A Semi-Supervised Bayesian Anomaly Detection Technique for Diagnosing Faults in Industrial IoT Systems

Fabrizio De Vita[†], Dario Bruneo[†], and Sajal K. Das[‡]
[†]*Department of Engineering, University of Messina, Italy*
[‡]*Department of Computer Science, Missouri University of Science and Technology, USA*
{fdevita, dbruneo}@unime.it, sdas@mst.edu

Abstract—**The Industry 4.0 paradigm has changed the way industrial systems with hundreds of sensor-actuator enabled devices, including industrial internet of things (IIoT), cooperate and communicate with the physical and human worlds. Given the intricacy, the diagnostics of such systems is extremely important. While anomaly detection is a valid approach to avoid unplanned maintenance or even complete breakdown, its effective realization in IIoT requires the design and implementation of frameworks for efficient monitoring, data collection, and analysis. Most of the existing anomaly detection techniques provide only a diagnosis of the fault without taking into account the uncertainty. Moreover, the lack of ground truth data (which is a typical problem in the industrial context), make their implementation even more challenging. This paper proposes an anomaly detection technique built on top of an industrial framework for the data collection and monitoring. Specifically, we address the lack of labeled data by designing a semi-supervised anomaly detection algorithm that exploits Bayesian Gaussian Mixtures to assess the working condition of the plant while measuring the uncertainty during the diagnosis process and we implement the proposed framework on a real-life IIoT testbed, namely a scale replica assembly plant. Experimental results demonstrate that our anomaly detection algorithm is able to detect the plant working conditions with 99.8% of accuracy, and the semi-supervised approach performs better than a supervised one.**

Index Terms—**Industry 4.0, IIoT, Anomaly Detection, Bayesian Gaussian Mixtures.**

I. Introduction

The advancement of information and communications technology (ICT) and smart sensing capabilities have revolutionized the industrial sector [1]. Indeed, the interconnected network of smart devices equipped with sensors and actuators enable the industrial systems interact with the physical and human worlds [2]. This paradigm shift involving Industrial Internet of Things (IIoT) is referred to as Industry 4.0 [3].

In Industry 4.0, effective fault diagnosis (assessment of working conditions) is extremely important because timely detection of a condition or event, which is not normal (i.e., anomalous), can avoid unplanned/unwanted system maintenance or even complete breakdown that could be very expensive. Undoubtedly, efficient diagnostics depends on the implementation of underlying frameworks for data collection, monitoring, and analysis. In an IIoT scenario, the heterogeneity of sensors and their communication protocols as well as the absence of labeled ground truth data pose significant challenges for the design and implementation of effective diagnostic systems.

For efficient realization of IIoT frameworks, the edge and cloud computing offer key enabling technologies [4]. Specifically, edge computing helps implement simple operations like preliminary on-board analysis. Although edge devices are resource-constrained, they can provide desired results very quickly, which make them suitable for monitoring industrial processes that typically work at a high frequency rate. Similarly, cloud computing helps store and keep track of the system's historical data and perform complex operations (in-depth analysis, execution of algorithms) that require higher computational power as compared to edge devices. The cloud also plays an important role in controlling and coordinating IIoT device operations in smart industrial plants. Moreover, AI and machine learning techniques not only improve (autonomous) decisions [5] by reducing uncertainty, but also develop smart applications for better diagnostics and maintenance.

In this paper, we propose an anomaly detection technique that we built on top of an IIoT framework for efficient data collection and monitoring of a real industrial plant. To design and implement the software architecture, we leverage an OpenStack module *Stack4Things* (S4T), a cloud based platform that controls and orchestrates the IoT devices without caring about their location, network configuration or technology [6]. In order to address the lack of data, which is typical for the industrial context, we tackled the anomaly detection problem as a semi-supervised approach and we exploited the Bayesian theory to model the uncertainty during the diagnosis process.

Our novel contributions are summarized as follows.

i) To assess working conditions of a real industrial testbed while addressing the lack of labeled ground truth data, we propose a semi-supervised machine learning based anomaly detection algorithm that exploits the Bayesian theory to model uncertainty in the diagnosis process.
ii) Based on Stack4Things, we implemented an IIoT framework, called Industrial Stack4Things (IS4T) for monitoring, data collection and labelling of an industrial plant.

iii) We created a scale replica of an industrial assembly plant and use it as a testbed for experimentation to validate the performance of the proposed anomaly detection technique and IIoT framework in a real setting.

The rest of the paper is organized as follows. Section II reviews related works and Section III describes the scale replica industrial plant testbed. Section IV provides a brief description of the IIoT framework we realized for data collection and monitoring. Section V presents the anomaly detection algorithm built on top of the IIoT framework to assess the working conditions of the industrial plant and detect anomalies. Section VI summarizes the experimental results. Finally, Section VII concludes the paper with directions of future research.

II. Related Work

A recent survey [3] described the challenges and future directions in Industry 4.0. In the following, we summarize existing works highlighting their differences from our approach.

In [7], a performance evaluation of two architectures is presented for industrial IoTs, called full-cloud and edge-cloud, on top of which is performed an anomaly detection scheme based on the Long Short Term Memories (LSTM). This approach consists of a single soft power line communication (PLC) to communicate with the IIoT computer using only Message Queue Telemetry Transport (MQTT) protocol. However, in a real industrial scenario, the number of communication protocols adopted by heterogeneous sensors can be quite large. Our proposed framework is designed to work with different communication protocols, making it suitable for Industry 4.0.

