozzle Temperature	✓	×
Suction Pressure	✓	✓
Discharge Pressure	✓	✓
Flow Temperature	✓	✓
Machine Vibrations	✓	✓
RPM	✓	✓
Active Power	✓	✓
Cos Phi	✓	✓
Motor Current	✓	✓
Motor Voltage	✓	✓
Ambient Humidity	✓	✓
Ambient Temperature	✓	✓
Atmospheric Pressure	✓	×
Water Temperature	✓	×

with cross-validation. Finally, the newly trained models are updated in the Hybrid Anomaly Detector. It should be noted that only the weights (obtained through the F_1-scores) that were acquired in the manufacturing process are used because, in the operation process, usually, there are no anomalous data to measure this performance. The operation stage pipeline can be seen in Fig. 3.

C. EXPERIMENTAL SETUP

The proposed ML Hybrid real-time anomaly detection pipeline was tested for three different industrial air-blowing machines from the local industry, with a data set generated by the quality-control process, and these machines are currently operational.

The period for collecting machines' data is between 7 January 2020 and 2 October 2020. The data are recorded and stored at 2-second intervals. The final data set comprises 16 columns (15 variables and timestamps) with 1990 observations for Machine A, 2009 observations for Machine B, and 2132 observations for Machine C. The above-mentioned data set characteristics are shown in the table 2.

The sensors' data set was composed of the variables measured by sensors installed in each machine in the Quality-Control stage. The measured variables were Flow Rate, Power, Water Temperature, Nozzle Temperature, Input Pressure, Output Pressure, Flow Temperature, Machine Vibrations, RPM, Active Power, Cos Phi, Motor Current, Motor Voltage, Ambient Humidity, Ambient Temperature, Atmospheric Pressure.

The pre-processing step selects the shared variables for the manufacturing and operation stages. The variables' pre-processing can be seen in Table 3, with a total of 11 variables selected (those with ticks in both manufacturing and operation). Additionally, samples with invalid or missing values were checked and removed from the data set in the pre-processing stage.

Afterward, the pre-processed data set was normalized to scale variables' values, as it is recommended for data preparation in ML since some of the variables have different ranges [40]. The normalization used for this experiment was the Min-Max scaling, which scaled the data to values between 0 and 1.

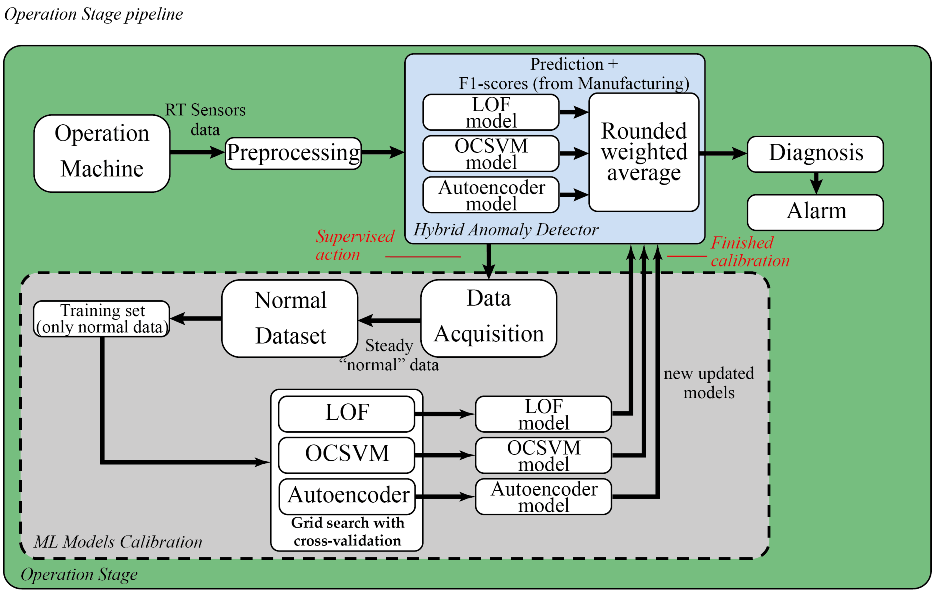


FIGURE 3. ML operation stage pipeline.

TABLE 4. Outlier detection using DBSCAN.

