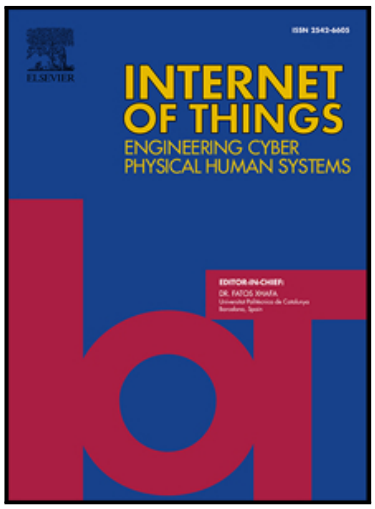

Anomaly Detection System for Data Quality Assurance in IoT infrastructures based on Machine Learning

Lucia Arnau Muñoz , José Vicente Berná Martínez , Francisco Maciá Pérez , Iren Lorenzo Fonseca

PII: S2542-6605(24)00037-4
DOI: https://doi.org/10.1016/j.iot.2024.101095
Reference: IOT 101095

To appear in: *Internet of Things*

Received date: 4 August 2023
Revised date: 30 September 2023
Accepted date: 27 January 2024

Please cite this article as: Lucia Arnau Muñoz , José Vicente Berná Martínez , Francisco Maciá Pérez , Iren Lorenzo Fonseca , Anomaly Detection System for Data Quality Assurance in IoT infrastructures based on Machine Learning, *Internet of Things* (2024), doi: https://doi.org/10.1016/j.iot.2024.101095

Anomaly Detection System for Data Quality Assurance in IoT infrastructures based on Machine Learning

*Lucia Arnau Muñoz[1,2], José Vicente Berná Martínez[1,3], Francisco Maciá Pérez[1,4], and Iren Lorenzo Fonseca[1,5]

[1]University of Alicante, Carretera San Vicente del Raspeig s/n, 03690, San Vicente del Raspeig

[2*]lucia.arnau@ua.es
[3]jvberna@ua.es
[4]pmacia@dtic.ua.es
[5]iren.fonseca@ua.es

* Corresponding Author

Abstract

The inclusion of IoT in digital platforms is very common nowadays due to the ease of deployment, low power consumption and low cost. It is also common to use heterogeneous IoT devices of ad-hoc or commercial development, using private or third-party network infrastructures. This scenario makes it difficult to detect invalid packets from malfunctioning devices, from sensors to application servers. These invalid packets generate low quality or erroneous data, which negatively influence the services that use them. For this reason, we need to create procedures and mechanisms to ensure the quality of the data obtained from IoT infrastructures, regardless of the type of infrastructure and the control we have over them, so that the systems that use this data can be reliable. In this work we propose the development of an Anomaly Detection System for IoT infrastructures based on Machine Learning using unsupervised learning. We validate the proposal by implementing it on the IoT infrastructure of the University of Alicante, which has a multiple sensing system and uses third-party services, over a campus of one million square meters. The contribution of this work has been the generation of an anomaly detection system capable of revealing incidents in IoT infrastructures, without knowing details about the infrastructures or devices, through the analysis of data in real time. This proposal allows to discard from the IoT data flow all those packets that are suspected to be anomalous to ensure a high quality of information to the tools that consume IoT data.

Keywords – Internet of Things; Anomaly Detection; Machine Learning; Isolation Forest;

1. Introduction

IoT infrastructures are now widely deployed in our society. They were first driven mainly by the needs of large companies to control certain processes, speed up and improve their efficiency, or reduce the occurrence of errors [1]. Subsequently, due to their use in Smart City environments, for any type of use, such as air quality monitoring, traffic monitoring, waste management or citizen safety [2]. Nowadays, they have become common sensor systems in

Digital Twin platforms [3], where a realistic representation of the controlled world requires knowledge of the real-time state of the system. It is precisely in these latter environments, where data quality is crucial to achieve the objectives.

These systems also generate new challenges and problems, some of them concerning the proper functioning of the infrastructures themselves. These types of problems related to monitoring anomalies in the performance of infrastructures have been widely addressed from the perspective of traditional TCP/IP ethernet networks, using for example network intrusion detection systems (IDS), which is just a subtype of anomaly detection systems (ADS) [4] but focused on the use of infrastructures. Such systems are capable of monitoring events occurring in the infrastructure, detecting anomalies of various nature, and generating an automated and controlled processing of these events to ensure the proper functioning of the system [5]. The strength of these systems lies in the use of highly standardised and accepted protocols, such as TCP/IP, and services whose behaviour is tightly controlled, such as HTTP, FTP, STMP, etc. However, in IoT networks we find a large number of technologies, encodings, packaging, applications and in general much more heterogeneous systems where we cannot analyse headers as is done in traditional IDSs over Ethernet networks [6]. In addition, in IoT environments it is common to use infrastructures based on Open Source initiatives, such as TTN [7], and therefore the data-emitting devices cross intermediate networks that are beyond the control of the administrator of the IoT sensors that emit data, so there is even metadata over which there is no control or access. It is also very common to find that, in our control system platform, Smart City platform or digital twin, the information that is being integrated comes from different subsystems, manufactured by different companies, with different technologies and different natures. In order for these tools to work as expected, it is necessary to ensure that the data with which we are going to feed them are valid and have an adequate degree of quality, discarding anything that may come from an element of the infrastructure that exhibits a malfunction. Likewise, it is necessary to do so even when we do not have details about the infrastructure that originates the data. That is why, to ensure the quality of the data generated by the IoT infrastructure, it is necessary to conceive mechanisms and procedures to be applied on these IoT data flows, of which the types of incidents that can be detected are unknown. That is, to create an Anomaly Detection System (ADS) that is capable of working on information generated by the IoT, detecting everything that is out of the expected normality.

For an ADS to work correctly and efficiently when detecting possible threats or anomalous elements, it is necessary to follow a series of processes [8]: data collection to obtain the parameters of the packet to be analysed; generation of rules and algorithms for the definition of anomalies; execution of filters and analysis using the rules and algorithms on the collected packets; and detection and treatment of the detected anomalies. Only the systematisation of the processes to be carried out can ensure a good result, and, in addition, these processes must be adapted to the scope of the problem to be addressed, since the search for anomalies or outliers requires techniques and strategies coupled to the environment [9]. If we look at some of the contributions that review the different techniques for the creation of anomaly detection systems [10], we can see that the systematic treatment of the data is one of the first tasks to be completed before moving on to the selection and training of the Machine Learning (ML) algorithms.

Once the data adequacy process has been completed, it is essential to select an algorithm in accordance with the objective to be achieved, so that the resulting trained model can correctly ensure the operation of the infrastructure. This work proposes the development of an ADS on IoT infrastructures based on Machine Learning, so that the system can be sufficiently generic to cover a wide range of platforms and at the same time be able to detect anomalies in the infrastructures. The objective is not to diagnose the infrastructures, to know the exact incident that is occurring, but to be able, on real-time traffic, to classify the data packets we receive as valid or invalid, regardless of their content or origin. In this way, a filter can be generated prior to the injection of this data on the applications that use it and prevent incorrect information from contaminating the services.