A Dirichlet Process Gaussian Mixed Model is proposed in [8] for fault detection in multi-mode processes. Although good results were obtained from a wastewater treatment plant, this work does not provide insights into fault detection uncertainty. In contrast, we demonstrate the feasibility of our anomaly detection scheme in a real scale replica industrial plant.

An IoT framework proposed in [9] monitors chemical emissions in an industrial plant. It consists of three layers: a lower layer of ZigBee nodes, a middle layer of long range (LoRa) nodes, and an upper layer providing connection to the Internet to store sensory data in an SQL-database. However, in an industrial setting, the interference due to a variety of devices can be very high; hence the use of only wireless communication protocols may result in a significant data loss.

In [10] is presented a causal-polytree anomaly detection framework for Supervisory Control and Data Acquisition (SCADA) systems, which uses a naive Bayes classifier for anomaly detection, preventing the system from malicious attacks. In this approach, the data acquisition and anomaly detection are integrated. Whereas, the synergy between data collection and anomaly detection makes our proposed framework more suitable for industrial applications.

III. Scale replica assembly plant

The IIoT testbed used for design and testing of our anomaly detection technique consists of a scale replica of an assembly plant for transportation of car pieces (see Fig. 1), equipped with two motors and six belts for transport of a mobile cart.

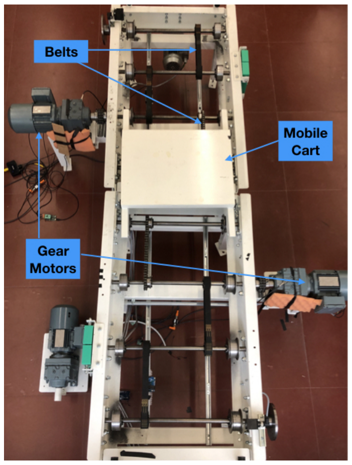


Fig. 1: The IIoT testbed.

We instrumented the IIoT system for monitoring purpose, both from electrical and mechanical points of view. For the electrical part, we used a non-invasive current sensor SCT-013 to detect current spikes or power supply malfunctions that can be related to the system overloading and short-circuits in the electric network or mechanical stress (i.e., gears friction).

For mechanical part, we used a vibration sensor VTV-122 by IFM electronics and a noise sensor SPQ0410HR5H-B by Mikroe, to monitor anomalous conditions due to unbalanced loads or proximity switch malfunctions. Finally, to monitor the engine's working conditions, we added a temperature sensor and a distance sensor to measure the tension of one of the rubber belts, preventing severe engine damages due to excessive strains. Since it is not possible to directly access the inside of the engines, we used a TS2229 sensor. On the other hand, the belt tension is measured using state-of-the-art VL53L0X distance sensor manufactured by STmicroelectronics

As mentioned, in a large scale industrial plant the heterogeneity of sensors using different communication protocols (i.e., 4-20ma, I2C, SPI, etc.), poses a major challenge due to the requirement of managing the sensors separately. (This is true even for our small IIoT testbed consisting of 5 sensors). This motivated us to build an efficient IIoT framework that will not only simplify the data collection process, but also serve as an abstraction layer between the embedded boards interacting with the sensors and the data acquisition system.

In such a context, an important feature of our testbed is the ability to inject manual faults, mostly mechanical, inside the plant, such as introduce external vibration inside the structure, change the belt tension, increase the friction of the gears by acting on the system brakes, and deactivate cart proximity switch, thus emulating a mechanical problem of the belts. This feature is instrumental for our machine learning algorithm to characterize the system behaviour, such as normal or anomalous operating conditions. Moreover, the possibility to manually inject the fault gives us the opportunity to build a labeled dataset in correspondence of the different operating conditions of the system and resulted to be fundamental for the validation of the proposed anomaly detection technique.

IV. Industrial Stack4Things (IS4T) Architecture

This section presents IS4T, the IIoT framework adapted for data collection and telemetry of the industrial plant. Figure 2 depicts the overall architecture consisting of three layers.

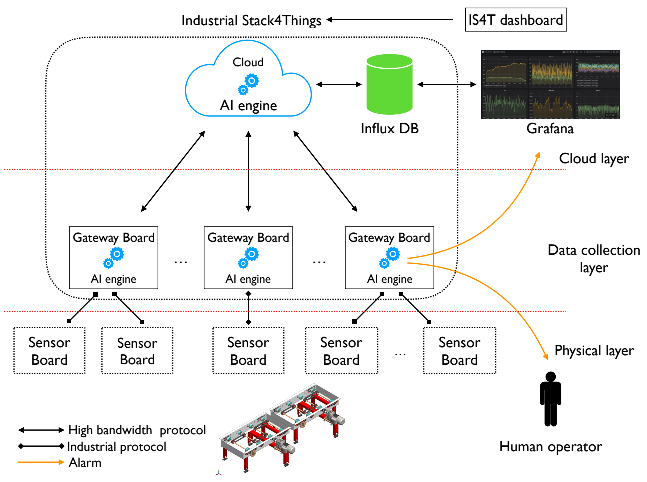


Fig. 2: Proposed IIoT framework.