Parameters	Machine A	Machine B	Machine C
min_samples	7	8	8
eps	0.026	0.029	0.029
Silhouette Coef.	0.272	0.163	0.175
# of Outliers	154	200	141

The "Standard Scaler" (Z-score Normalization) was not used as the normalization method due to two main reasons: i) In the presence of outliers, the "Standard Scaler" does not guarantee balanced scales of characteristics due to the influence of outliers on the calculation of the empirical mean and standard deviation, and ii) the "Standard Scaler" assumes a normally distributed data set, which is not the case of our data set. In cases where the distribution is not Gaussian or the standard deviation is small, the "Min-Max" scaling works better [41]. Besides, "Min-Max" preserves the original distribution, does not significantly change the information embedded in the original data, and does not reduce the importance of outliers.

Following Data Normalization, a Feature-Selection step was carried out, where all the data features were validated with the expert in the domain of the machines tested. The expert determined that the "environmental" variables (Ambient Humidity, Temperature, and Atmospheric Pressure) should not be taken into account since they can present a change not necessarily related to the machine's behavior and generate information that can disturb the final prediction of the system. The variable Cos-phi was removed because it had zero variance. Finally, the motor voltage could be explained through the motor current, and it was removed, as it was considered redundant. Finally, seven variables remained, and none of them had zero variance, so no additional variable selection step was required.

A dimensionality reduction was performed using a two-component PCA with the selected features, which explained the variance by 90% for each machine. A clustering was then performed using k-Means to separate the data between the Transient State and the Steady-state with $k = 2$ groups. Furthermore, the Silhouette coefficient was used to measure the clustering's quality, presenting a value of 0.6547 for machine A, 0.5895 for machine B presented, and 0.6744 for machine C.

TABLE 5. Labelled data sets final samples observations.

Labels	Machine A	Machine B	Machine C
Normal	1349	1112	1509
Anomaly	641	897	623

Once the Transient and Steady-state data groups were separated, outliers were detected using DBSCAN in the Steady-state part. For this algorithm, two parameters called minimum samples (min_samples) and epsilon (eps) are required, which are assigned to a list of initial values. Then the best values are found automatically to maximize the Silhouette coefficient. The list of initial values for the three machines are displayed in equations 3 and 4.

$$\text{initial\_min\_samples} = [2, 3, 4, 5, 6, 7, 8] \tag{3}$$

$$\text{initial\_eps} = [0.010, 0.011, 0.012, \ldots, 0.029, 0.030] \tag{4}$$

The selected DBSCAN parameters, their performance, and the resulting number of outliers for the three machines are shown in Table 4.

Afterward, the labeled data set was created for each machine. The previously identified Transient group and Outliers are labeled as anomalies ("-1"), and the rest of the Steady-state group is labeled as normal data ("1"). The final sample observations of the three labeled data sets are shown in Table 5.

TABLE 6. Hyper-parameters selection table.

Model	Hyper-parameters	Possible Values
LOF	number of neighbours	[5, 10, 20, 30, 40]
	contamination	[0.01, 0.015, 0.02, 0.025, 0.03]
OCSVM	kernel	[linear, poly, rbf]
	gamma	[0.001, 0.005, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
	nu	[0.015, 0.025, 0.035, 0.045]
Autoencoder	epochs	[5, 10, 15]
	batch size	[10, 20, 30, 40]

TABLE 7. Hyperparameters and F_1-score for each generated submodel of Machine A.

Model	Hyper-parameters	Value	F_1-Score (Validation Set) -1	1
LOF	number of neighbours	20	0.803	0.900
	contamination	0.010		
OCSVM	kernel	poly	0.887	0.936
	gamma	0.001		
	nu	0.015		
Autoencoder	epochs	5	0.462	0.642
	batch size	30		

TABLE 8. Hyperparameters and F_1-score for each generated submodel of Machine B.

Model	Hyper-parameters	Value	F_1-Score (Validation Set) -1	1
LOF	number of neighbours	20	0.838	0.924
	contamination	0.010		
OCSVM	kernel	rbf	0.861	0.922
	gamma	0.001		
	nu	0.015		
Autoencoder	epochs	15	0.367	0.645
	batch size	30		

The labeled data set was then separated into three sets: 20% Validation set, 60% Training set (with only normal data), and 20% Test set, as explained in the Manufacturing stage pipeline section. For the Training set, a grid search with cross-validation was performed with five folds ($k = 5$), where a set of hyper-parameters for each model was defined so that the search algorithm finds the best ones according to their respective F_1-score. These initial hyper-parameters are displayed on Table 6.

Tables 7, 8, and 9 show the selected hyper-parameters and the obtained F_1-score values for the three machines.