The paper proposes the design and development of an ADS system based on Machine Learning and also to validate its operation, it is instantiated on the Smart City platform of the University of Alicante, in which several IoT systems coexist, thus generating the previous data analysis that ensures the quality of the data before being used by the platform. The motivation to propose, develop and implement this model in our platform comes from the need to control a large infrastructure, in which different interconnected sensorization areas coexist, in which being aware of each anomaly that may occur is a complex and laborious work for the people responsible for its management. The rest of the work is organised into the following sections: section 2 contains a preliminary study of the techniques and problems related to the project objectives; section 3 shows the design of the ADS proposal and the processes and algorithms involved; section 4 carries out the instantiation of the system on a real platform, the Smart University platform of the University of Alicante; section 5 finally draws the main conclusions of the work and sets out the lines of future work.

2. Background and related work

With the increasing evolution of cities and the lifestyle of the citizens living in them, IoT environments are evolving faster and faster, mainly due to the number of devices in these environments, using different communication protocols, great diversity of data to be sent, and varied formats and packaging for transmission. This is why the level of complexity for their control also increases along with the evolution of these systems, and with it, the need to develop solutions that can guarantee their security, efficiency, and reliability in these heterogeneous systems [11].

This complexity has derived in a need to expand research to find feasible solutions to the control of these systems, and over the last few years, the use of Machine Learning based applications has been chosen, so that they can be applied to IoT environments, since they are already implemented in many different areas of our daily life, such as medicine, in research or applications such as cancer detection [12].

Moreover, within the new difficulties that these environments already present when controlling them, even with the use of artificial intelligence, we must consider the aforementioned heterogeneity of the sensorisation devices, both in terms of the values they emit and the rest of the parameters they present, including their semantics and syntax [13], since they are still systems with a dynamic nature. This is why developing an automated model for this type of detection is a challenge to say the least, since the data for training cannot always be labelled, as required in some Machine Learning algorithms, as it can be very

difficult to categorise them for this purpose. In addition, the data often contains noise and other types of values that can interfere with the reliability of the data, thus resulting in false anomalies [14]. Even today, even with the information available to address this problem, there is still no definitive solution due to the diversity of these environments, coupled with the lack of standardisation in the IoT domain.

Although today's IoT environments are still very much in flux, AI-based techniques have managed to provide a new approach from which to work and continue to gain knowledge. In AI, there are different valid techniques for anomaly detection, which already take into account this diversity of the system, both in the data sent and in the device that sends it [15], with approaches focused on Machine Learning to adjust to these changes in the data, or other techniques used in the transformation of the data itself, such as its normalisation before the application of the algorithm [16].

Some of the most outstanding algorithms, mainly due to their great adaptive capacity, are Neural Networks [17], Support Vector Machines (SVM) [18], or Random Forest [19], working satisfactorily in systems where the data used have a heterogeneous and uncategorized nature [20]. Some examples of their use are SPAM detection [21], mainly developed using supervised learning techniques, such as collaborative approaches, or content-based models [22] or the detection of malicious URLS [23], using techniques such as Decision Trees or Random Forest [24], mainly due to the large amount of features that must be extracted for this type of detections.

Within the field of anomaly detection systems development, algorithms based on the creation of random trees and decision trees show high efficiency and accuracy in detecting such anomalies, testing these algorithms with different types of datasets [25]. In the context of IoT, the Isolation Forest [26] algorithm has been one of the most prominent ones currently, due to its effectiveness in such varied, heterogeneous, and dynamic environments, as well as presenting good results when working with a large amount of data volumes, such as data broadcasts from IoT devices. Some of the reasons for its effectiveness is that it does not require a lot of training data when developing the model, and its decision making returns a concrete yes or no result, identifying the anomaly or not [27]. An example of the use of this technique can be found in the detection of fraudulent banking transactions [28]. However, these proposals tend to focus detection on already identified anomalies and have not been applied to open and uncertain scenarios.

Table 1. Comparative summary of the main techniques used for anomaly detection.

Algorithm	Type	Description	Typical IoT applications	Advantages	Disadvantages
Random Forest	Supervised	Set of decision trees.	Classification and regression	Good performance and handling characteristics.	Requires adjustment of hyperparameters.
Neural Networks	Supervised	Model inspired by the human brain.	Pattern recognition	Ability to learn complex relationships.	Requires large data sets and computation
SVM	Supervised	Finds a hyperplane that maximizes margin.	Anomaly detection	Efficient in high-dimensional spaces.	Requires adjustment of hyperparameters.
Isolation Forest	Not Supervised	Based on construction of random trees.	Anomaly detection	Efficient and scalable.	Sensitive to noise.

3. Proposed solution

For the development of our proposal, we will use a sequence of independent processes that form the framework, as shown in Fig. 1. The aim of this framework is to provide a generic procedure that can be extrapolated to any IoT infrastructure.

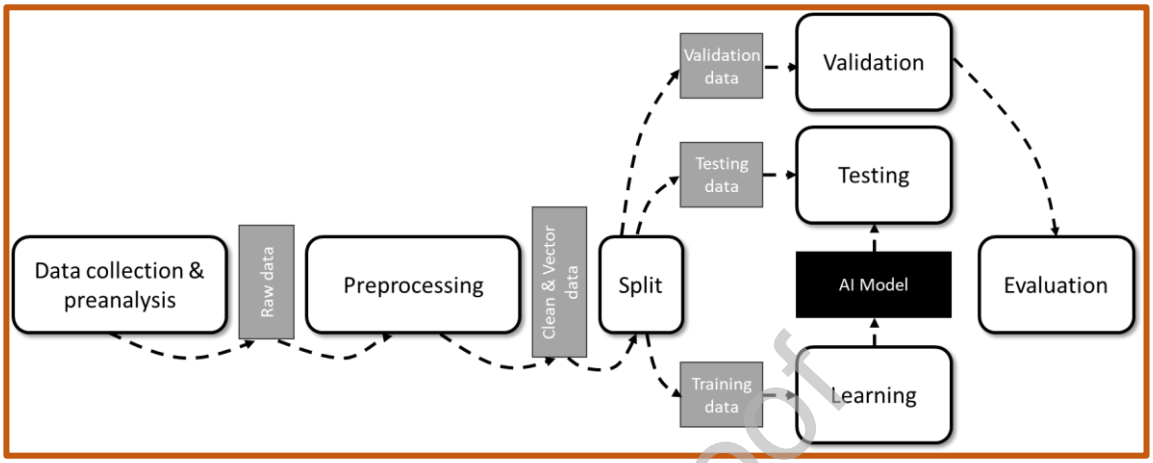

Figure 1. Framework for general anomaly detection in IoT.

