Revista Facultad de Ingeniería

Title: Anomaly classification in IIoT edge devices

Authors: Diana Patricia Tobón-Vallejo[1] https://orcid.org/0000-0003-4659-7693, Danny Alexandro Múnera-Ramírez[2] https://orcid.org/0000-0003-0762-0571 and Martha Lucía Rodríguez-López[2*] https://orcid.org/0000-0001-9718-7673

[1]Departamento de Ingeniería Electrónica, Universidad de Antioquia, Calle 67 #53-108. Medellín, Antioquia.
[2]Departamento de Ingeniería de Sistemas, Universidad de Antioquia, Calle 67 #53-108. Medellín, Antioquia.

Corresponding Author: Martha Lucía Rodríguez-López
E-mail: mlucia.rodriguez@udea.edu.co

DOI: **10.17533/udea.redin.20250368**

To appear in: *Revista Facultad de Ingeniería Universidad de Antioquia*

Received: March 19, 2024
Accepted: March 17, 2025
Available Online: March 17, 2025

Revista Facultad de Ingeniería, March, 2025

Anomaly Classification in IIoT Edge Devices

Clasificación de anomalías en dispositivos del borde de IIoT

Authors: Double-blind review

KEYWORDS:
Anomaly detection, anomaly classification, neural networks, Industrial Internet of Things.

Detección de anomalías, clasificación de anomalías, redes neuronales, Internet Industrial de las Cosas.

ABSTRACT: An early Industrial Internet of Things (IIoT) Anomaly Detection reduces maintenance costs, minimizes machine downtime, increases safety, and improves product quality. A multi-class classifier that detects events, failures, or attacks is much more efficient than a simple binary classifier, as it relieves a human operator of the task of identifying anomaly causes, thereby avoiding wasted time that could compromise process performance and security. With these issues in mind, this paper aims to determine whether it can differentiate between a failure that generates a temperature increase in an IIoT device processor, a denial-of-service attack on an MQTT broker, and an event caused by an application executing on the IIoT edge device. Data used to perform the classification comes from a Raspberry Pi 3, specifically from its CPU (e.g., temperature, load, free memory, Wi-Fi sent and received packets). A k-nearest neighbors (KNN), random forest (RF), support vector machine (SVM), and Multilayer Perceptron (MLP) algorithms were trained. Considering metrics such as false positive rate, false negative rate, accuracy, F1-score, and execution time, we concluded that SVM and MLP were the best methods for the case study because of their accuracy (78.6 and 76.1, respectively) and low execution time (17.3ms and 0.35ms).

RESUMEN: Una detección temprana de anomalías en Internet Industrial de las Cosas (IIoT) reduce los costos de mantenimiento, minimiza el tiempo de inactividad de la máquina y aumenta la seguridad de la planta. Un clasificador multiclase que detecta eventos, fallas o ataques libera al operador humano de la responsabilidad de identificar la causa de la anomalía, evitando desperdicio de tiempo que podría comprometer el rendimiento y la seguridad del proceso. Con estas cuestiones en mente, este artículo tiene como objetivo determinar si es posible diferenciar entre una falla de temperatura en un dispositivo IIoT, un ataque de negación de servicio a un broker MQTT y un evento causado por una aplicación que se ejecuta en el dispositivo de borde IIoT. Los datos utilizados para realizar la clasificación provienen de una Raspberry Pi 3, concretamente, datos de su CPU (temperatura, carga, memoria libre, paquetes Wi-Fi enviados y recibidos). Se entrenaron algoritmos del tipo k vecinos más cercanos (KNN), bosque aleatorio (RF), máquina de soporte vectorial (SVM) y un perceptrón multicapa (MLP). Teniendo en cuenta métricas como tasa de falsos positivos, tasa de falsos negativos, precisión, F1-score y el tiempo de ejecución, llegamos a la conclusión de que SVM y MLP fueron los mejores métodos para el caso de estudio, debido a la precisión (78,6 y 76,1, respectivamente) y el bajo tiempo de ejecución (17,3 ms y 0,35 ms).

1. Introduction

The Industrial Internet of Things (IIoT) presents a realm of wireless connectivity, enabling data collection and processing within interconnected industrial settings. Coupled with other advances characteristic of the fourth industrial revolution (Industry 4.0), it furnishes manufacturing environments that are real-time, secure, and autonomous [1–3]. Enhancing production efficacy requires reliable computing and communication tools to address security vulnerabilities arising from extensive subsystem interconnections [2, 4, 5]. Moreover, IIoT components leverage diverse technologies to collect various types of data, often lacking structure, plagued by noise, and exhibiting redundancy [6]. Security loopholes and data collection challenges pose significant hurdles to ensuring data integrity, potentially rendering IIoT-gathered data unfit for user-centric services [7]. Anomaly detection systems offer a means to identify subpar data quality.

In their study, Gosh et al. [8] characterize anomalies as observations or subsets that deviate significantly from the dataset's norm, classifying them as events, failures, or attacks. They define a "failure" as data from a malfunctioning sensor due to calibration issues or device faults [9]. Conversely, an "event" denotes a real-world occurrence altering monitored variables, such as natural phenomena impacting specific metrics [7]. Unauthorized access or intrusions pose substantial security threats across the system [10, 11], with a "malicious attack" encompassing actions compromising IIoT network nodes, thus jeopardizing

overall security [8, 11].