The last step of the proposed ML pipeline consisted of implementing an ensemble of three models: LOF, OCSVM, and Autoencoder, through a weighted average distribution. Autoencoder's architecture is detailed in Table 10. Table 11 shows the weights for the predictions of each model, which were determined as the ratio of each F_1-score value in Tables 7, 8, and 9 with respect to the sum of all F_1-score values for each class ("-1" and "1"). As an illustrative example, for a given sample, the LOF model predicted an anomaly (-1), the OCSVM predicted normality (1), and the Autoencoder predicted an anomaly (-1) again, each output

TABLE 9. Hyperparameters and F_1-score for each generated submodel of Machine C.

Model	Hyper-parameters	Value	F_1-Score (Validation Set) -1	1
LOF	number of neighbours	10	0.886	0.889
	contamination	0.010		
OCSVM	kernel	rbf	0.864	0.874
	gamma	0.005		
	nu	0.015		
Autoencoder	epochs	5	0.764	0.714
	batch size	30		

TABLE 10. Autoencoder's architecture.

N	Layer	Output Shape	# of Parameters
1	Input Layer	(None, 7)	-
2	Dense	(None, 4)	32
3	Dropout	(None, 4)	0
4	Dense	(None, 2)	10
5	Dense	(None, 4)	12
6	Dense	(None, 7)	35

TABLE 11. Weights for the predictions of each submodel.

Model	Machine	Weights (-1)	Weights (1)
LOF	A	0.373	0.363
	B	0.412	0.378
	C	0.215	0.259
OCSVM	A	0.406	0.371
	B	0.417	0.370
	C	0.177	0.259
Autoencoder	A	0.352	0.359
	B	0.344	0.353
	C	0.304	0.288

is multiplied by its respective weight, this computing the final classification of the hybrid model. Thus, considering the weights from Table 10, the output of the hybrid model will be 0.8. If this value is greater than 0, the hybrid model will classify it as a normal data point ("1").

IV. RESULTS

In addition to the pipeline proposed for real-time anomaly detection, the proposed hybrid model must present improved performance metrics for the individual models. In this case, the precision, recall, and F_1-score values, as well as the Area Under the ROC Curve (AUC) of all models, were compared.

A. MANUFACTURING-PIPELINE RESULTS

Three machines were selected corresponding to three different model versions to check that the hybrid models worked equally well on heterogeneous equipment.

The confusion matrix allows checking which types of hits and errors (type I or false-negative errors and type II or false-positive errors) the current models have through their different metrics, such as accuracy, precision, sensitivity, and specificity. Finally, the confusion matrix of the ensemble model was analyzed to check whether it improves the individual models' performance or not. In this respect, we focus on two metrics: i) Precision: Anomaly data are classified as normal. Also known as the False Positive Rate (FP) or Type I error. ii) Recall: Normal data are classified as an anomaly, also known as False Negative Rate (FN) or Type II error.

TABLE 12. Machine A - confusion matrix (test set).

Model LOF		Predicted	
Actual	Anomaly (-1)	120	39
	Normal (1)	2	237
		Anomaly (-1)	Normal (1)
Model OCSVM		Predicted	
Actual	Anomaly (-1)	130	29
	Normal (1)	4	235
		Anomaly (-1)	Normal (1)
Model Autoencoder		Predicted	
Actual	Anomaly (-1)	63	96
	Normal (1)	97	142
		Anomaly (-1)	Normal (1)
Model Hybrid		Predicted	
Actual	Anomaly (-1)	132	27
	Normal (1)	1	238
		Anomaly (-1)	Normal (1)

TABLE 13. Machine B - confusion matrix (test set).

Model LOF		Predicted	
Actual	Anomaly (-1)	122	31
	Normal (1)	3	271
		Anomaly (-1)	Normal (1)
Model OCSVM		Predicted	
Actual	Anomaly (-1)	137	16
	Normal (1)	23	251
		Anomaly (-1)	Normal (1)
Model Autoencoder		Predicted	
Actual	Anomaly (-1)	56	97
	Normal (1)	98	176
		Anomaly (-1)	Normal (1)
Model Hybrid		Predicted	
Actual	Anomaly (-1)	126	27
	Normal (1)	4	270
		Anomaly (-1)	Normal (1)

TABLE 14. Machine C - confusion matrix (test set).