The first process is responsible for collecting the dataset from the IoT infrastructure and performing the pre-analysis of the data. In this process, data capture from IoT data channels is performed and stored in persistence systems, and meticulous observation of the data is performed to know the types of data captured. The following process performs a processing of the data in such a way that invalid or incomplete data is cleaned, feature engineering is carried out to determine which of all the data received are necessary for the study, the construction of processable data vectors is performed and finally a data representation is made in order to be able to observe the distribution of the dataset. This obtained dataset is divided into three subsets with a ratio of 70-20-10. The first subset will be used for training, the second will be used for testing the training, and the third for validating the testing and therefore the trained model. Through these processes of testing and validation, or cross-validation, we can ensure that the trained model meets its objectives. Finally, the last process will be the evaluation of the model through its implementation in a real scenario.

3.1 Data collection and pre-analysis

To develop our proposal, we used a dataset collected directly from the IoT infrastructures of the University of Alicante. The use of open-source datasets such as those provided by Kaggle [29] could allow us to follow strategies and proposals from other authors and compare results more easily, but it did not provide us with a realistic context, nor would it be useful to apply it to our infrastructures. It was therefore decided to generate our own dataset. For this purpose, a packet collector was developed using Message Queuing Telemetry Transport (MQTT) to connect to the IoT management system based on TheThingStack [30]. A dataset of approximately 320,000 packets has been collected, the equivalent of

one day of sensing on the platform. Each packet consists of 117 features. This dataset is made up of sensor packages from different sources:

- Climatology: sensorisation data on environment such as outdoor temperature, relative humidity, light level, UV rays, rain.
- Quality&Comfort: sensorisation related to indoor spaces of buildings such as temperature, humidity, air quality, CO2 concentration, suspended particles, presence of harmful gases, etc.
- Consumption: information related to infrastructure consumption, water, electricity, and gas consumption.
- Production: sensorisation of the energy generation of photovoltaic plants.
- Recharging electric vehicles: data on the use and consumption of electric vehicle charging stations.
- WIFI: data on the real-time use of the campus wireless network infrastructures.
- Luminaire: data concerning lighting systems including outdoor streetlights, signage, monument courtesy lights, etc.
- Others: other sensorisations, where data received that do not belong to any previous collection will be dumped.

In addition, the amount of data generated by each sensor group is different. Fig 2 shows the distribution of data per set.

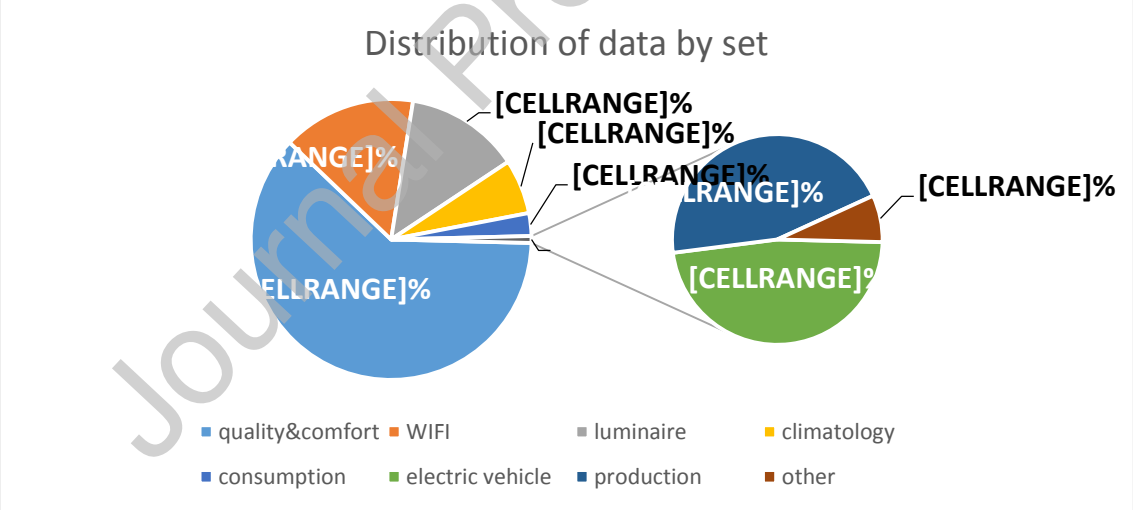

Figure 2. Percentage of data of the total collected by type of sensorisation.

In the pre-analysis, all fields in each packet were inspected and from the initial 117, only 47 fields were selected. All fields that are unique tokens, additional timestamps or unique identifiers are eliminated. These are fields that do not correlate or do not provide useful information in principle.

3.2. Processing

The use of Machine Learning techniques requires the ability to explore the dataset, i.e., it must be able to undergo classification techniques. This involves processing the dataset to perform several operations.

First, null or invalid values must be dealt with, cleaning the dataset of these records to avoid contamination. Unexpected values are also analysed in order to treat them and transform them into expected values. The aim is to have a dataset that can be processed by the algorithms.

On the set of 47 selected features, a study of the entropy of the data has been carried out through correlation, to decide which ones are determinant in the Machine Learning algorithms. Fig. 3 shows the result of the study. As can be seen, there are many variables that, although important from a technical point of view, such as the timestamp of the data emission, do not contribute value within the correlation, because they are fields that will always grow.

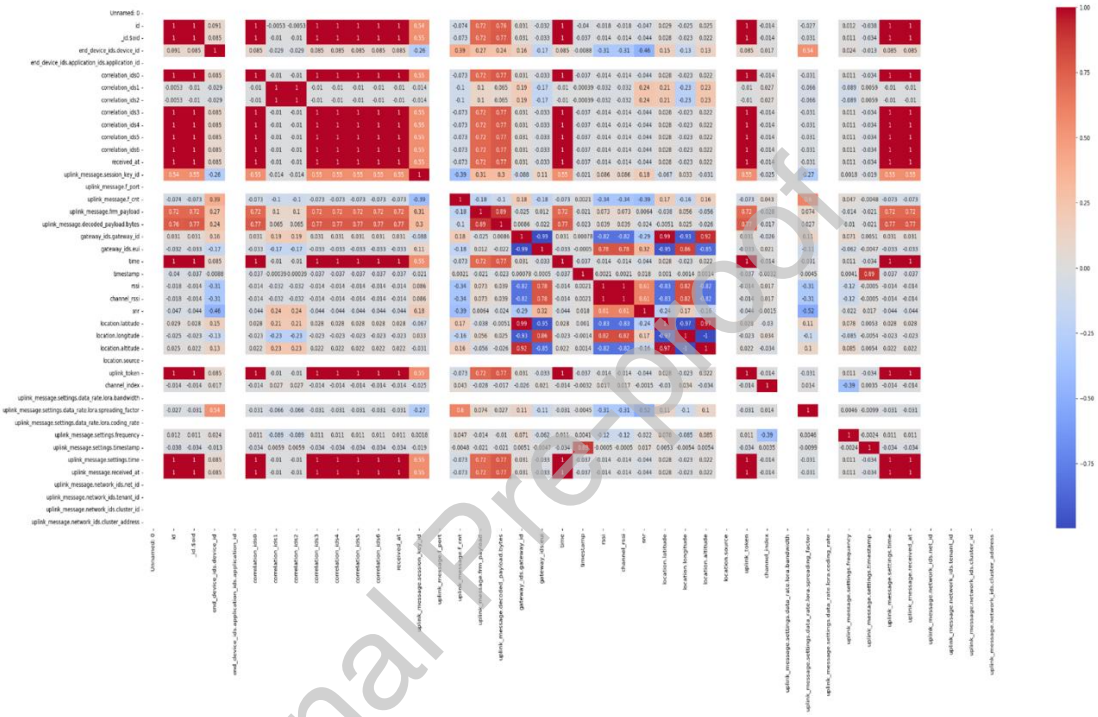

Figure 3. Results of the correlation study.