The first layer is the physical layer closest to the plant itself. The micro-controllers of different sensor boards perform computations over electric signals generated by the sensors that provide measurements to the upper layer. The sensor board is equipped with a micro-controller unit (MCU) and capable of managing several sensor interfaces such as I2C, SPI, 4-20mA, and others. Moreover, owing to different expansion shields, the board can also work with traditional communication protocols like Bluetooth Low Energy (BLE) and Wi-Fi, or industry oriented protocols like MODBUS and CAN.

The second layer, called data collection layer, acts as a bridge between the edge and the cloud. One of the biggest challenges here is that the communication protocols and the format of sensory data from the physical layer can be very different; this layer has to collect, manage, and organize the data to make them homogeneous. Moreover, the collected data are analyzed by an AI engine for early anomaly detection running on the edge. In case of an anomaly, an alarm is generated and sent to the operator and the upper layer.

In the third layer, represented by the cloud, a SQL-like time series database maintains the entire history of the plant by storing the data coming from the data collection layer. We decided to adopt *InfluxDB*[1], a non-relational database particularly optimized for embedded devices due to its very high read/write speeds. Just like the previous layer, the cloud is provided with an AI engine, that can perform both the training and anomaly detection on the collected data thanks to the higher computing power available.

The gateway board (Arancino.cc[2] board) exhibits a double environment made of MCU with a larger number of interfaces to interact with the external world, and a micro-processing unit (MPU) provided by a Raspberry compute module that extends the board capabilities with computing power. The board also manages wired/wireless communication links with different sets of boards, and uses the computing module for early processing of gathered data directly on the edge.

For data visualization, we adopted a tool called *Grafana*[3], which is integrated with the InfluxDB database by connecting to it and showing the plant monitoring data through a dashboard. Grafana also notifies the occurrence of an anomaly inside the plant with an alarm on the dashboard.

The labeling of the data collected through the IS4T framework is accomplished automatically by defining the starting and ending dates and time and selecting the label representative for the operating conditions of the system (e.g., working, proximity switch break, gears friction, etc.). By doing so, the system automatically generates an insertion query to the InfluxDB database storing the collected data together with the associated label. In this sense, the framework introduces a new level to the data collection process providing a labelling mechanism which requires a minimal interaction with the human being and adding more information to the data.

Finally, The *IS4T dashboard* accessed by a normal web browser, interacts with the IS4T functionalities.

V. Semi-Supervised Anomaly Detection Technique

This section presents the proposed anomaly detection technique on the collected data through the IS4T framework and used to assess the working conditions of the IIoT testbed.

Given diagnosis or prognosis of an IIoT system is an important task, a viable approach is to equip the system with alarms that warn the human operator about the operating conditions. However, the main drawback is that it may be too late by the time the user is warned [11].

In the context of Industry 4.0, the heterogeneity of a large number of sensors and protocols poses further challenges to correlate all the data and determine if the industrial system is properly working or not. To this end, machine learning based anomaly detection can be a valid approach [12]. Indeed, a timely detection of a faulty condition (or event) that is not considered normal (i.e., anomalous) may avoid catastrophic consequences such as breakdown, and/or unplanned maintenance [13].

Although anomaly detection is a viable solution, the majority of the proposed techniques do not consider the uncertainty of the diagnosis process. In the industrial scenario, where the generation of *false alarms* can be very expensive in terms of time and money, we believe that the uncertainty characterization will result in a better and more intelligent anomaly detection scheme [1]. Bayesian theory is a powerful tool to model the uncertainty.

Another challenge is the lack of ground truth (i.e., anomalous) data essential for validation of anomaly detection. The

[1]https://www.influxdata.com, accessed February 2021
[2]https://arancino.cc, accessed February 2021
[3]https://grafana.com, accessed February 2021

absence of historical databases or the use of intermittent data recording techniques result in poor labeled datasets. Semi-supervised learning can address this issue as it can work with a small number of labeled data samples (e.g., the number of labeled samples range between 10% and 20% of the total dataset) during the training phase. However, this technique assumes *data continuity* (i.e., close datapoints share the same label) and *data clustering* (i.e, datapoints that share the same label are likely to form a cluster).

A. Data Collection

The possibility to manually inject faults on the scale replica industrial plant helps to observe our system when subject to mechanical stress. Using our IIoT framework, we collected a dataset of $20,008$ samples under three different operating conditions: a normal one during which the mobile cart of the industrial plant can move back and forth, and two anomalous conditions during which the plant was subject to a mechanical stress induced by simulating the break of the cart proximity switch (referred to as *Anomaly 1*), or increasing the friction of the gears (referred to as *Anomaly 2*).

The following methodology was adopted for data collection. We collected data under normal operating condition for about one hour. Then, we injected the first fault (i.e., Anomaly 1) and collected the data for two and half hours. To avoid an excessive mechanical stress, we returned to data collection under normal working condition for one hour. Finally, we injected the second fault (i.e., Anomaly 2) and collected the data for another hour.

Since the injected anomalies generate a mechanical stress, we selected only a subset of the available sensors to perform the anomaly detection. In particular, those sensors were chosen that exhibited a visible change in the trend so far as the three system operating conditions are concerned.