Early detection of anomalies within industrial processes proves pivotal in facilitating real-time decision-making, curtailing maintenance expenses, mitigating machinery downtime, enhancing safety, and elevating product quality [12, 13]. Pinpointing the origin of the anomaly (whether an event, failure, or attack) helps to select appropriate recovery measures to minimize abnormal behavior [3, 14]. This study sought to ascertain the feasibility of discerning between a temperature-induced failure in an IIoT device's processor, a denial-of-service attack on the MQTT broker, and an event triggered by an application executing on the IIoT edge device. We investigated whether this differentiation could be achieved using internal CPU data such as temperature, load, available memory, and Wi-Fi transmission metrics, employing machine learning techniques for real-time anomaly detection. This paper compares multiple machine learning approaches based on several metrics for classifying anomalies in real time on an edge device.

The remainder of this paper is organized as follows: Section 2 describes the machine learning algorithms compared in this work. Section 3 mentions some related works. Section 4 describes the testbed, dataset, and machine learning techniques used in this report. Section 5 shows the results of applying ML methods to classify anomalies. Finally, we highlighted the open challenges (6) for future research and conclusion (7).

2. Background

This section describes the algorithms that classify anomalies as events, failures, and attacks. These methods are described as follows.

2.1 K-Nearest Neighbor

The K-nearest neighbor (KNN) is a widely used nonparametric pattern classification method [15]. Despite its simplicity, KNN has been successful in many classifications and regression problems, including handwritten digits and satellite image scenes [16]. It is often successful in classification situations where the decision boundary is irregular since it is a non-parametric method [17]. Most KNN algorithms are developed for classification problems with balanced training sample sets [17]. A KNN Euclidean is the nearest neighbor technique based on the Euclidean distance between values; the nearest nodes have the nearest captured values [18]. A KNN classifier consists of the annotated training samples in the feature space for all the classes [16].

2.2 Support Vector Machine

Support vector machines are supervised learning methods for classification, regression, and outlier detection. Support Vector Machine (SVM) method is an algorithmic application of statistical learning theory [19]. SVM is a strategy for classifying linear and nonlinear information [20]. SVM creates an N-dimensional hyperplane, ideally separating the information into different categories [21]. SVM works by maximizing the edge to achieve the most effective performance execution in terms of classification [10, 20].

Specifically, this work uses the SVC algorithm (C-Support Vector Classification) from the Scikit Learn library [22] to classify anomalies. SVM is effective in high dimensional spaces and uses a subset of training points in the decision function (called support vectors), so it is also memory efficient. However, the fit time scales at least quadratically with the number of samples, and SVM does not directly provide probability estimates; these are calculated using an expensive fold cross-validation [10, 22].

2.3 Random Forest

The Random Forest (RF) model is an ensemble learning process for classification and regression tasks [23]. The RF method can be grouped under the category of ensemble models. The building block of a Random Forest is the Decision Tree [24]. A decision tree is a tree-like structure in which each internal node represents a "split" based on an attribute, and each leaf node represents a prediction result of classification or regression [23, 25]. Random Forest gathers a group (or ensemble) of decision trees and uses their combined predictive capabilities to obtain relatively strong predictive performance [24]. Since decision trees are sensitive to class imbalance, each tree will be biased in the same direction and magnitude by class imbalance [23].

Random forest uses multiple classification trees. A tree classification algorithm requires a different bootstrap sample from the original data. When the forest formation is completed, each tree casts a vote for the object's class. The forest selects the class with the most votes for the object [10]. RF model is based on decision trees and the bootstrap aggregating mechanism to avoid the overfitting problem of complex decision trees [23].

2.4 Multilayer Perceptron

A Multilayer perceptron (MLP) is a feedforward neural network with a gradient descent feature. MLP is composed of multiple neurons and uses back

propagation supervised learning [19]. Except for the input neurons, every neuron has an activation function, continuously reducing the gradient to achieve the convergence [26, 27]. An MLP has the same structure of a single layer, adding one or more hidden layers with all the nodes connecting each other between layers [19, 28]. The network trains itself with an algorithm called backpropagation. This supervised learning algorithm first computes outputs using a sigmoid function and then propagates the errors backward. Each unit receives the amount of error generated, and the weights are adjusted [29]. Artificial neural networks fire a potential action if the cumulative input of the signals exceeds a threshold. The output is calculated with a transfer function through the sum of every input multiplied by its weights. The weights are computed during training [29, 30].

3. Related Works

Machine learning (ML) approaches are effective techniques for data analysis and hidden patterns search [19]. ML techniques are even used for anomaly detection systems in incomplete and unbalanced data [20, 29]. This work compares several ML methods to classify anomalies on IIoT edge devices. The literature revision found that ensemble regression performs better since it combines several single methods to improve the accuracy and stability of a single regressor [31]. An ensemble regression method by minimizing total least square loss in multiple reproducing kernel Hilbert spaces is proposed in [31] and evaluated in several classification datasets.

Similarly, [32] suggests a kernel ridge regression based on intuitionistic fuzzy membership from binary classification without giving the same significance to the critical and unimportant samples. The report in [33] proposes a non-overlapped risk-based bagged ensemble model to handle imbalance and noise contained in credit card transactions. Since ensemble models can use any classification method, [34] proposes a homogeneous bagging ensemble of 3-layer fully connected networks for anomaly detection in surveillance camera video.