Through the correlation matrix it was decided to finally select 18 characteristics, which are shown in table 2.

Regarding the decision factors for feature selection, we have focused on two determining elements, the numerical factor obtained in the correlation study, from the perspective of "Pearson's Correlation" [31], and also on the know-how of the technical staff, since the experience gained from daily work with IoT traffic helps us to know which attributes can contribute to indicate infrastructure failures.

The elements that were finally selected have correlation indexes from 0.1 to 0.99, indicating different levels of correlation between the different values. If we look at the central part of Figure 3, we realize that most of the values present fall within this range of correlation values between each other. After this first indicator, we move on to the experience factor of the technical staff. During data transmission, there are essential values to control such as the noise signal, the received signal strength indicator, the byte packet sent, or the gateway that received the packet, among others. For this reason, these attributes will be taken into account in the analysis regardless of their correlation factor.

The discarded values are fundamentally unique values, credentials to be exact, which are repeated during data sending, providing information such as the cluster in which the device is located, or the fixed port through which it works, so it did not provide us with useful information for the detection of anomalies in our system.

Table 2. List of attributes to be used for algorithm training.

Attributes	Description
received_at	The date and time the data packet was received at the TTN server.
uplink_message.session_key_id	The uplink message session key identifier. This key is used in data encryption.
uplink_message.frm_payload	The raw data field of the uplink message.
uplink_message.decoded_payload.bytes	The decoded data contained in the uplink message, expressed in bytes.
gateway_ids.gateway_id	The unique identifier of the gateway that received the message.
time	The date and time of the message
timestamp	A timestamp indicating when the message was received, usually in Unix timestamp format.
rssi	Received Signal Strength Indicator - Received Signal Strength Indicator, which measures the strength of the signal received from the device.
channel_rssi	The signal strength on the specific channel on which the message was received.
snr	Signal-to-Noise Ratio - The signal-to-noise ratio, which indicates the quality of the received signal.
location.latitude	The geographic latitude of the location of the gateway that received the message.
location.longitude	The geographic longitude of the location of the gateway that received the message.
location.altitude	The geographic altitude of the location of the gateway that received the message.
uplink_token	A token associated with the uplink message.
channel_index	The index of the channel used to transmit the message.
uplink_message.settings.data_ rate.lora.spreading_factor	The spreading factor used in the LoRa modulation of the message.
uplink_message.received_at	The date and time the uplink message was received.
uplink_message.consumed_airtime	The airtime consumed by the uplink message in the network.

Finally, another important operation is the transformation of categorical values to nominal values. In the dataset there are many columns of data indicating identifiers of network elements, services, or application. One Hot Encoding labelling is used for this purpose, converting the text data into feature vectors that can be further processed.

3.3. Theoretical considerations.

After the analysis of viable AI techniques in our context, it was determined that the Isolation Forest algorithm can generate the most correct results. This algorithm belongs to the most widely used unsupervised algorithms for anomaly detection and provides great flexibility in training as it does not need to label the data as valid or invalid from the beginning.

This type of algorithm works on the detection of erroneous and unlabelled values within the datasets, also called outliers or anomalies. When training and implementing this type of algorithms, it has been demonstrated in different studies its efficient performance when dealing with large volumes of data, in addition to presenting a linear time complexity with a very low memory cost [32]. In our work, we will define outliers as: "an observation that, being atypical and/or erroneous, deviates decidedly from the general behaviour of the experimental data with respect to the criteria that should be analysed about it" [33].

The methodology employed by the Isolation Forest algorithm is based on the detection of outliers by using decision trees to isolate outliers from the rest of the data. To do this, a feature is selected and a

random split between the minimum and maximum value is performed, repeating this process until all possible data splits are performed, or a specified limit on the number of splits is reached. The number of divisions needed to isolate a data item will be smaller for an outlier, while for normal values the number of divisions will be larger, since the algorithm attributes to each division an "anomaly score", calculated as the average of the number of subdivisions needed to isolate the outlier. The "anomaly score" is a value calculated using the following formula (1):

$$s(x,n) = 2\frac{-E(h(x))}{c(n)} \quad (1)$$

The parameters of which are:

- h(x): is the average depth of constructed trees.
- c(n): is the average height to find a node in one of the trees.
- n: size of the dataset.
- s: if the value obtained is close to 1, it is generally an anomaly, while if the value of s is less than 0.5, it is a correct value.

The term E(h(x)) in formula (1) is calculated as:

$$E(h(x)) = \frac{\sum_{t\in\mathcal{F}} h_t(x) + \sum_{t\in\mathcal{F}} c(|l_t(x)|)}{|\mathcal{F}|} \quad (2)$$

where *t* is a tree, $c(|l_t(x)|)$ is a normalization factor needed when *t* is not fully grown (which estimates the average tree depth that can be constructed from $l_t(x)$) and $h_t(x) = |P_t(x)|$ with $P_t(x)$ being the path of *x*, the set of nodes visited by *x* from the root to the leaf containing *x*. From formula (1) it can be inferred that the score of an object *x* is proportional to the inverse of the average length of its path in the forest: if *x* ends in very deep leaves of the trees, its score will be quite low (close to 0), if on the contrary its path ends very soon the score will be high (close to 1) [34].

The reason for using this algorithm over other existing algorithms is mainly because it is an easily scalable algorithm for use on large datasets. In addition, it works well when features that may initially be irrelevant are included, as multi-modal datasets. This is the case for IoT infrastructures, where the cohesion or internal correlation between the data being sent is unknown, and we simply acquire a dataset with its corresponding parameters and want to detect outliers.

In terms of implementation for model building, we must consider the basic elements with which we can train and subsequently improve the accuracy of the result:

- "contamination", the amount of overall data that we expect to be considered outliers, indicates the estimated proportion of outliers that the dataset possesses. Based on this value, the limit by which the values are classified as anomalous or normal is set.
- “n_estimators", this value represents the number of isolation trees to be used to construct the Isolation Forest itself. Using higher estimator values can improve the accuracy for detection, but should always be appropriate to the dataset used, as it also results in increased training time. Varying the value of this parameter will serve to adjust the final performance of the model.

- “max_samples", number of observations used to train each tree; serves to control the maximum number of samples to be used to train each tree generated. You can use the value of auto which implies that all the samples in the dataset will be used, however, if you have a dataset that is too large, this value will have to be readjusted.
- “max_features", which indicates the maximum number of features to be used when splitting each node of the tree. By setting the value to 1, we specify that all available features will be used for each split. This parameter can be used as such in this case since many columns have been filtered during the data fitting process, however, if the dataset has many different columns, it can be adjusted in a way that improves performance and avoids overfitting.