Model LOF		Predicted	
Actual	Anomaly (-1)	174	46
	Normal (1)	2	180
		Anomaly (-1)	Normal (1)
Model OCSVM		Predicted	
Actual	Anomaly (-1)	163	57
	Normal (1)	5	177
		Anomaly (-1)	Normal (1)
Model Autoencoder		Predicted	
Actual	Anomaly (-1)	170	50
	Normal (1)	51	131
		Anomaly (-1)	Normal (1)
Model Hybrid		Predicted	
Actual	Anomaly (-1)	177	43
	Normal (1)	2	180
		Anomaly (-1)	Normal (1)

The Confusion matrix for machine A, machine B, and machine C are shown in Tables 12, 13, and 14 respectively.

The confusion matrix shows a generalized improvement of the hybrid model's performance compared to the other models in all three machines, both for recall and precision. For the experiments being analyzed, precision should be maximized as much as possible since it is indicative of the anomalous values detected by the system.

TABLE 15. Machine A - metrics table (test set).

Model	Label	Precision	Recall	F_1-score	AUC
LOF	-1	0.980	0.750	0.854	0.873
	1	0.860	0.990	0.920	
OCSVM	-1	0.970	0.820	0.887	0.900
	1	0.890	0.980	0.934	
Autoencoder	-1	0.390	0.400	0.394	0.495
	1	0.600	0.590	0.595	
Hybrid	-1	0.990	0.830	0.904	0.913
	1	0.900	1.000	0.944	

TABLE 16. Machine B - metrics table (test set).

Model	Label	Precision	Recall	F_1-score	AUC
LOF	-1	0.980	0.800	0.877	0.893
	1	0.900	0.990	0.941	
OCSVM	-1	0.860	0.900	0.875	0.905
	1	0.940	0.920	0.928	
Autoencoder	-1	0.360	0.370	0.365	0.504
	1	0.640	0.640	0.643	
Hybrid	-1	0.970	0.820	0.890	0.905
	1	0.910	0.990	0.946	

TABLE 17. Machine C - metrics table (test set).

Model	Label	Precision	Recall	F_1-score	AUC
LOF	-1	0.990	0.790	0.878	0.890
	1	0.800	0.990	0.882	
OCSVM	-1	0.970	0.740	0.840	0.856
	1	0.760	0.970	0.851	
Autoencoder	-1	0.770	0.770	0.771	0.746
	1	0.720	0.720	0.722	
Hybrid	-1	0.990	0.800	0.887	0.897
	1	0.810	0.990	0.889	

Tables 15, 16, and 17 show the models' summary results, both individually and jointly, using their metrics for comparison.

As seen in the above tables, the performance obtained by the hybrid model improves the performance of the individual models. Thus, this justifies integrating models through a hybrid model using a weighted average improves the whole pipeline's final performance. It should also be noted that the results presented by the Autoencoder are relatively low compared to the other model; this is because the Autoencoder operates better for anomaly detection using time windows and a convolutional network architecture, which is not the case. The problem of using a convolutional architecture is that it requires time windows that could add significant delay in the operation stage and would make it difficult to compare its metrics to those of the rest of the models due to the transformation of the training, validation, and testing data that is needed to be done for being able to use the data with this type of model.

B. OPERATION PIPELINE RESULTS

The above anomaly detection algorithm would not be useful if it could not process the trained models smoothly in a standard, real-time operation environment.

In order to measure performance, a data batch comprising 2012 samples was run for all individual models in a common computer (8GB RAM and a minimum of Intel Core i5 or equivalent; no graphic card required); the computation time needed to get the results was measured. After that, we ran the

TABLE 18. Performance results of each model in microseconds.

	LOF	OCSVM	Autoencoder	Hybrid
mean	803.6	175.4	34445.7	35982.6
std	2515.4	21.3	9999.4	11254.8
min	674.6	159.8	30300.3	31399.4
max	112896.7	446.4	174986.8	187873

same data for the hybrid model and analyzed the computation time needed to process the data. The results are presented in table 18.

As expected, the hybrid model was slower than the individual ones. Nevertheless, its time response is still over the real-time response threshold defined for a run-of-the-mill computer of 2020 (under 200 milliseconds in the worst loop of the batch analysis), thus achieving the objective established for the operation stage: real-time anomaly detection.