Fig. 3 depicts the raw data measurements gathered from vibration, the three phases of the current (only one is showed in the plot for a better visualization), and distance sensors. Despite their high variability, it is possible to notice three distinctive trends that allowed us to discriminate between the normal and anomalous conditions. For data pre-processing, we applied a time overlapping sliding window and extracted the maximum, minimum, mean, and standard deviation for each of the above signals (except for the current, which is a periodic signal; hence we extracted only the maximum value). The sliding window width has been fixed empirically to the value of 400 samples. At the end of this procedure, we obtained the final labeled dataset of $19,177$ samples.

This dataset had $12,103$ anomalies and 11 features consisting of distance maximum, minimum, mean, and standard deviation values; current first phase, second phase and third phase maximum values; vibration maximum, minimum, mean, and standard deviation values. Fig. 4 depicts these 11 features that exhibit a smoother trend. Finally, in order to maintain each of the above mentioned features in the same range values, we applied a normalization process using a min-max scaling technique.

B. Semi-supervised Bayesian Gaussian Mixtures

In this subsection, we provide a detailed description of the proposed anomaly detection technique.

Given the aim is to cluster and separate normal and anomalous datapoints, let us model the uncertainty during the diagnosis process using Bayesian statistics. Specifically, we attempt to characterize the uncertainty and measure a degree of belief (i.e., the probability) as a combination of what happens before and after checking the evidence. For this purpose, Bayesian Gaussian Mixtures (BGM) help cluster a set of datapoints while measuring the level of uncertainty. Compared to traditional clustering algorithms (e.g., K-Means), the BGM can be considered as a more generalized approach because it assumes clusters as ellipsoids with different sizes and orientations in space. Due to its probabilistic nature, the algorithm performs a *soft clustering* by computing the probability that a datapoint belongs to a generic cluster instead of making a direct assignment (*hard clustering*). Such a feature allows us to measure the level of uncertainty (respectively, confidence) in the diagnosis of datapoints.

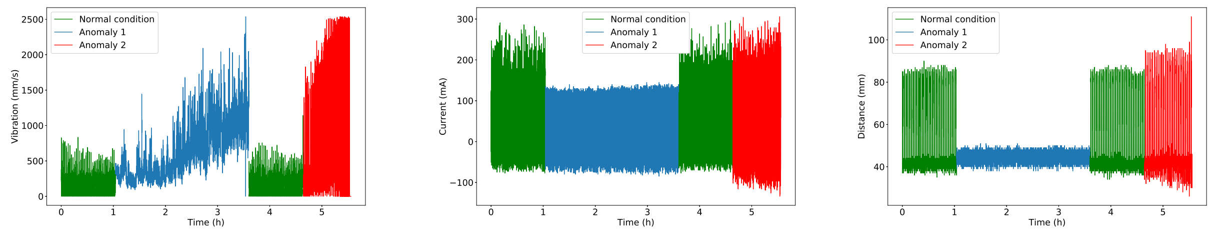
From a mathematical standpoint, a BGM is a probabilistic model that assumes the data is generated by a mixture of Gaussian distributions with unknown parameters. The BGM considers each Gaussian distribution as a cluster whose parameters are the mean (μ), the covariance matrix (Σ), and the weight (w) corresponding to a scale factor.

Given a mixture of K Gaussian distributions, the Probability Density Function (PDF) is defined as: $p(x) = \sum_{k=1}^{K} w^{(k)} \cdot \mathcal{N}(x; \mu^{(k)}, \Sigma^{(k)})$, where the constraints $0 < w^{(k)} < 1$ and $\sum_{k=1}^{K} w^{(k)} = 1$ must be satisfied to obtain a valid Gaussian mixture with the corresponding PDF integrating to 1.

During the training phase, for each datapoint x_i, the algorithm associates a latent random variable $z_i \in [1, 2, ..., K]$. These random variables follow a multi-nomial distribution $Mult(w^{(1)}, w^{(2)}, ..., w^{(K)})$ such that $p(z_i = k) = w^{(k)}$ is the probability of assigning the i^{th} datapoint to the k^{th} cluster. Once a datapoint is assigned to a cluster, the probability (or the likelihood) to observe it in that cluster is computed as $p(x_i|z_i = k) = \mathcal{N}(x_i; \mu^{(k)}, \Sigma^{(k)})$, assuming that the datapoints are be sampled from a Gaussian distribution. Now the PDF equation can be rewritten as $p(x) = \sum_{k=1}^{K} p(z = k) \cdot p(x|z = k)$, where $p(z)$ is the prior distribution related to the cluster assignments, and $p(x|z = k)$ is the likelihood that a datapoint falls in the k^{th} cluster.

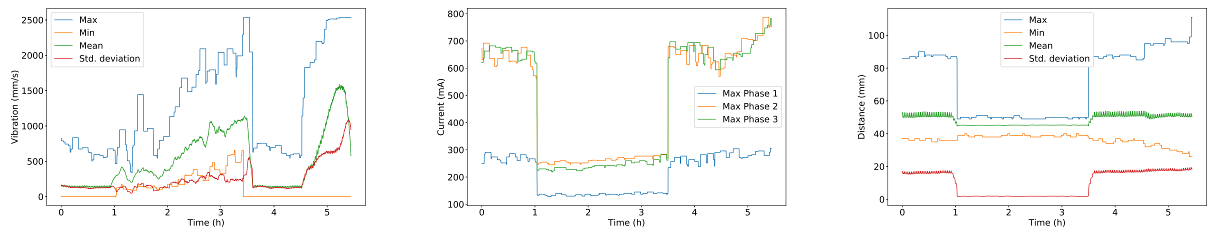
Besides the cluster assignments, the BGM also treats the cluster weights, the means, and the covariance matrices as random variables. The BGM algorithm finds the set of parameters which best fit the data. Let θ denote the vector corresponding to these random variables and let X be a set of observable datapoints (i.e., the evidence). Then the BGM applies the Bayes theorem to update the probability distributions of these variables as follows: $p(\theta|X) = \frac{p(\theta) \cdot p(X|\theta)}{p(X)}$.