Furthermore, a comprehensive study of anomaly detection of malware-based k-nearest neighbor algorithms is presented in [17]. In contrast, [16] designs a KNN-based classifier for sea-surface small target detection methods in high-resolution maritime ubiquitous radars. Additionally, [18] describes how to detect abnormalities from spatial distribution data using KNN technique and Euclidean distance in a wireless sensor network. Some reports combine two or more methods to improve performance. The report in [23] designs an anomaly detection method based on an autoencoder and random forest for solving the credit card fraud detection problem; first, the autoencoder reduces the dimensionality of data, and then the random forest classifies data as anomalous or normal.

The report in [20] proposes a weighted hybrid model utilizing a Support Vector Machine and Naïve Bayes for anomaly discovery; they show how to merge the prediction from multiple systems to improve the generalization over a single estimator. While [10] proposed a two-layered anomaly detection model with Random Forest and Support Vector Machine as classifiers for intrusion detection on a computer network, authors in [24] implemented various classification algorithms—such as ridge classification, decision trees, and random forests—to predict the probability of employee attrition in a large company. In addition, a comparison between multilayer perceptron neural networks and support vector machines on the heart diseases dataset is conducted in [19], a medical field where decisions deal with patient outcomes, highlighting the importance of high accuracy in data mining for medical decision-making.

Authors in [30] propose a multilayer perceptron with optimal stochastic gradient descent to extract meaningful conclusions from medical datasets. Other work using MLP is described in [29]. In this study, a multilayer perceptron approach is used to monitor the packet headers of the network traffic rather than the attached data, in order to keep up with the growing bandwidth of networks and to maintain the privacy of the users on their Intrusion Detection Systems. Moreover, the authors in [28] compare the different performances of multilayer perceptron algorithms for solving the electrical impedance tomography inverse problem. The work in [26] proposes an anomaly detection based on multilayer perceptron for reducing overhead in Software Defined Networks. Finally, [27] evaluates some multilayer perceptron networks based on Easy-Ensemble to learn the imbalance transient stability data of power systems after using principal component analysis to reduce the dimensionality of the data.

A previous work carried out by the authors [13] showed through an analysis of variance (ANOVA) that a temperature fault, a denial-of-service (DoS) attack on the MQTT broker, and the normal operation of a testbed similar to the one used in this work, were statistically different. It was also found that, while current and frequency differentiated between normal and attack, temperature allowed for the differentiation between failure and normal operation of the system.

4. Research strategy

4.1 Methodology

The research goal is to determine whether it is possible to differentiate between a temperature failure, a denial-of-service (DoS) attack, an event caused by an application running on an IIoT device, and normal operation of an Industrial Internet of Things edge device, using only internal CPU variables and simple machine learning methods. Figure 1 shows this work summary. A testbed was implemented, with an IIoT system monitoring an AC three-phase motor. Temperature failures, DoS attacks in MQTT Broker, and a non-dangerous event were simulated in the edge device of the testbed. Data were collected daily for a week to reduce the errors caused by external factors, such as indoor temperature changes.

Collected data passed by a subsampling process to delete randomly as many samples as was necessary for all classes to have the same amount of data. A balanced data set was used to optimize hyperparameters on four non-complex machine learning algorithms. The optimization used Python and libraries such as ScikitLearn and TensorFlow running on Google Colaboratory (Colab).

The multilayer perceptron model was trained with an optimized hyperparameter on Google Colab and converted by TFLite to a model suitable for running on a constrained device such as Raspberry Pi. The rest of the models were trained directly on the edge device. Each model ran on edge devices to process internal CPU variables and predict events (E), failures (F), attacks (A), or normal (N) operations. Predicted and actual states (E, F, A, N) are saved for later metrics analysis.

4.2 Testbed

An IIoT system was selected to simulate events, failures, and attacks in its edge device. This system consists of several low-cost sensors that monitor a three-phase motor. Each sensor sends motor current, temperature, sound, speed, and vibration data to a Raspberry Pi 3 edge device, through a low-cost ESP32 development system using MQTT. Motor data is processed on Raspberry to activate alarms when a value exceeds a normal range. Then, processed information is sent to a cloud application that stores and displays statistics on a website.

This work tried to differentiate between a high-temperature failure, a DoS attack, and an event triggered by a harmless application. The DoS attack was controlled by a computer running Kali Linux in a local network. A fan regulated the edge device processor temperature. The harmless application calculated the average time to perform 100000 floating point divisions. Raspberry collected internal data such as temperature, CPU load, sent and received packets through Wi-Fi, and free memory. This Internal data was obtained through RPi-Monitor [35] application. That information was written in a CSV file by a Python script. It was not possible to determine internal data accuracy. Figure 2 shows the Testbed used to collect the data set.