V. CONCLUSION

This research work has developed and presented a Hybrid Machine-Learning Ensemble for Anomaly Detection for a Real-Time Industry 4.0 System. This ensemble consists of implementing two stages inspired by a standard industrial system: i) A Manufacturing Stage and ii) An Operation Stage. Up to our knowledge, there are no other ML methods that consider these industrial stages. The ensemble system was tested on three machines, presenting an increased F_1-score value and AUC concerning individual ML submodels (LOF, OCSVM, and Autoencoder). The ensemble model for Machine A presented a F_1-score value of 0.904 for anomalies (-1), a F_1-score value of 0.944 for normal data (1), and an AUC value of 0.913; the ensemble model for Machine B presented a F_1-score value of 0.890 for anomalies (-1), a F_1-score value of 0.946 for normal data (1), and an AUC value of 0.905; finally, the ensemble model for Machine C presented a F_1-score value of 0.887 for anomalies (-1), a F_1-score value of 0.889 for normal data (1), and an AUC value of 0.897.

The proposed system allows vertical scaling in the number of algorithms used for the ensemble. As seen in section Results, subsection B, the hybrid model presented a maximum computation time of approximately 190 milliseconds, fast enough for real-time anomaly detection. Concerning individual models' performance, the Autoencoder results showed a low F_1-score value, so it is proposed to test other algorithms (e.g., Isolation Forest, Elliptic Envelope) to improve the overall performance of the whole assembly. However, a study of the computational cost linked to the retraining of more types of algorithms must be carried out.

Future work is proposed to study system retraining in the Operation Stage pipeline and its computational cost. It is also proposed to study the proposed system developed on machines with different levels of degradation. Additionally, a data imputation study should be carried out to generate synthetic samples for systems where some information is missing (a loss of data due to communication breakdowns is a common problem in industrial systems). Deep Learning techniques could be considered when creating meta-classifiers using different base classifiers such as recurrent neural networks, like LSTMs, where time series need to be considered. Furthermore, a study with a larger number of machines must be carried out to see how well the hybrid model generalizes against the individual sub-models. In cases where the hybrid model does not provide any improvement, other ensemble strategies such as taking the best of the individual sub-models are considered.

Finally, as this project focuses on single-type anomaly detection, a challenge to be addressed in future work will be to be able to classify or categorize different types of faults. For that, the authors might use appropriate methods such as explainable ML or correspondingly labeled datasets.

ACKNOWLEDGMENT

The authors would like to thank the Vicomtech Foundation for providing the necessary resources for the proper execution of this research project and University EAFIT for the research grant awarded to the principal author.

REFERENCES

[1] M. Xu, J. M. David, and S. H. Kim, "The fourth industrial revolution: Opportunities and challenges," *Int. J. Financial Res.*, vol. 9, no. 2, pp. 92–95, 2018.
[2] M. Reis and G. Gins, "Industrial process monitoring in the big data/industry 4.0 era: From detection, to diagnosis, to prognosis," *Processes*, vol. 5, p. 35, Jun. 2017. [Online]. Available: http://www.mdpi.com/2227-9717/5/3/35
[3] S. H. An, G. Heo, and S. H. Chang, "Detection of process anomalies using an improved statistical learning framework," *Expert Syst. Appl.*, vol. 38, no. 3, pp. 1356–1363, Mar. 2011.
[4] A. Boukerche, L. Zheng, and O. Alfandi, "Outlier detection: Methods, models, and classification," *ACM Comput. Surv.*, vol. 53, no. 3, pp. 1–37, May 2021.
[5] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. D. Carvalho, and J. Gama, "Data stream clustering: A survey," *ACM Comput. Surv.*, vol. 46, no. 1, pp. 1–31, 2013.
[6] S. Ahmad, A. Lavin, S. Purdy, and Z. Agha, "Unsupervised real-time anomaly detection for streaming data," *Neurocomputing*, vol. 262, pp. 134–147, Nov. 2017.
[7] V. Chandola, V. Mithal, and V. Kumar, "Comparative evaluation of anomaly detection techniques for sequence data," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 743–748.
[8] J. Rabatel, S. Bringay, and P. Poncelet, "Anomaly detection in monitoring sensor data for preventive maintenance," *Expert Syst. Appl.*, vol. 38, no. 6, pp. 7003–7015, Jun. 2011.
[9] V. Vercruyssen, W. Meert, G. Verbruggen, K. Maes, R. Baumer, and J. Davis, "Semi-supervised anomaly detection with an application to water analytics," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2018, pp. 527–536.
[10] M. Fahim and A. Sillitti, "Anomaly detection, analysis and prediction techniques in IoT environment: A systematic literature review," *IEEE Access*, vol. 7, pp. 81664–81681, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8733806, doi: 10.1109/ACCESS.2019.2921912.
[11] B. R. Priyanga and D. Kumari, "A survey on anomaly detection using unsupervised learning techniques," *Int. J. Creative Res. Thoughts (IJCRT)*, vol. 6, no. 2, pp. 2320–2882, 2018. [Online]. Available: http://www.ijcrt.org/papers/IJCRT1812118.pdf
[12] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for discrete sequences: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 5, pp. 823–839, 2012. [Online]. Available: https://ieeexplore.ieee.org/document/5645624, doi: 10.1109/TKDE.2010.235.
[13] A. I. Rana, G. Estrada, M. Sole, and V. Muntes, "Anomaly detection guidelines for data streams in big data," in *Proc. 3rd Int. Conf. Soft Comput. Mach. Intell. (ISCMI)*, Nov. 2016, pp. 94–98.
[14] M. Hubert and E. Vandervieren, "An adjusted boxplot for skewed distributions," *Comput. Statist. Data Anal.*, vol. 52, no. 12, pp. 5186–5201, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167947307004434