4. Implementation and analysis of results

4.1. Experimental setup.

The experimentation was carried out on an HP Pro SFF 400 G9, with Windows 11 Pro 64-bit operating system, Intel® Core I7-12700 CPU up to 4.9GHz and 12 cores, 32GB RAM, and NVIDIA Quadro T400 graphics card. For the development of the algorithm, use was made of different libraries. NumPy [35] was used to carry out the relevant numerical arrays to process data, and Pandas [36] was used for the analysis and manipulation of structured data, providing the DataFrame format required for the algorithm. As for the implementation of the algorithm, the scikit-learn library [37] was used for its development, specifically the modules: *sklearn.ensemble.IsolationForest*, as the final algorithm for anomaly detection; *sklearn.model_selection.train_test_split*, to split the dataset for the training, testing and validation phases; sklearn.impute.SimpleImputer, mainly used for handling null values in the dataset, so that missing values are filled in a certain way; and finally, *sklearn.preprocessing.RobustScaler*, whose function is to scale features in a dataset, being very useful in datasets that may contain outliers.

For the implementation of our model in particular, the following parameters were used in the configuration of the Isolation Forest algorithm.

- contamination=float(0.1667). In the case of the training dataset, it is estimated that 16.67% of the data contains an erroneous value that must be detected, so the training is adjusted to this contamination rate.
- n_estimators=100
- max_samples='auto
- max_features=1.0

The choice of parameters depends on several issues and may condition the effectiveness of the model. The parameter "contamination" depends on the known number of erroneous packets, this value is known since the erroneous packets are introduced in a random but controlled way. The value of "max_samples" is set to "auto" to take all the data in the set, and the value of "max_features" is set to "1" to use all the properties of the dataset, since we have previously selected the significant properties. To set the value of "n_estimators", an empirical study was carried out for which: a reduced dataset was generated from the

original one; and the Isolation Forest algorithm was tested by varying only the value of n_estimators until reaching the minimum value above which detection was no longer improved.

Once the training is finished, we can visualise the results obtained for each field of the dataset. In order to have a graph for each field, we consider as a common element the "time" field (the timestamp of the packet), so the following graphs correspond to the time field, and another parameter of the dataset, such as snr, uplink_message.senssion_key_id,channel_index,uplink_message.setittings.data_rate.lora.spreading_factor,uplink_message.consumed_airtime or uplink_message.decode_payload.bytes. The following Fig. 4-9 is the result of the training with some variables where the outliers can be seen.

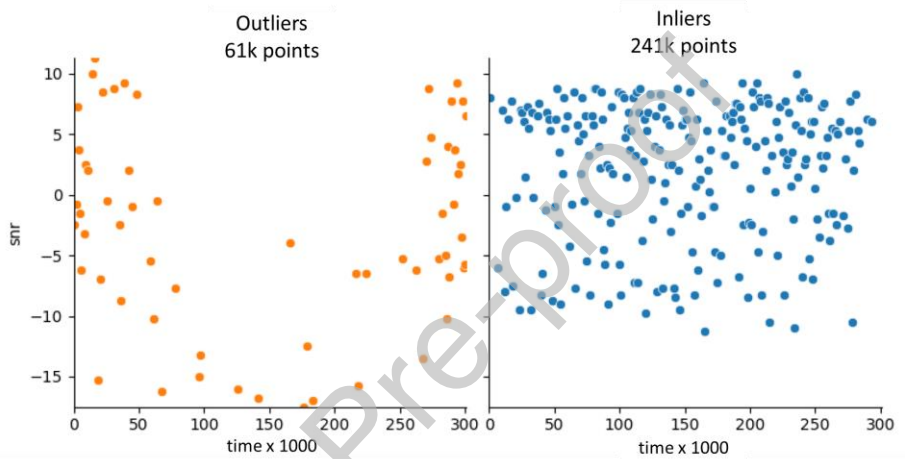

Figure 4. Result of the detection on the snr column in relation to time.

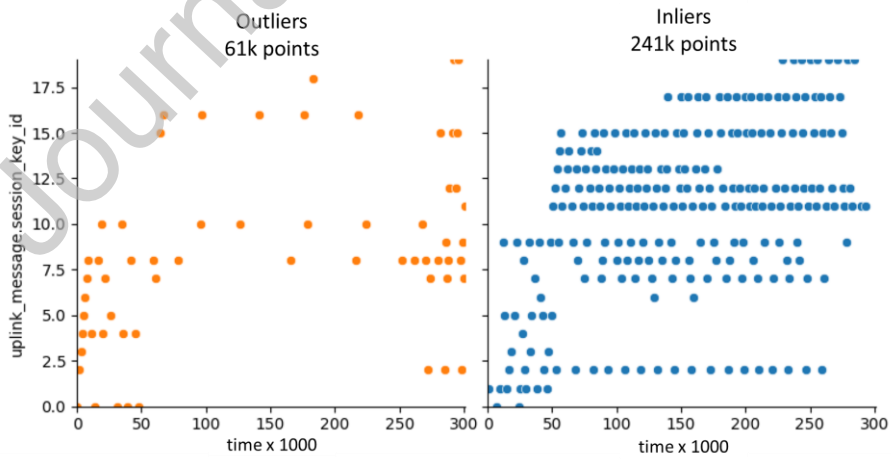

Figure 5. Detection result on the uplink_message_session_key_id column in relation to time.