4.3 Dataset

The selected internal CPU variables for classifying events, failures, attacks, and normal operations include temperature, average last-minute load, free memory, and sent and received Wi-Fi packets. The last three variables were transformed to use the delta between two consecutive values, avoiding upward or downward trends unrelated to the anomaly occurrence. Several data collection sessions lasting one hour were taken during one week. In each, normal operation system data was collected along with anomaly data (event, failure, attack). Selected data for the training phase were combined in a single data set, sub-sampling was also performed to provide a balanced data set, and scaling was applied between 0 and 1. Separate anomalies are explained as follows,

- **Failure.** It simulated a temperature failure by turning off the edge device's fan. The 24-watt AC fan was placed 20 centimeters above the Raspberry.
- **Attack.** It ran a Python script from a local computer that subscribes 1024 clients to the "#" topic and publishes a short message every second to make a DoS attack on the MQTT Mosquitto broker running on the same edge device.
- **Event.** It simulated an event by running a Python script calculating the average time needed to make 100,000 divisions. The script did not interfere with the normal operation of the IIoT system but consumed the edge device's CPU resources.

Figure 3 shows labels of data testing with anomalies (a), temperature (b), and last-minute-load CPU (c). Since free memory and sent and received Wi-Fi data have a cumulative behavior, this work does not directly use these variables but the delta between current and previous values, as shown in graphs (d), (e), and (f).

4.4 Machine learning methods

The dataset gathers 13117 samples for each class, such as event, failure, attack, and normal. It is worth

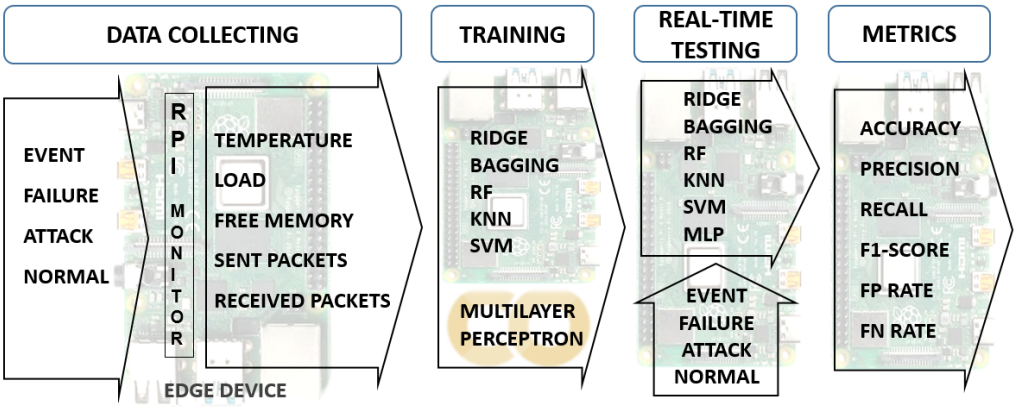

Figure 1 Summary of the experimental setup for anomaly classification.

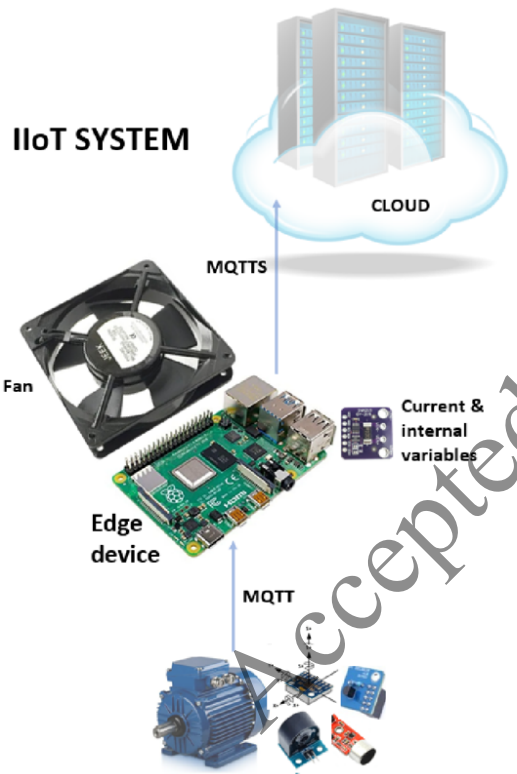

Figure 2 Testbed used for data collection.

noting that a balanced set is rare in anomaly detection problems. This dataset had been specially collected for machine learning algorithms that require a balanced dataset. The testing dataset was collected separately and did not need to be balanced. Then, some ML algorithms were selected to classify data as normal, event, failure, or attack: random forest, k-nearest neighbor, support vector machine, and multilayer perceptron. The first three algorithms were trained on Raspberry Pi with the ScikitLearn library. In contrast, MLP was trained with the TensorFlow library on Google Collaboratory and then converted to a TFLite model. All the algorithms were evaluated on Raspberry Pi in real time; the results are described in the next section.

4.5 Metrics

To evaluate which methods detected anomalies better, some metrics were compared, including file size, execution time, accuracy, precision, recall, and f1-score. Since an IIoT system monitors an industrial process, it is also essential to evaluate false positive rates that generate alarms and move resources to solve an unreal anomaly and false negative rates that ignore abnormal situations.

A false positive rate (FP) is normal data predicted as some anomaly. In FP, there are two categories: one is unrelated to actual anomalies, triggering unnecessary alarms that waste resources and exhaust staff, potentially leading them to ignore future alerts if FPs occur too frequently. The other type arises as a residual effect of a past anomaly, where certain variables—such as temperature or load—take time to return to normal ranges, causing continued false alerts even after the anomaly itself has ended (Figure 4).