[15] S. Agrawal and J. Agrawal, "Survey on anomaly detection using data mining techniques," *Proc. Comput. Sci.*, vol. 60, pp. 708–713, Jan. 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050915023479

[16] S. Ferreiro, B. Sierra, I. Irigoien, and E. Gorritxategi, "A Bayesian network for burr detection in the drilling process," *J. Intell. Manuf.*, vol. 23, no. 5, pp. 1463–1475, Oct. 2012, doi: 10.1007/s10845-011-0502-z.

[17] B. Sierra, E. Lazkano, E. Jauregi, and I. Irigoien, "Histogram distance-based Bayesian network structure learning: A supervised classification specific approach," *Decis. Support Syst.*, vol. 48, no. 1, pp. 180–190, Dec. 2009.

[18] Y. Yuan, S. Li, X. Zhang, and J. Sun, "A comparative analysis of SVM, naive Bayes and GBDT for data faults detection in WSNs," in *Proc. IEEE Int. Conf. Softw. Qual., Rel. Secur. Companion (QRS-C)*, Jul. 2018, pp. 394–399.

[19] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1–58, Jul. 2009, doi: 10.1145/1541880.1541882.

[20] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," *ACM SIGMOD Rec.*, vol. 29, no. 2, pp. 93–104, Jun. 2000.

[21] P.-Y. Chen, S. Yang, and J. A. McCann, "Distributed real-time anomaly detection in networked industrial sensing systems," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3832–3842, Jun. 2015.

[22] S. Lee, G. Kim, and S. Kim, "Self-adaptive and dynamic clustering for online anomaly detection," *Expert Syst. Appl.*, vol. 38, no. 12, pp. 14891–14898, Nov. 2011.

[23] E. H. M. Pena, S. Barbon, J. J. P. C. Rodrigues, and M. L. Proenca, "Anomaly detection using digital signature of network segment with adaptive ARIMA model and paraconsistent logic," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2014, pp. 1–6.

[24] S. C. Tan, K. M. Ting, and T. F. Liu, "Fast anomaly detection for streaming data," in *Proc. IJCAI Int. Joint Conf. Artif. Intell.*, 2011, pp. 1511–1516.

[25] N. Ding, H. Ma, H. Gao, Y. Ma, and G. Tan, "Real-time anomaly detection based on long short-term memory and Gaussian mixture model," *Comput. Electr. Eng.*, vol. 79, Oct. 2019, Art. no. 106458.

[26] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation-based anomaly detection," *ACM Trans. Knowl. Discovery Data*, vol. 6, no. 1, pp. 1–39, 2012.

[27] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, Mar. 2014, Art. no. 44. [Online]. Available: https://dl.acm.org/doi/10.1145/2523813, doi: 10.1145/2523813.

[28] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2001, pp. 97–106.

[29] N. Laptev, S. Amizadeh, and I. Flint, "Generic and scalable framework for automated time-series anomaly detection," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2015, pp. 1939–1947.

[30] D. Velásquez, A. Sánchez, S. Sarmiento, M. Toro, M. Maiza, and B. Sierra, "A method for detecting coffee leaf rust through wireless sensor networks, remote sensing, and deep learning: Case study of the caturra variety in Colombia," *Appl. Sci.*, vol. 10, no. 2, p. 697, Jan. 2020.

[31] J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa, "Ensemble approaches for regression: A survey," *ACM Comput. Surv.*, vol. 45, no. 1, p. 10, Nov. 2012.