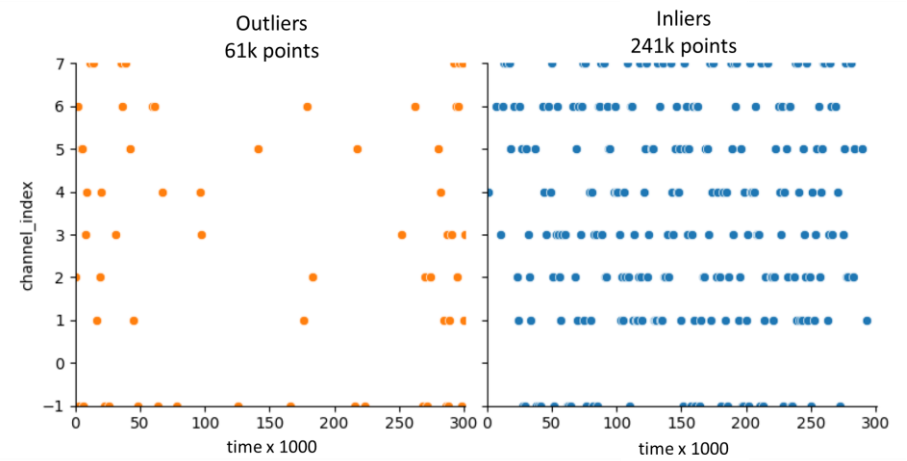

Figure 6. Detection result on the channel_index column in relation to time

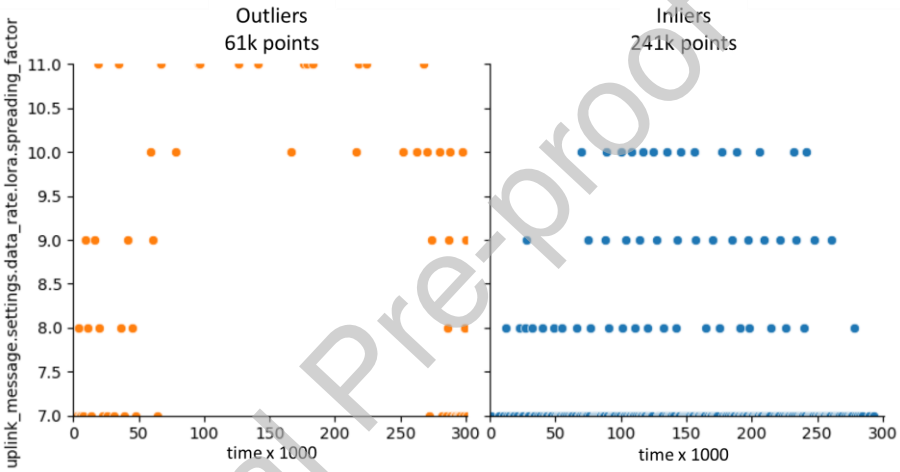

Figure 7. Result of the detection on the uplink_message_settings_data_rate.lora.spreading_factor column in relation to time.

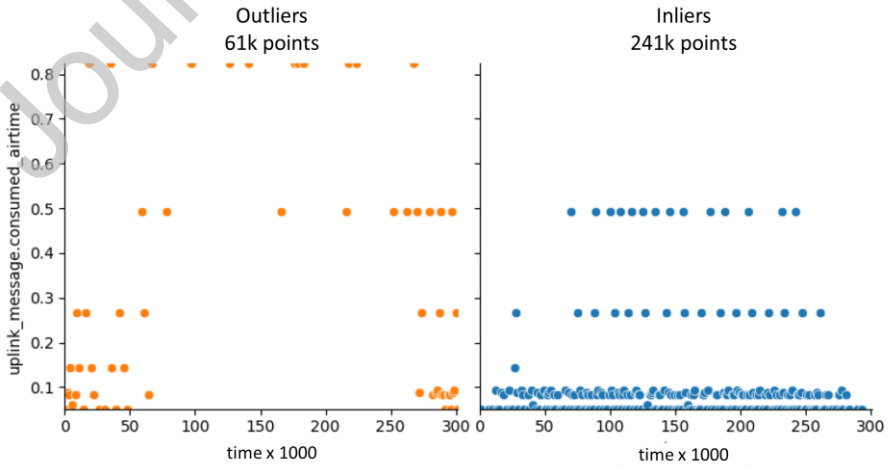

Figure 8. Result of the detection on the uplink_message.consumed_airtime column in relation to time

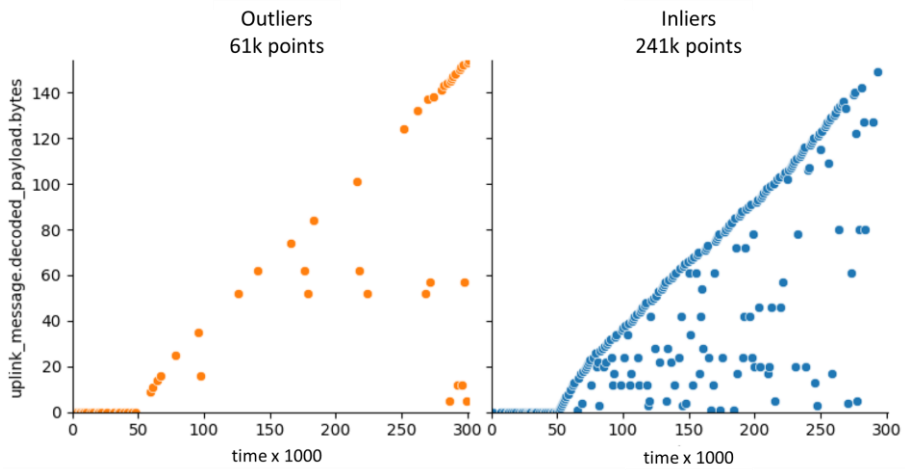

Figure 9. Result of the detection on the uplink_message.decoded_payload.bytes column in relation to time

4.2. Analysis of results and evaluation

The marking of outliers requires a subsequent analysis and evaluation, to validate whether or not errors are really being detected. To this end, a sampling has been carried out on the set of outliers, selecting several dozens to analyse them in detail and find out the causes of their cataloguing as outliers, and therefore anomalous value within the packet flow. The detailed analysis has identified that most of the anomalous packets are due to three types of malfunctions.

4.2.1. Drop in signal strength

One of the first anomalies detected by the system was a sudden change in the signal strength of the sensors. This issue in a device that is stable in its power was unusual, so the system flagged an anomaly at that instant, and after analysis of this detection, it was possible to locate that there had been a change in the position of the antenna that caused a shielding of the signal. Fig. 10 shows an instant in which this sudden change in signal power detected by the algorithm can be seen.

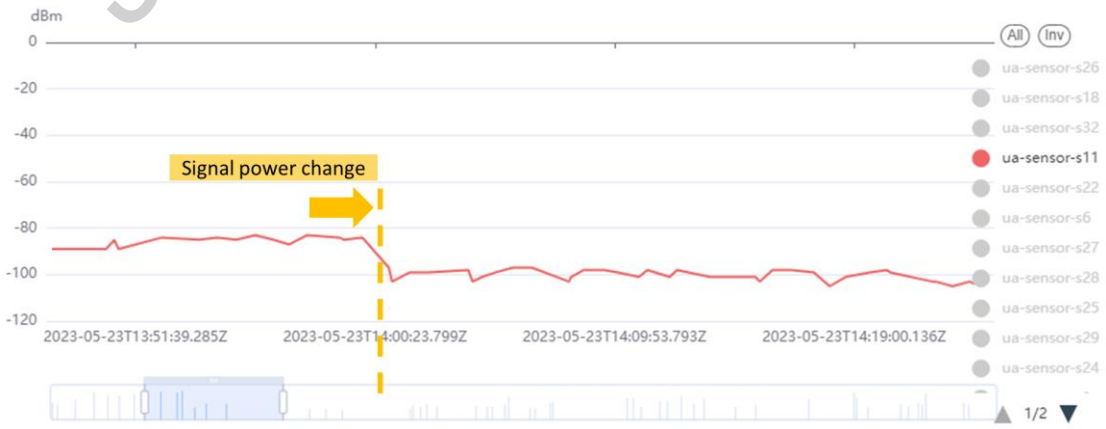

Figure 10. Sudden change of signal power

4.2.2. Detection of unusual gateways

Another anomaly detected by the algorithm is the appearance of gateways that do not belong to the organisation's infrastructure. This can occur because the Lora technology used has a large range and by using The Things Networks as the base application, there are more open-source antennas in the project nearby. Being something out of the ordinary, two possibilities are established, the probe has changed location and with it the gateways around it, or a new gateway has been activated within its range, without being one of the ones they have configured from the Smart University group. After analysing the packet following the detection of the anomaly, using the TTN Mapper tool [38] it was concluded that one of the devices had not only been moved from a room, but also taken out of the university area, without anyone notifying of this fact.