A false negative rate (FN) is abnormal data the classifier model evaluates as a normal value. FN could make a system or human operators ignore potentially hazardous situations. Here, there are two categories; one occurs immediately after the anomaly begins, and it could be considered a system delay to detect an anomaly; in the other category, FN avoids alerting human operators for a long time (Figure 4). Misclassified rate anomalies are events, failures, and attacks confused with each other. This can result in deploying the wrong personnel, such as alerting maintenance staff to address a physical failure during a cyberattack.

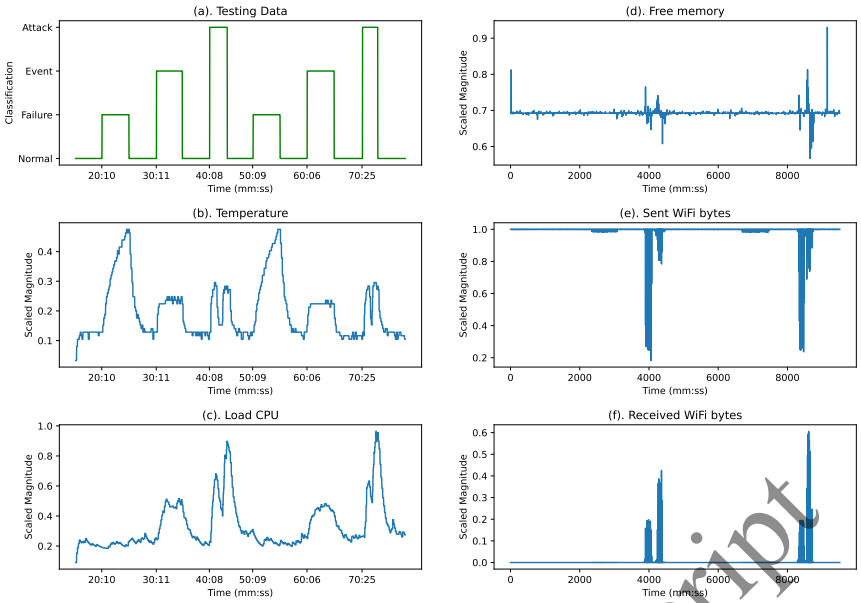

Figure 3 Selected variables to classify anomalies in an IIoT edge device.

5. Results and Discussion

During a real-time evaluation of classification methods, predictions and true labels were stored for subsequent metrics calculation and their corresponding analysis. Table 1 shows the disk space occupied, accuracy, execution time, and training time for each method; since these are multi-class classifiers that are evaluated on an unbalanced dataset, it also shows averaged and weighted precision, recall, and F1-score. KNN and MLP occupy the smaller disk space. Since Raspberry Pi is a device with disk and memory restrictions, file size is a parameter to be considered. However, it is not the most critical metric as long as it stays within the acceptable range for the edge device.

For the case study, execution time (runtime) is an important metric to justify the selection of the real-time anomaly classification method. In this work, the authors consider that "real-time" implies that the anomaly is classified before a second. According to Table 1, the methods with the shortest execution time are MLP (0.35ms), KNN (5.71ms), and SVM (17.28ms). Suppose the reaction time of a human operator is considered in that case; it could be stated that all the methods evaluated allow real-time classification since their execution times are below one second.

As for weighted average precision, Table 1 shows that SVM (82.22) and MLP (82.45) show a higher ratio between true anomalies and all samples labeled as anomalies. Likewise, SVM (78.59) and MLP (76.06) have the highest weighted average recall, i.e., the highest proportion of true anomalies detected. Finally, the methods with the highest proportion of correctly classified samples (accuracy) are MLP (76.09) and SVM (78.68).

Table 2 shows other metrics to evaluate the performance of an anomaly classifier in IIoT. The rate of normal data labeled as an anomaly (false positive, FP) dedicates resources to solving an abnormal situation that does not exist. FP rate for SVM is 0.267 and 0.296 for MLP. Meanwhile, the false negative (FN) rate quantifies undetected anomalies. FN for SVM is 0.2166 and 0.1803 for MLP. The rate of misclassified anomalies, events, failures, and attacks confuse each other and can delay anomaly resolution. For example, the maintenance department may try to resolve a fault that could be a cyberattack that needs to be addressed by the IT department.

Figure 4 confirms that SVM and MLP perform better, where both methods only generate FP after the anomaly cause has finished. The remaining effect that causes the classifier to continue detecting the anomaly could be because variables such as temperature or CPU load require some time to return to normal values. The above is an *apriori* conclusion from the superposition of these variables' plots and the anomaly labels' plot in Figure (c). What cannot be determined from the data extracted from Figure 4 is which range is considered normal for each variable in each classifier model. As

Table 1 Results of each method.