[32] N. Garcia-Pedrajas, C. Hervas-Martinez, and D. Ortiz-Boyer, "Cooperative coevolution of artificial neural network ensembles for pattern classification," *IEEE Trans. Evol. Comput.*, vol. 9, no. 3, pp. 271–302, Jun. 2005.

[33] J. Kittler, M. Hatef, R. P. W. Duin, and J. Matas, "On combining classifiers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 3, pp. 226–239, Mar. 1998.

[34] H. Judi, R. Jenal, and D. Genasan, *Quality Control Implementation in Manufacturing Companies: Motivating Factors and Challenges*. London, U.K.: IntechOpen, Apr. 2011, ch. 25.

[35] A. Jovic, K. Brkic, and N. Bogunovic, "A review of feature selection methods with applications," in *Proc. 38th Int. Conv. Inf. Commun. Technol., Electron. Microelectron. (MIPRO)*, May 2015, pp. 1200–1205.

[36] E. Schubert, J. Sander, M. Ester, H. Kriegel, and X. Xu, "DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN," *ACM Trans. Database Syst.*, vol. 42, no. 3, pp. 1–21, 2017.

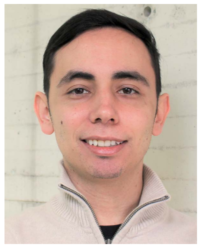
[37] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.

[38] D. M. Tax and R. P. Duin, "Uniform object generation for optimizing one-class classifiers," *J. Mach. Learn. Res.*, vol. 2, pp. 155–173, Dec. 2002.

[39] T. Amarbayasgalan, B. Jargalsaikhan, and K. Ryu, "Unsupervised novelty detection using deep autoencoders with density based clustering," *Appl. Sci.*, vol. 8, no. 9, p. 1468, Aug. 2018.

[40] A. Zheng and A. Casari, *Feature Engineering for Machine Learning: Principles and Techniques for Data Scientists*, 1st ed. Sebastopol, CA, USA: O'Reilly Media, 2018.

[41] D. Freedman, R. Pisani, and R. Purves, *Statistics: Fourth International Student Edition* (International Student Edition). New York, NY, USA: W. W. Norton & Company, 2007. [Online]. Available: https://books.google.es/books?id=mviJQgAACAAJ

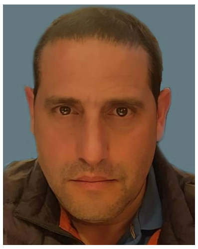


DAVID VELÁSQUEZ received the B.S. degree in mechatronics engineering from the University Escuela de Ingeniería de Antioquia (EIA), in 2011, and the master's degree in engineering from Universidad EAFIT, with emphasis on technical systems integrated design, in 2014. He is currently pursuing the Ph.D. degree in informatics with the University of the Basque Country, Spain, in collaboration with research projects from the VICOMTECH Research Center. He is also working as an Assistant Professor with the Department of Systems and Informatics Engineering and as a Researcher with the TICs Development and Innovation Research Group (GIDITIC) and the Design Engineering Research Group (GRID), Universidad EAFIT. His research interests include adaptive systems control design, mechatronics design, industry 4.0, machine learning, computer vision, electronics optimization, embedded systems, the Internet of Things implementation, and biomedical signal processing applications.

ENRIQUE PÉREZ received the graduate degree in information technology engineering from the Universidad Nacional de Educación a Distancia (UNED), in 2019. He is currently pursuing the master's degree in data science with the Universitat Oberta de Catalunya (UOC), carrying out the external end of master's work in the field of artificial intelligence, developing a proposal for intelligent services for industrial blowers with the VICOMTECH Research Center, Data Intelligence for Energy and Industrial Processes Department, through an educational cooperation agreement between the university and company. He carried out the end-of-degree project (PFG) in the field of machine learning (ML) associated with predictive maintenance in industry 4.0 environments. His research interests include the field of machine learning and deep learning (DL), creation of predictive models through advanced analytics in the Industrial Internet of Things (IIoT) systems, data visualization, and its practical application for the industry 4.0.

XABIER OREGUI received the Ph.D. degree in telecommunications engineering from the Centro de Estudios e Investicación Técnicas (CEIT), University of Navarra, more precisely in the area of electronics and communications, where he researched about multi-source virtual machine management and automatic scaling. After a small recess from research, where he spent his acquired knowledge developing educational games and "serious-games" for the company Ikasplay, in 2016, he camed back to the research world in Vicomtech to the area of data intelligence and industrial processes. Back on his incorporation on Vicomtech, he has been working on multiple projects focused on data management on industrial environments using different kinds of protocols, and projects oriented on the management and analysis on big data and the visualization of that information from that same environment.