However, the denominator is intractable as it would require to consider all possible values of θ (i.e., all possible clusters assignments, shapes, sizes and orientations in space). Vari-



(a) Vibration (b) Current (c) Distance

Fig. 3: Plots of the raw features of the system considering anomalous and non-anomalous working conditions.



(a) Vibration (b) Current (c) Distance

Fig. 4: Plots of the extracted features after the raw data preprocessing.

ational inference, a method from machine learning that approximates probability densities through optimization, solves this problem by picking a family of variational distributions, denoted as $Q(\theta)$, and then finding the set of parameters λ that minimizes the Kullback-Leibler divergence (KL-divergence): $D_{KL}(Q||P) = \sum_{\theta} Q(\theta;\lambda) \cdot log(\frac{Q(\theta;\lambda)}{p(\theta|X)})$.

Since the KL-divergence measures similarity between two probability distributions, its minimization is equivalent to finding variational distribution to best approximate the posterior distribution. This process is repeated by the BGM algorithm such that the posterior distribution becomes the new prior distribution at the next iteration until reaching the convergence.

The above algorithm performs clustering in an unsupervised way. However, for anomaly detection, we require labeled samples to distinguish if the data contained in a cluster is normal or anomalous. To apply the algorithm to Industry 4.0, we extended it by proposing a semi-supervised approach.

The methodology for anomaly detection using the semi-supervised BGM algorithm is as follows. We first train our model in an unsupervised way without considering any data labels. At this point, the BGM is able to identify the clusters, but without knowing if these clusters are related to normal or anomalous data (i.e., a faulty condition). After the training process, we consider a small portion of "normal" labeled datapoints (those related to normal operating condition of the system). Let $\tilde{X}$ be a set of $m << N$ normal labeled datapoints. For each sample, we perform a soft clustering by computing the likelihood for each one of the K clusters using the likelihood equation $p(x_i|z_i = k)$ and returning a matrix:

$$P = \begin{bmatrix} p(\tilde{x}_1|z_1 = 1) & p(\tilde{x}_1|z_1 = 2) & ... & p(\tilde{x}_1|z_1 = K) \\ p(\tilde{x}_2|z_2 = 1) & p(\tilde{x}_2|z_2 = 2) & ... & p(\tilde{x}_2|z_2 = K) \\ ... & ... & ... & ... \\ p(\tilde{x}_m|z_m = 1) & p(\tilde{x}_m|z_m = 2) & ... & p(\tilde{x}_m|z_m = K) \end{bmatrix}, \tag{1}$$

where the generic $P_{i,j}$ element of the matrix P represents the probability that the i^{th} datapoint belongs to the j^{th} cluster.

Assuming the semi-supervised assumptions hold, we expect that the majority of the normal datapoints have very high likelihoods to fall in the same clusters. This information can be exploited to distinguish between the normal clusters and the anomalous ones. For this purpose, we compute the mean along each column of the P matrix, thus obtaining a vector $C = [c_1, c_2, ..., c_K]$, where the c_i represents the (average) likelihood for $\tilde{X}$ to belong to the i^{th} cluster. Then, we set a threshold $\tau \in [0, 1]$ and mark each cluster as normal or anomalous as follows:

$$c_i = \begin{cases} anomalous & if\ c_i \leq \tau \\ normal & otherwise \end{cases} \tag{2}$$

After identifying the anomalous clusters, the anomaly detection can be performed by computing the soft clustering for a given test datapoint and marking it as an anomaly with the corresponding uncertainty measured through the likelihood $p(x_i|z_i = k)$.

VI. Experimental Results

In this section, we present the results obtained from testing the proposed anomaly detection technique in the industrial scenario considered. We demonstrate the feasibility of our approach by its performance in detecting the occurrence of anomalies and we also show the advantages of our semi-supervised approach over a supervised one.

Fig. 5 depicts the split of the dataset into training (80%) and test (20%) datasets of BGM. Now, from the training dataset, we extracted 10% of the normal labeled samples to create the normal dataset. Since the system operating conditions have been manually injected during the dataset construction phase (described in Subsection V-A), the three obtained datasets are provided with the corresponding labels. However, to emulate the lack of labeled data (a common problem in the industrial setting), after the normal dataset extraction, we dropped the labels of the training dataset, maintaining them in the test only for performance comparison purposes.

The normal dataset was used to train BGM in a supervised way. In the semi-supervised approach, the normal dataset was used for identifying the cluster associated with the normal operating condition of the system using the methodology in Subsection V-B. On the other hand, the training dataset containing normal and anomalous samples, was used to train the BGM in a semi-supervised way (considering only the labels related to the normal dataset). Finally, the same test dataset was used to test the performance of both the approaches.