Method	Library	Sized (kB)	Training time (s)	Exec time (ms)	Wighted Precision	Wighted Recall	Wighted F1-score	Accuracy
SVM	SkLearn	2596.86	737.12	17.28	82.22	78.59	79.19	78.62
MLP	TensorFlow	8.19	56.69	0.35	82.45	76.06	77.42	76.09
RF	SkLearn	233209.85	359.79	678.2	78.84	74.28	75.44	74.29
KNN	SkLearn	4.78	0.69	5.71	77.72	71.65	73.05	71.66

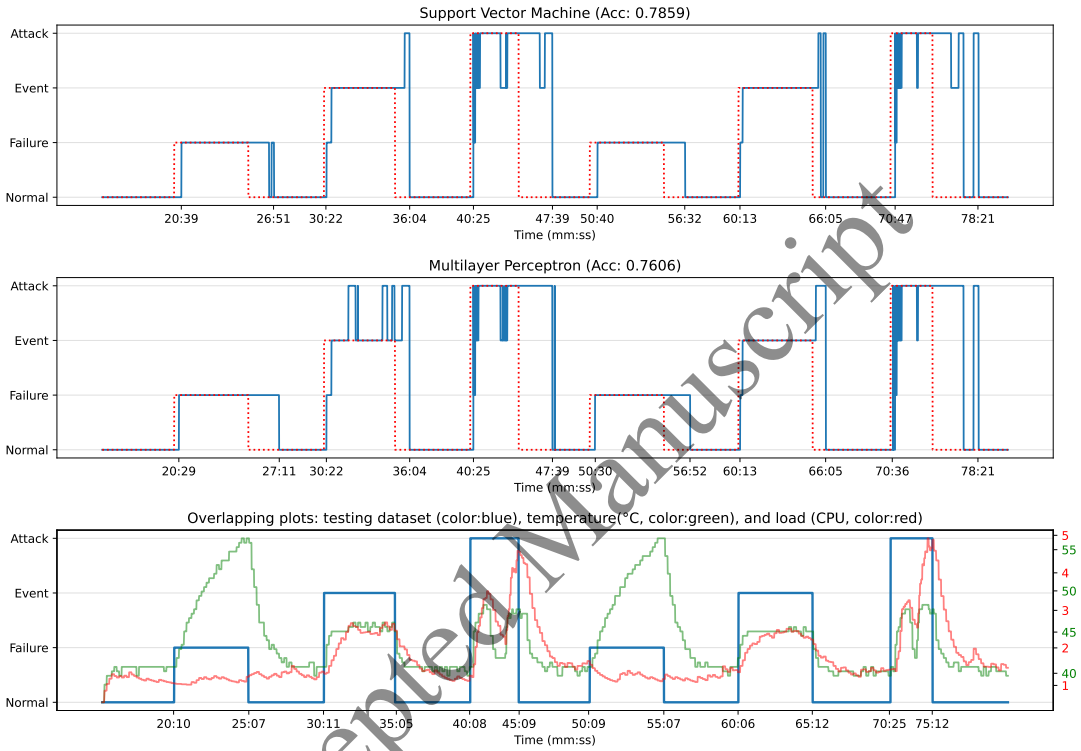

Figure 4 Performance of SVM and MLP classifiers.

for FN for SVM and MLP, it occurred immediately after the anomaly began; this could be interpreted as a delay in the classifier's ability to perform detection and could be associated with the temperature change rate and CPU load.

From the time indicated in the three plots in Figure 4 we could say that these classifiers take about 30 seconds to detect a failure, less than 10 seconds to detect the execution of a script considered an abnormal event, and less than 25 seconds to alert to the occurrence of a cyberattack in the case study. The delay before the classifier detects that an anomaly has ended and the system has returned to its normal state is approximately two minutes after a failure, one minute after an event, and 2.5 minutes after a DoS attack.

The variable free memory and packets sent and received via Wi-Fi do not have a continuous change during abnormal or normal operation. This aspect is illustrated in Figure 5, where on the left are the labels of the anomalies (below) and the deltas of the received packets (up). Zooming in the graph during the occurrence of an attack (right), the discrete character of the variable is better observed. Since not all samples have a high Wi-Fi received packet value while the attack is carried out, it could be argued that it is the cause of the confusion between events and attacks.

This work represents an advance in the research topic presented by the authors in their previous work [13]. In the work discussed in this report, machine learning models were tested to facilitate the classification of the event, failure, and attack described. However, using time series-based models could improve the results in

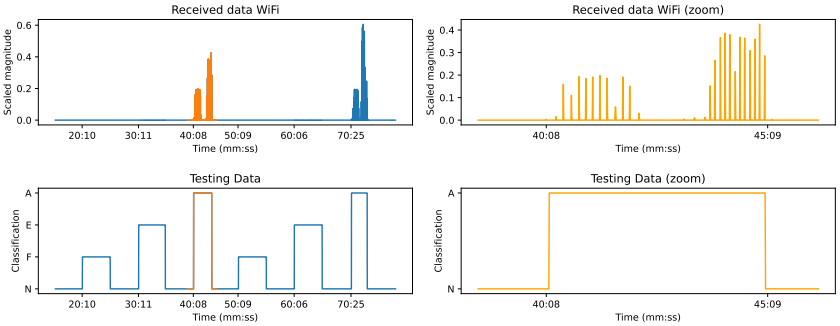

Figure 5 The plot on the top left shows that the data received over Wi-Fi is not a continuous variable. A zoom (top right) around minute 40 better illustrates this idea.

Table 2 Confusion matrix.

		PREDICTED				
	Class	A	E	F	N	Total
ACTUAL	A	SVM=69.12% MLP=71.23%	SVM=18.75% MLP=16.86%	SVM=3.47% MLP=3.47%	SVM=8.64% MLP=8.42%	949
	E	SVM=0% MLP=10.09%	SVM=92% MLP=81.96%	SVM=5.05% MLP=5.04%	SVM=2.95% MLP=2.88%	1525
	F	SVM=0% MLP=0%	SVM=0% MLP=0%	SVM=89.92% MLP=93.26%	SVM=10.07% MLP=6.73%	1558
	N	SVM=12.85% MLP=16.73%	SVM=5.34% MLP=1.96%	SVM=8.51% MLP=10.92%	SVM=73.28% MLP=70.36%	5484

future works, since these models allow the detection of spatial and temporal correlations between samples.