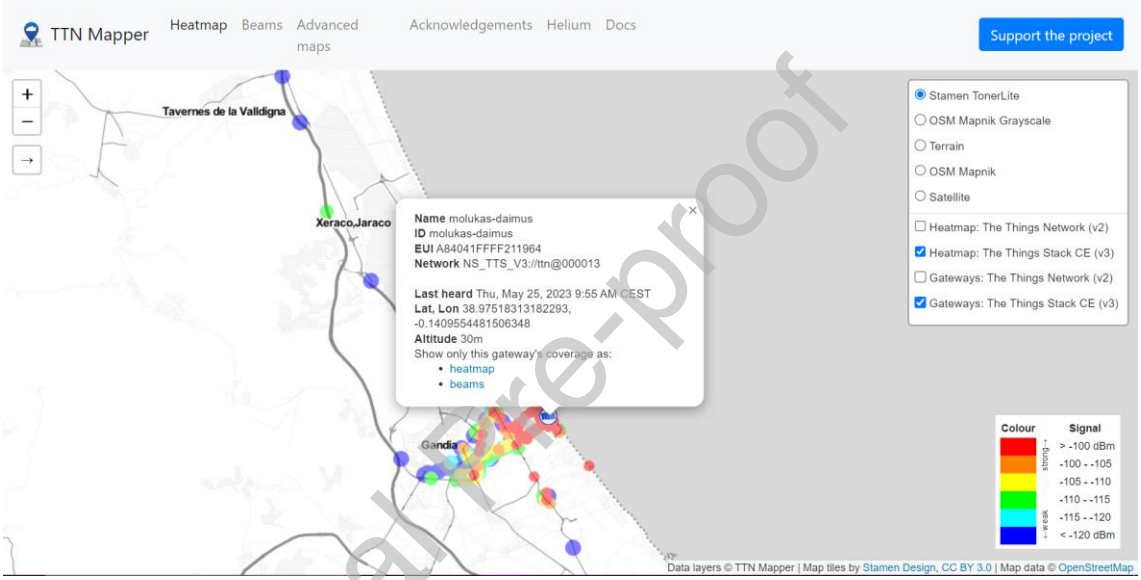

Figure 11. Location of the gateway detected from TTN Mapper

4.2.3. Receiving messages without a data packet

Finally, one of the most important anomalies detected is when a device stops sending data within the packet. This is the case when a sensor is not working properly, but the board to which it is connected to the sensor, together with its LoRa antenna, are in good condition, so that the packets arrive at the gateway, but there is no coded message inside. Fig. 12 shows an example of one of the devices that exhibited this behaviour.

Figure 12. Sample of the data sending flow until its sudden cut-off.

As can be seen in the image, from approximately 13:38, the sensor stopped emitting quantitative values, whether they were C02, GPS, or other sensors. In the IoT platform of the University of Alicante, the data could not be visualised, while the package had been processed. This implies that data is being received by the platform, but does not actually exist, which can lead to errors and contradictory actions in the IoT platform.

5. Conclusions

Several interesting results have been generated through this project. Firstly, a model for anomaly detection based on Machine Learning on IoT infrastructure has been proposed. One of the most outstanding characteristics of IoT infrastructures is precisely not having control over them, using services and even open intermediate infrastructures (as in our case TTN). This means that we are faced with the ignorance of information, which in traditional network infrastructure is very useful for detecting anomalies, such as IPs, MACs or protocols. Our model aims to perform broad-spectrum detections, i.e., to detect packets that are not correct, remove them from the data ingestion flow in the applications that use the IoT, and then analyse them and extract the anomaly.

An analysis of the characteristics of the IoT data packets has also been carried out, choosing the parameters to be considered for anomaly analysis after the correlation study, together with our own experience gained from working with the incoming packets on the TTN platform. This has allowed us to focus and narrow down the dimension in which these unexpected values can be detected, a factor that has helped to improve the quality of the anomalies detected and to optimise the training time.

Based on the model developed, an implementation has been carried out on the campus of the University of Alicante, based on IoT infrastructure, The Things Network, which uses the open-source software The Thing Stack as a base, which is widely used by the IoT community. This

instance of the algorithm has allowed to implement an anomaly detector on the real IoT platform of the University of Alicante, so that it has been possible to detect possible failures in the infrastructure and unexpected situations in terms of the behaviour of the sensors, has allowed to take measures in this regard, being an important support point for the overall monitoring of the project. In other words, the proposal is valid, it is useful, and it is up and running. And it can be used by the TTN community.

The proposed work has some limitations. The most important one is the resources needed to train the algorithm, since it is necessary to perform the learning process with large volumes of data, including a large variety of packets, originating from the correct operation of the infrastructure to quickly detect anomalies. This affects us especially when there are permanent changes in the infrastructures since it is necessary to retrain the algorithm with this new, but unknown, configuration. If retraining is not performed, the new infrastructure elements would generate packets that would be flagged as anomalies by the ADS. Another limitation of the proposal is that the anomaly is not classified, which requires further analysis by the administrator after detection.

As for possible improvements and future work, we want to implement different Machine Learning modules for anomaly detection and identification. First, we will look for the comparison of different algorithms that allow us to detect more situations of anomalies, to avoid that incorrect packets can "sneak in". We are even considering the possibility of having several algorithms processing in parallel and generating an aggregation of their results. This first step should serve to solve the problem of the resources needed for training. And, secondly, once we can discriminate packets and put them in a state of observation, other specialised algorithms can be used to generate a cataloguing that identifies the specific anomaly, such as: transmission failures, device subtraction, device malfunction or the implantation of unexpected devices (a very common problem in IoT, which appear to be sensors of unknown origin). Then, chaining anomaly detection systems with anomaly identification can help to have scalable, heterogeneous systems where there is uncertainty due to the lack of knowledge of uncontrolled intermediate IoT infrastructures. Being able to obtain a library of trained modules for different infrastructures would facilitate the construction of more efficient systems as it would only incorporate the necessary modules, and could even automate the system to search, among a possible collection of analysis and identification modules, those that best fit the infrastructure or circumstances.

Acknowledgements

This project has been funded by the UAIND22-01B project "Adaptive control of urban supply systems" of the University of Alicante.