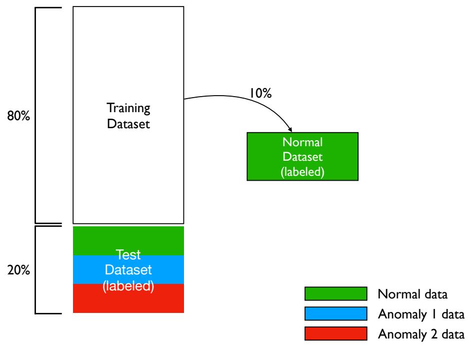


Fig. 5: Datasets organization for training and testing of BGM using supervised and semi-supervised approaches.

We first analyze the supervised approach where we consider a totally labeled dataset (i.e., normal dataset) for training the algorithm. This is the baseline to compare with our proposed semi-supervised approach.

In the supervised approach, the base methodology involves in training the BGM only on data related to the normal working condition of the system with the aim to learn its probability distribution. Since we are training only on the normal dataset, we used one cluster to fit the data. To visualize the cluster found by the BGM, we applied Principal Component Analysis (PCA) to reduce the data into a two-dimensional space. Fig. 6 depicts the contour plot from testing the BGM on the test dataset and considering the first two principal components (PC1 and PC2). Here the contour lines delineate areas with the same Negative Log Likelihood (NLL). Due to the strictly decreasing trend of the NLL, the higher the likelihood (i.e, probability) of a datapoint, the lower is the NLL. Thus, the points falling on the dark purple areas correspond to the points for which the BGM has a low uncertainty. On the other hand, those falling on green and yellow areas correspond to the datapoints for which the algorithm has a higher amount of uncertainty. Here the green, blue, and red points are referred to the normal, Anomaly 1, and Anomaly 2 respectively.

Normal datapoints are well concentrated around the mean of the cluster (marked as a black "X"), such that the ones which are far away from it are likely to be considered as anomalies. The soft clustering performed well; looking at the figure, even if three green datapoints fell outside the dark purple area, the BGM algorithm still has a good level of confidence on them, marking them as "normal." The distinction between normal and anomalous datapoints can be performed by following the BGM analysis. As mentioned, the model assumes normal distribution of data and finds the best set of parameters to fit in (i.e., maximize the probability). Exploiting that the probability maximization in a normal distribution is equivalent to approaching the data to the mean of the distribution, we can identify anomalies inside the cluster. To compute the threshold for distinguishing between a normal condition and an anomalous one, we computed the distances for each training datapoint and the mean of the distribution. Thus, the distance threshold is computed as: $threshold = \widetilde{distances} + \sigma_{distances}$ where $\widetilde{distances}$ is the median of the distances previously computed and $\sigma_{distances}$ is its standard deviation.

To measure the performance of the algorithm, we used hard clustering by setting the above threshold, and marked as an anomaly each datapoint beyond the threshold. The obtained results are very good with an average on 100 different training or test split of 98% for accuracy. Moreover, to better analyze the results, we computed the confusion matrix on the test set to capture the performance of our machine learning algorithm through the extraction *precision*, *recall*, and F_1-*score*.

Considering the same 100 different training/test split used for the accuracy computation, the algorithm scored on average 97% for the precision, 100% for the recall, and 98.4% for the F_1-*score* (see Table I).

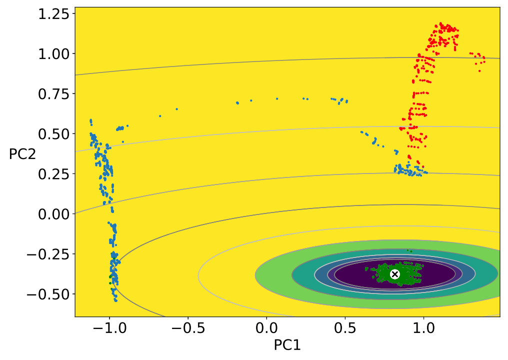


Fig. 6: Contour plot of BGM performed on the test dataset.

For the semi-supervised approach, the BGM is trained with the above defined training dataset whose samples are referred to as normal or anomalous operating condition, and for which the corresponding labels are not known. In this sense, the very first step is to choose the correct number of clusters to fit the data in an unsupervised way (i.e., without the use of labels). For traditional algorithms like K-Means, it is possible to use metrics like the silhouette score to derive the optimal number of clusters. However, this can not be done with BGM because it assumes clusters with different sizes and shapes which make these metrics unreliable. To address this problem, we exploit the probabilistic nature of the algorithm by choosing a fairly large number of clusters and then looking at the multinomial distribution of the weights introduced in Section V.

Fig. 7 depicts the cluster weights distribution inferred from data by the BGM after the training process. Despite an initial selection of 10 clusters at the end of the training, only 3 of them reached a probability higher than 0.1 implying that the remaining ones are unnecessary for the clustering process. For this reason, we selected $K = 3$ clusters in our semi-supervised approach, which is a coherent choice if we consider that the system worked on three different operating conditions as we mentioned in Subsection V-A.