6. Open Challenges

Some of the methods compared, such as SVM and MLP, showed that they could detect all anomalies, and the FP and FN rates could be partly explained by the physical nature of the process, reducing the waste of resources attending to false anomalies. Some challenges are identified and described as follows,

- More complex classification models with time series could reduce detection time and persistence time after the anomaly's cause has ceased; they could also reduce the confusion between events and attacks presented in this case study.
- Detecting and classifying several anomalies coinciding is another challenge not addressed in this paper.
- More complex machine learning models may require devices with more storage and processing capacity than is available on an edge device. Multiple IIoT devices, or a combination of cloud and edge, could run the models, considering latency, security, and processing capacity issues.
- Another open research topic is real-time classification, or at least detecting anomalies not included in the training set.
- A single-edge device was used in this case study. A more complex IIoT system may require models combining data from other devices to make their own real-time predictions, using the edge device's limited processing and storage resources.
- Optimizing methods for working with time series can exploit the spatial and temporal correlation between variables. Finding algorithms robust enough to handle time series and yet simple enough to be implemented on edge devices and classify anomalies in real-time is a challenge.
- Another open challenge is to test these machine learning models on IoT devices with greater storage and processing constraints, considering security, latency, and processing requirements.

7. Conclusion

In this work, we found that it is possible to run machine learning algorithms on an edge device to differentiate a temperature fault, a DoS attack on the MQTT broker, and a harmless script execution event using

internal CPU data. Methods with low complexity, such as SVM and MLP, showed a good performance in classifying anomalies, with FP just after a real anomaly facilitating the interpretation of the FP by the detection system or the human operator, avoiding the waste of resources in dealing with false alarms. Future work with time series models could reduce detection times and the persistence of anomaly effects. More complex machine learning models could help to correctly differentiate a higher number of anomalies and differentiate anomalies that coincide.

8. Declaration of competing interest

We declare that we have no significant competing interests, including financial or non-financial, professional, or personal interests, that interfere with the complete and objective presentation of the work described in this manuscript.

9. Funding

The author(s) received no financial support for this article's research, authorship, and/or publication.

10. Author contributions

All the authors conceived, designed, performed the analysis, and corrected the paper. M.R. collected the data and edited the document.

11. Data availability statement

The authors confirm that the data supporting the findings of this study are available within the article [and/or] its supplementary materials.

References

[1] S. Alexandra and M. Vitaliy, "Approaches for data collection and process standardization in smart manufacturing: systematic literature review," *Journal of Industrial Information Integration*, p. 100578, 2024.

[2] A. Angelopoulos, E. T. Michailidis, N. Nomikos, P. Trakadas, A. Hatziefremidis, S. Voliotis, and T. Zahariadis, "Tackling faults in the industry 4.0 era—a survey of machine-learning solutions and key aspects," *Sensors*, vol. 20, no. 1, p. 109, 2019.

[3] M. Rodríguez, D. P. Tobón, and D. Múnera, "Anomaly classification in industrial internet of things: A review," *Intelligent Systems with Applications*, p. 200232, 2023.

[4] C. Ni and S. C. Li, "Machine learning enabled industrial iot security: Challenges, trends and solutions," *Journal of Industrial Information Integration*, p. 100549, 2024.

[5] E. Luján, A. Otero, S. Valenzuela, E. Mocskos, L. A. Steffenel, and S. Nesmachnow, "An integrated platform for smart energy management: the cc-sem project," *Revista Facultad de Ingeniería Universidad de Antioquia*, no. 97, pp. 41–55, 2020.

[6] M. Chen, S. Mao, and Y. Liu, "Big data: A survey," *Mobile networks and applications*, vol. 19, no. 2, pp. 171–209, 2014.

[7] A. Karkouch, H. Mousannif, H. Al Moatassime, and T. Noel, "Data quality in internet of things: A state-of-the-art survey," *Journal of Network and Computer Applications*, vol. 73, pp. 57–81, 2016.

[8] N. Ghosh, K. Maity, R. Paul, and S. Maity, "Outlier detection in sensor data using machine learning techniques for iot framework and wireless sensor networks: A brief study," in *2019 International Conference on Applied Machine Learning (ICAML)*. IEEE, 2019, pp. 187–190.

[9] N. Mohamudally and M. Peermamode-Mohaboob, "Building an anomaly detection engine (ade) for iot smart applications," *Procedia computer science*, vol. 134, pp. 10–17, 2018.

[10] S. Afroz, S. A. Islam, S. N. Rafa, and M. Islam, "A two layer machine learning system for intrusion detection based on random forest and support vector machine," in *2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE)*. IEEE, 2020, pp. 300–303.

[11] N. Moustafa, N. Koroniotis, M. Keshk, A. Y. Zomaya, and Z. Tari, "Explainable intrusion detection for cyber defences in the internet of things: Opportunities and solutions," *IEEE Communications Surveys & Tutorials*, 2023.