References

1. Eason, G., Noble, B., & Sneddon, I. N. (1955). On certain integrals of Lipschitz-Hankel type involving products of Bessel functions. Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences, 247(935), 529-551.
2. Nasution, T. H., Muchtar, M. A., & Simon, A. (2019, October). Designing an IoT-based air quality monitoring system. In IOP conference series: materials science and engineering (Vol. 648, No. 1, p. 012037). IOP Publishing..
3. Minerva, R., Lee, G. M., & Crespi, N. (2020). Digital twin in the IoT context: A survey on technical features, scenarios, and architectural models. Proceedings of the IEEE, 108(10), 1785-1824.
4. Davis, J. J., & Clark, A. J. (2011). Data preprocessing for anomaly based network intrusion detection: A review. computers & security, 30(6-7), 353-375.
5. Veasey, T. J., & Dodson, S. J. (2014). Anomaly detection in application performance monitoring data. International Journal of Machine Learning and Computing, 4(2), 120.
6. Xu, K., Qu, Y., & Yang, K. (2016). A tutorial on the internet of things: from a heterogeneous network integration perspective. IEEE network, 30(2), 102-108.
7. Blenn, N., & Kuipers, F. (2017). LoRaWAN in the wild: Measurements from the things network. arXiv preprint arXiv:1706.03086.
8. Ahmad, Z., Shahid Khan, A., Wai Shiang, C., Abdullah, J., & Ahmad, F. (2021). Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Transactions on Emerging Telecommunications Technologies, 32(1), e4150.
9. Fernández Oliva, A., Maciá Pérez, F., Berna-Martinez, J. V., & Abreu Ortega, M. A Meth-od Non-Deterministic and Computationally Viable for Detecting Outliers in Large Datasets. (2020)..
10. Gu, T., Abhishek, A., Fu, H., Zhang, H., Basu, D., & Mohapatra, P. (2020, August). Towards learning-automation IoT attack detection through reinforcement learning. In 2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM) (pp. 88-97). IEEE.
11. Perera, C., Zaslavsky, A., Christen, P., & Georgakopoulos, D. (2014). Context aware computing for the Internet of Things: A survey. IEEE Communications Surveys & Tutorials, 16(1), 414-454.
12. Hasan, M. M., Islam, M. M., Zarif, I. I., & Hashem, M. (2019). Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. Internet of things, 7, 100059. https://doi.org/10.1016/j.iot.2019.100059
13. Ukil, A., Bandyoapdhyay, S., Puri, C., & Pal, A. (2016, March). IoT healthcare analytics: The importance of anomaly detection. In 2016 IEEE 30th international conference on advanced information networking and applications (AINA) (pp. 994-997). IEEE.
14. Chatterjee, A., & Ahmed, B. S. (2022). IoT anomaly detection methods and applications: A survey. Internet of Things, 19, 100568.

15. Nweke, H. F., Teh, Y. W., Al-Garadi, M. A., & Alo, U. R. (2018). Deep learning algorithms for human activity recognition using mobile and wearable sensor networks: State of the art and research challenges. Expert Systems with Applications, 105, 233-261.
16. Erhan, L., Ndubuaku, M., Di Mauro, M., Song, W., Chen, M., Fortino, G., ... & Liotta, A. (2021). Smart anomaly detection in sensor systems: A multi-perspective review. Information Fusion, 67, 64-79.
17. Albuquerque Filho, J. E., Brandão, L. C., Fernandes, B. J., & Maciel, A. M. (2022). A review of neural networks for anomaly detection. IEEE Access.
18. Hosseinzadeh, M., Rahmani, A. M., Vo, B., Bidaki, M., Masdari, M., & Zangakani, M. (2021). Improving security using SVM-based anomaly detection: issues and challenges. Soft Computing, 25, 3195-3223.
19. Primartha, R., & Tama, B. A. (2017, November). Anomaly detection using random forest: A performance revisited. In 2017 International conference on data and software engineering (ICoDSE) (pp. 1-6). IEEE.
20. Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM computing surveys (CSUR), 41(3), 1-58.
21. Cid, I., Janeiro, L. R., Méndez, J. R., Glez-Peña, D., & Fdez-Riverola, F. (2008). The impact of noise in spam filtering: a case study. In Advances in Data Mining. Medical Applications, E-Commerce, Marketing, and Theoretical Aspects: 8th Industrial Conference, ICDM 2008 Leipzig, Germany, July 16-18, 2008 Proceedings 8 (pp. 228-241). Springer Berlin Heidelberg.
22. Maimon, O. Z., & Rokach, L. (2014). Data mining with decision trees: theory and applications (Vol. 81). World scientific.
23. Vundavalli, V., Barsha, F., Masum, M., Shahriar, H., & Haddad, H. (2020, November). Malicious URL detection using supervised machine learning techniques. In 13th International Conference on Security of Information and Networks (pp. 1-6).
24. Janet, B., & Kumar, R. J. A. (2021, March). Malicious URL detection: a comparative study. In 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS) (pp. 1147-1151). IEEE.
25. Douiba, M., Benkirane, S., Guezzaz, A., & Azrour, M. (2023). An improved anomaly detection model for IoT security using decision tree and gradient boosting. The Journal of Supercomputing, 79(3), 3392-3411.
26. Lesouple, J., Baudoin, C., Spigai, M., & Tourneret, J. Y. (2021). Generalized isolation forest for anomaly detection. Pattern Recognition Letters, 149, 109-119.
27. Liu, F. T., Ting, K. M., & Zhou, Z. H. (2008, December). Isolation forest. In 2008 eighth ieee international conference on data mining (pp. 413-422). IEEE.
28. Dhankhad, S., Mohammed, E., & Far, B. (2018, July). Supervised machine learning algorithms for credit card fraudulent transaction detection: a comparative study. In 2018 IEEE international conference on information reuse and integration (IRI) (pp. 122-125). IEEE.
29. DS2OS traffic traces, Kaggle. https://www.kaggle.com/francoisxa/ds2ostraffictraces. Accessed September 2023.

30. The Things Industries. Platform for LoraWAN Networks Server. www.thethingsindustries.com. Accessed September 2023.
31. Jebli, I., Belouadha, F. Z., Kabbaj, M. I., & Tilioua, A. (2021). Prediction of solar energy guided by pearson correlation using machine learning. Energy, 224, 120109.
32. Mohy-eddine, M., Guezzaz, A., Benkirane, S., & Azrour, M. (2022). An effective intrusion detection approach based on ensemble learning for IIoT edge computing. Journal of Computer Virology and Hacking Techniques, 1-13.
33. Hawkins, D. M. (1980). Identification of outliers (Vol. 11). London: Chapman and Hall.
34. Mensi, A., & Bicego, M. (2019). A novel anomaly score for isolation forests. In Image Analysis and Processing–ICIAP 2019: 20th International Conference, Trento, Italy, September 9–13, 2019, Proceedings, Part I 20 (pp. 152-163). Springer International Publishing.
35. NumPy. package for scientific computing with Python. https://numpy.org/. Accessed September 2023.
36. Pandas. Open source data analysis and manipulation tool. https://pandas.pydata.org/. Accessed September 2023.
37. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, 2825-2830.
38. TTN Mapper. Tool to map coverage of The Things Networks. https://ttnmapper.org/. Accessed September 2023.

Declaration of interests

☐The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☒The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Jose Vicente Berna Martinez reports financial support was provided by University of Alicante.