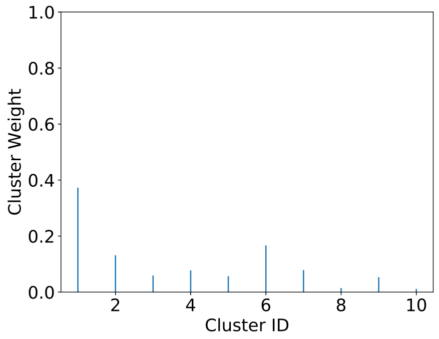


Fig. 7: Cluster weights multi-nomial distribution after BGM training.

Fig. 8 shows the semi-supervised BGM contour plot of the PCA training dataset for which we again considered the first two principal components; and the dashed red curves represent the decision boundaries for the clusters. The BGM identified the 3 clusters, however, it was unable to establish which were related to the normal or anomalous operating conditions. In order to identify the cluster related to the normal operating condition, we used the labeled normal dataset (same as for supervised BGM training) and then applied the methodology in Subsection V-B. Fig. 9 depicts the contour plot of the same test dataset for the supervised approach after applying the methodology, where the green datapoints are labeled as normal samples for cluster identification.

As expected, the majority of the normal datapoints (in green color) fell in the dark purple area of the same cluster on the bottom right, implying very high BGM confidence in belonging to the cluster. This allowed us to identify the bottom right cluster as the one associated with the normal working condition of our system, and mark the datapoints falling outside as anomalies. Moreover, looking at the rest of the plot, only a small portion of the datatpoints falls in the green and yellow areas, which implies that the algorithm has most of the times a low level of uncertainty with respect to data clustering of these points. In terms of soft clustering, in this case also the BGM performed very well. Specifically, a small number of blue and green datapoints fell outside the dark purple area of their corresponding clusters. Nevertheless, the BGM has a high level of confidence with respect to these two group of datapoints which enable to cluster them correctly.

In general, the semi-supervised BGM outperformed the supervised one, This is due to the fact that the former approach benefits from the use of a larger training set containing both normal and anomalous data, which help the algorithm to better define the cluster decision boundaries, thus improving the performance of anomaly detection.

TABLE I: Accuracy, precision, recall and F_1 scores computed for each approach.

Performance indices		
$Index$	$Supervised$	$Semi-supervised$
$Accuracy$	98%	99.8%
$Precision$	97%	99.7%
$Recall$	100%	100%
F_1	98.5%	99.8%

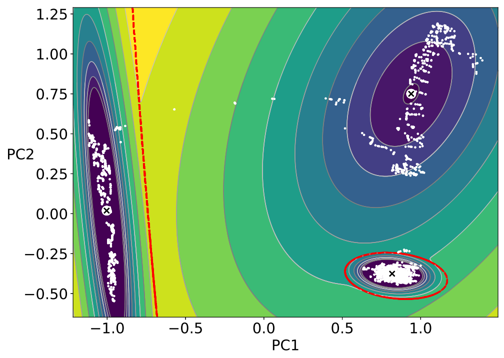


Fig. 8: Contour plot of BGM performed on the training set.

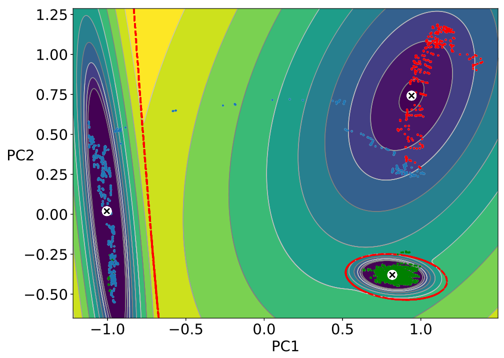


Fig. 9: Contour plot of the BGM performed on the test set after identification of the cluster of the labeled data (green).

Additionally, we performed the hard clustering, computed the confusion matrix, and measured the performance of our algorithm by extracting the *precision*, *recall*, and F_1-*score* indices. Table I reports these indices considering a threshold of $\tau = 0.2$ that we set empirically keeping fixed the the train and test sets, varying τ from 0 to 1.0 (with a step of 0.01), and measuring the values of precision and recall obtained by the proposed BGM algorithm (as showed in Fig. 10). The obtained results are very good with an average of 99.8% for the accuracy, 99.7% for the precision, 100% for the recall, and 99.8% for the F_1-*score*, thus demonstrating the effectiveness of our anomaly detection algorithm.

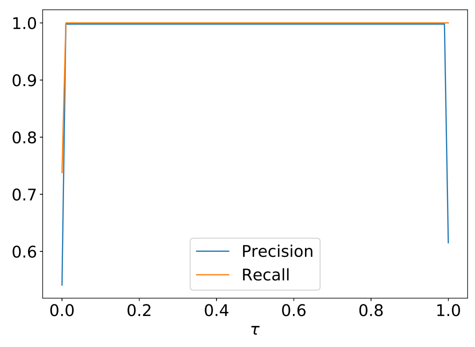


Fig. 10: Precision and recall measurements when varying τ for the anomaly detection threshold setup.