[12] Y. Wang, M. Perry, D. Whitlock, and J. W. Sutherland, "Detecting anomalies in time series data from a manufacturing system using recurrent neural networks," *Journal of Manufacturing Systems*, 2020.

[13] C. EUREKA, G. de Antioquia *et al.*, "Engineering for transformation," in *Expo Ingenieria*. Fondo Editorial EIA, 2022.

[14] G. Tertytchny, N. Nicolaou, and M. K. Michael, "Classifying network abnormalities into faults and attacks in iot-based cyber physical systems using machine learning," *Microprocessors and Microsystems*, vol. 77, p. 103121, 2020.

[15] L. E. Contreas-Bravo, N. Nieves-Pimiento, and K. González Guerrero, "Prediction of university-level academic performance through machine learning mechanisms and supervised methods," *Ingeniería*, vol. 28, no. 1, 2023.

[16] Z.-X. Guo and P.-L. Shui, "Anomaly based sea-surface small target detection using k-nearest neighbor classification," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 56, no. 6, pp. 4947–4964, 2020.

[17] S. Rani, K. Tripathi, Y. Arora, and A. Kumar, "Analysis of anomaly detection of malware using knn," in *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, vol. 2. IEEE, 2022, pp. 774–779.

[18] A. Abid, A. Kachouri, A. B. F. Guiloufi, A. Mahfoudhi, N. Nasri, and M. Abid, "Centralized knn anomaly detector for wsn," in *2015 IEEE 12th International Multi-Conference on Systems, Signals & Devices (SSD15)*. IEEE, 2015, pp. 1–4.

[19] P. Naraei, A. Abhari, and A. Sadeghian, "Application of multilayer perceptron neural networks and support vector machines in classification of healthcare data," in *2016 Future Technologies Conference (FTC)*. IEEE, 2016, pp. 848–852.

[20] S. Shakya and S. Sigdel, "An approach to develop a hybrid algorithm based on support vector machine and naive bayes for anomaly detection," in *2017 International Conference on Computing, Communication and Automation (ICCCA)*.

IEEE, 2017, pp. 323–327.
[21] S. A. Arenas-Hoyos and Á. Bernal-Noreña, "Support vector machines implementation over integers modulo-m and residue number system," *Dyna*, vol. 90, no. 226, pp. 17–26, 2023.
[22] Scikit-Learn, "sklearn.svm.SVC," https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html, 2022, [Online; accessed 10-August-2022].
[23] T.-H. Lin and J.-R. Jiang, "Anomaly detection with autoencoder and random forest," in *2020 International Computer Symposium (ICS)*. IEEE, 2020, pp. 96–99.
[24] K. Bhuva, K. Srivastava *et al.*, "Comparative study of the machine learning techniques for predicting the employee attrition," *IJRAR-International Journal of Research and Analytical Reviews (IJRAR)*, vol. 5, no. 3, pp. 568–577, 2018.
[25] R. Porteiro, L. Hernández-Callejo, and S. Nesmachnow, "Electricity demand forecasting in industrial and residential facilities using ensemble machine learning," *Revista Facultad de Ingeniería Universidad de Antioquia*, no. 102, pp. 9–25, 2022.
[26] Y.-C. Lai, K.-Z. Zhou, S.-R. Lin, and N.-W. Lo, "F1ow-based anomaly detection using multilayer perceptron in software defined networks," in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2019, pp. 1154–1158.
[27] H. Zhang, S. Shen, and Y. Shen, "Power system transient stability evaluation based on multilayer perceptron neural network," in *2021 China Automation Congress (CAC)*. IEEE, 2021, pp. 3313–3316.
[28] T. Huuhtanen and A. Jung, "Anomaly location detection with electrical impedance tomography using multilayer perceptrons," in *2020 IEEE 30th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2020, pp. 1–6.
[29] L. Van Efferen and A. M. Ali-Eldin, "A multi-layer perceptron approach for flow-based anomaly detection," in *2017 international symposium on networks, computers and communications (ISNCC)*. IEEE, 2017, pp. 1–6.
[30] S. Ranjeeth, V. A. K. Kandimalla *et al.*, "Predicting diabetes using outlier detection and multilayer perceptron with optimal stochastic gradient descent," in *2020 IEEE India Council International Subsections Conference (INDISCON)*. IEEE, 2020, pp. 51–56.
[31] X.-J. Shen, Y. Dong, J.-P. Gou, Y.-Z. Zhan, and J. Fan, "Least squares kernel ensemble regression in reproducing kernel hilbert space," *Neurocomputing*, vol. 311, pp. 235–244, 2018.
[32] B. B. Hazarika, D. Gupta, and P. Borah, "An intuitionistic fuzzy kernel ridge regression classifier for binary classification," *Applied Soft Computing*, vol. 112, p. 107816, 2021.
[33] S. Akila and U. S. Reddy, "Credit card fraud detection using non-overlapped risk based bagging ensemble (nrbe)," in *2017 IEEE international conference on computational intelligence and computing research (ICCIC)*. IEEE, 2017, pp. 1–4.
[34] Y. Zahid, M. A. Tahir, and M. N. Durrani, "Ensemble learning using bagging and inception-v3 for anomaly detection in surveillance videos," in *2020 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2020, pp. 588–592.
[35] X. Berger, "RPi-Monitor," https://xavierberger.github.io/RPi-Monitor-docs/index.html, 2018, [Online; accessed 10-August-2022].