ARKAITZ ARTETXE received the degree in computer engineering and the M.Sc. degree in computational engineering and intelligent systems from the University of the Basque Country (UPV/EHU), San Sebastian, in 2011 and 2014, respectively, and the Ph.D. degree in computer science with emphasis on the application of knowledge engineering and machine learning to the medical domain from the University of the Basque Country, in 2017. Since 2011, he has been working as a Researcher in the field of biomedical applications with the Technological Centre Vicomtech. Since 2018, he has been working as a Researcher with the Data Intelligence for Energy and Industrial Processes Department, Vicomtech. His research interests include machine learning, imbalanced classification, and data fusion techniques in the context of industry 4.0.



JORGE MANTECA is currently a Senior Industrial Engineer (specialty mechanics, machines) with ETSIIG, University of Oviedo. He is also the Technical Director of MAPNER, a company dedicated to the manufacture of machinery in the field of compressors and vacuum pumps. As main functions, he is an in-charge of technical support in commercial tasks, management and implementation of resources for the organization of the different departments of the company, and management and coordination of the technical office, research, optimization of equipment performance, and development of new products. His previous experience includes his participation in a research project on the behavior of cryogenic fluids at CERN (European Center for Particle Physics Research), Geneva, Switzerland. He has published several scientific papers, including the publication in the international conference on cryogenics in Anchorage (Alaska), in 2003:"CONCLUSION OF THE HE SPILL SIMULATIONS IN THE LHC TUNNEL."

JORDI ESCAYOLA MANSILLA received the bachelor's and master's degrees in statistics and operations research, the master's (Executive) degree in business administration, and the Ph.D. degree in economics and business. He has been working in data science and predictive analytics for the last 11 years in different industries (insurance, banking, pharma, and public sector) and consulting, which includes a relevant multinational experience helping organizations and governments. Since 2017, he has been a fellow of the prestigious organization Betta Gamma Sigma, which represents one of the highest honor institutions in business worldwide. He is currently the Consultancy Manager and the Practice Leader in data science, an Associate Professor in probability and statistics with the Universitat Politècnica de Catalunya (UPC), and an Associate Professor with the Universitat Oberta de Catalunya (UOC).

MAURICIO TORO received the B.S. degree in computer science and engineering from Pontificia Universidad Javeriana, Colombia, in 2009, and the Ph.D. degree in computer science from the Université de Bordeux, France, with emphasis on artifcial intelligence, in 2012. He has been a Postdoctoral Fellow with the Computer-Science Departament, University of Cyprus, since 2013. Since 2014, he has been working as an Assistant Professor with the Department of Systems and Informatics Engineering and as a Researcher with the TICs Development and Innovation Research Group (GIDITIC), Universidad EAFIT. His research interests include artificial intelligence, industry 4.0, machine learning, computer vision, and agricultural applications.

MIKEL MAIZA received the degree in automatic engineering and industrial electronics from the University of Mondragon, in 2000, and the Ph.D. degree from the University of York, U.K., in 2003, with emphasis on parallel computing for real-time systems. He was an External Professor with the University of Mondragon, from 2002 to 2004, an Associate Professor with the School of Engineering, University of Navarra, from 2009 to 2013, and an Associate Professor with the Department of Applied Mathematics, University of the Basque Country, from 2015 to 2017. He has been collaborating as an External Professor with the Ecole Supérieure des Technologies Industrielles Avancées (ESTIA), since 2017. Since 2016, he has been working as a Senior Researcher with the Technological Centre Vicomtech, Data Intelligence for Energy and Industrial Processes Department. His research interests include parallel processing systems, heuristic algorithms and stochastic techniques of mathematical optimization and their integration with artificial intelligence techniques, for building stochastic models aimed at the simulation and optimization of processes and systems, and applications, such as data mining, pattern recognition, and automatic learning or early fault detection.

BASILIO SIERRA is currently a Full Professor with the Computer Sciences and Artificial Intelligence Department, University of the Basque Country (UPV/EHU). He is also the Co-Director of the Robotics and Autonomous Systems Group, RSAIT. He is also a Researcher in the fields of robotics and machine learning, where he is working on the use of different paradigms to improve robot's behaviors. He works as well in multidisciplinary applications of machine learning paradigms, in agriculture, natural language processing, and medicine. He has published more than 50 journal articles, and several book chapters and conference papers.