To further demonstrate the benefits of BGM semi-supervised approach, we compared its performance with the supervised one and the One Class Support Vector Machines (OCSVM). OCSVM have been tuned using a radial basis function kernel and gamma parameter to "auto", a special value supported by the Scikit-learn framework for automatically setting the best value. Fig. 11 depicts a comparison in terms of accuracy between the three models when varying the percentage of normal labeled samples in training dataset. The OCSVM maintains an accuracy of about 81.5% which is worst compared to other models. We also observe that the BGM fully supervised approach reaches an accuracy which is slightly less than the one obtained with the semi-supervised model. However, the performance tends to decrease as the number of labeled samples decreases. On the other hand, the proposed BGM semi-supervised approach exhibits a higher accuracy which remains almost constant even for small percentages of labeled samples (e.g., 20% and 10%), demonstrating the effectiveness of our approach for an industrial scenario.

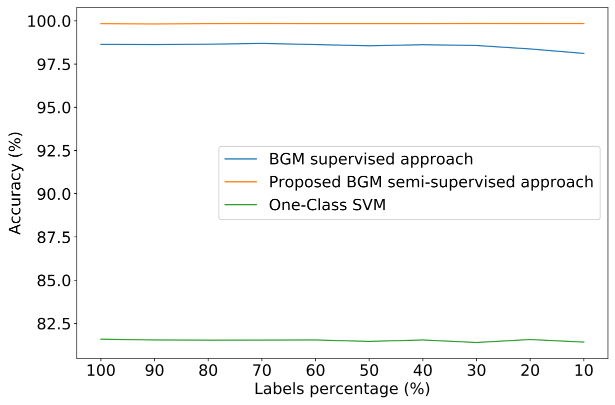


Fig. 11: Accuracy comparison between BGM supervised approach, the proposed semi-supervised one, and One-Class SVM algorithm with varying percentage of labeled samples.

VII. Conclusions

In this paper, we presented an anomaly detection technique for the diagnosis of a scale replica plant. Precisely, we extended the functionalities of S4T architecture to suit it for industrial applications, and implemented on top of it a BGM based semi-supervised anomaly detection algorithm to assess the working condition of the plant. The results of anomaly detection algorithm are very good. It was able to correctly detect the system working conditions with a 99.8% of accuracy and a high level of precision, recall, and F_1-*score*.

As part of future work, we plan to improve the performance of the proposed technique in order to detect a higher number of anomalies. This can be accomplished with the help of additional sensors (e.g., a weight sensor on the cart) such that the telemetry can provide more information about the system working conditions, and the anomaly detection algorithm will be enhanced to identify not only the presence of an anomaly, but also which type of anomaly is occurring.

Acknowledgments

The work of S. K. Das was supported by NSF grants SaTC-2126619, CNS-2008878, CNS-1818942, and DGE-1433659.

References

[1] W. Peng, Z. Ye, and N. Chen, "Bayesian deep-learning-based health prognostics toward prognostics uncertainty," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 3, pp. 2283–2293, March 2020.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions," *Future Generation Computer Systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[3] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, Opportunities, and Directions," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 11, pp. 4724–4734, Nov 2018.

[4] P. Bellavista, J. Berrocal, A. Corradi, S. K. Das, L. Foschini, and A. Zanni, "A Survey on Fog Computing for the Internet of Things," *Pervasive and Mobile Computing*, vol. 52, pp. 71 – 99, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1574119218301111

[5] R. Y. Zhong, X. Xu, E. Klotz, and S. T. Newman, "Intelligent Manufacturing in the Context of Industry 4.0: A Review," *Engineering*, vol. 3, no. 5, pp. 616 – 630, 2017.

[6] F. Longo, D. Bruneo, S. Distefano, G. Merlino, and A. Puliafito, "Stack4Things: A Sensing-and-Actuation-as-a-Service Framework for IoT and Cloud Integration," *Annals of Telecommunications*, vol. 72, no. 1, pp. 53–70, Feb 2017.

[7] P. Ferrari, S. Rinaldi, E. Sisinni, F. Colombo, F. Ghelfi, D. Maffei, and M. Malara, "Performance Evaluation of Full-cloud and Edge-cloud Architectures for Industrial IoT Anomaly Detection based on Deep Learning," in *2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4.0 IoT)*, June 2019, pp. 420–425.

[8] B. Wang, Z. Li, Z. Dai, N. Lawrence, and X. Yan, "Data-driven mode identification and unsupervised fault detection for nonlinear multimode processes," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 3651–3661, June 2020.

[9] T. Addabbo, A. Fort, M. Mugnaini, L. Parri, S. Parrino, A. Pozzebon, and V. Vignoli, "An IoT Framework for the Pervasive Monitoring of Chemical Emissions in Industrial Plants," in *Workshop on Metrology for Industry 4.0 and IoT*, April 2018, pp. 269–273.

[10] W. Ren, T. Yu, T. Yardley, and K. Nahrstedt, "Captar: Causal-polytree-based anomaly reasoning for scada networks," in *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, Oct 2019, pp. 1–7.

[11] D. Bruneo and F. D. Vita, "On the Use of LSTM Networks for Predictive Maintenance in Smart Industries," in *IEEE International Conference on Smart Computing (SMARTCOMP)*, June 2019, pp. 241–248.

[12] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, Jul. 2009. [Online]. Available: https://doi.org/10.1145/1541880.1541882

[13] S. Bianchi, R. Paggi, G. L. Mariotti, and F. Leccese, "Why and When Must the Preventive Maintenance be Performed?" in *2014 IEEE Metrology for Aerospace (MetroAeroSpace)*, May 2014, pp. 221–226.