applied sciences

MDPI

Article

Application of Machine Learning Algorithms for the Validation of a New CoAP-IoT Anomaly Detection Dataset

Laura Vigoya *, Alberto Pardal, Diego Fernandez and Victor Carneiro

Centre for Information and Communications Technology Research (CITIC), Campus de Elvina s/n, 15071 A Coruna, Spain
* Correspondence: l.v.vigoya@udc.es; Tel.: +34-881-011-213

Abstract: With the rise in smart devices, the Internet of Things (IoT) has been established as one of the preferred emerging platforms to fulfil their need for simple interconnections. The use of specific protocols such as constrained application protocol (CoAP) has demonstrated improvements in the performance of the networks. However, power-, bandwidth-, and memory-constrained sensing devices constitute a weakness in the security of the system. One way to mitigate these security problems is through anomaly-based intrusion detection systems, which aim to estimate the behaviour of the systems based on their "normal" nature. Thus, to develop anomaly-based intrusion detection systems, it is necessary to have a suitable dataset that allows for their analysis. Due to the lack of a public dataset in the CoAP-IoT environment, this work aims to present a complete and labelled CoAP-IoT anomaly detection dataset (CIDAD) based on real-world traffic, with a sufficient trace size and diverse anomalous scenarios. The modelled data were implemented in a virtual sensor environment, including three types of anomalies in the CoAP data. The validation of the dataset was carried out using five shallow machine learning techniques: logistic regression, naive Bayes, random forest, AdaBoost, and support vector machine. Detailed analyses of the dataset, data conditioning, feature engineering, and hyperparameter tuning are presented. The evaluation metrics used in the performance comparison are accuracy, precision, recall, F1 score, and kappa score. The system achieved 99.9% accuracy for decision tree models. Random forest established itself as the best model, obtaining a 99.9% precision and F1 score, 100% recall, and a Cohen's kappa statistic of 0.99.

Keywords: IoT; CoAP; sensors; dataset validation; machine learning; intrusion detection systems; analysis; metric; algorithm design

Citation: Vigoya, L.; Pardal, A.; Fernandez, D.; Carneiro, V. Application of Machine Learning Algorithms for the Validation of a New CoAP-IoT Anomaly Detection Dataset. *Appl. Sci.* **2023**, *13*, 4482. https://doi.org/10.3390/app13074482

Academic Editors: Bhanu Shrestha, Seongsoo Cho and Changho Seo

Received: 20 February 2023
Revised: 24 March 2023
Accepted: 27 March 2023
Published: 1 April 2023

1. Introduction

The construction of Internet of Things (IoT) networks has advanced significantly in the last couple of years, adding a new dimension to the world of information and communication technologies. The versatility in the use of sensors in Internet of Things (IoT) networks and the reduced cost through improved process efficiency, asset utilisation, and productivity have endorsed this network for new development opportunities in environments such as smart home management, intelligent transport system, smart energy management, and e-health.

The IoT consists of a large amount of nodes, and every node that connects with a server maintains a persistent connection. Besides, each node in the network connects with many other nodes, generating large loads on limited IoT devices. Congestion results in packet retransmissions, increasing energy consumption, latency, and packet loss, while reducing throughput and the packet delivery ratio (PDR) [1]. The use of specialised, lightweight IoT mechanisms can ease the computational load on the system, free up resources, and improve security. Thus, an alternative way to deal with the challenges of the IoT is the use of specially designed standards and communication protocols. At the transport layer, the main protocols are the transmission control protocol (TCP) and user datagram protocol (UDP).

However, the requirements for the distribution of messages may vary according to the specifications of the IoT application. Diverse and popular IoT protocols at the application layer available today are geared towards the machine to machine (M2M) communication, where one of the predominant protocols is constrained application protocol (CoAP). This protocol is a customised and compressed version of the hypertext transfer protocol (HTTP), and shares the same methods and principles, light weight, and low energy consumption [2].

Despite CoAP having difficulties managing congestion because of low bandwidth and higher network traffic, security is a hot topic as it does not have reliable standards for secure architectures [3]. Furthermore, the extensive implementation of sensor monitoring technologies, low-cost solutions, and high impacting diverse application domains generate an enormous amount of data while monitoring physical spaces and objects. The huge data streams collected can suffer from unhealthy behaviours or anomalies that can be normal in one scenario, while abnormal in another. Therefire, to reduce functional risks, avoid unseen problems, and prevent downtime of systems, analysis is required [4].

Depending on the type of analysis performed, security problems can be handled through intrusion detection systems (IDSs), which are intended to strengthen the security of information and communication systems. Intrusion dd=etection systems are classified as either signature-based or anomaly-based. Machine learning (ML) anomaly-based techniques have proven to be a reliable approach for the detection of network anomalies, including for IoT networks. These techniques take a decision based on the learning of anomalous/normal network behaviour. The machine learning approach is categorised into various learning paradigms: supervised, unsupervised, semisupervised, reinforcement, and active learning. Labelled data are required to train supervised learning approaches and, as a result, the output is optimised compared to other models. Depending on the enhancement in various applications, various learning methodologies and algorithmic models have to be tested and evaluated [5]. Therefore, to develop supervised ML anomaly-based IDS, it is necessary to collect a dataset of traffic with correct labelling, both normal and abnormal. The inclusion of various types of anomalies can allow a validation and evaluation of system effectiveness, considering the real trends in security vulnerabilities [6].

1.1. Motivation

Modern monitoring technology has evolved significantly, as optimised systems allow data to be collected at multiple measurement points, obtaining a large quantity of data that, when processed with artificial intelligence, have obtained deep and meaningful research results [7]. The lack of dedicated IoT datasets with specific protocols and conditions limits the emergence of novel mechanisms to recognise abnormal network behaviour. The detection of traffic anomalies can be improved by the proposal of new datasets representing real systems with wireless sensor networks and light communication, where IoT protocols are modified.

The implementation of models that recognise abnormal circumstances is possible due to the representative knowledge of the environment. These facts are motivated the presentation of CIDAD (available at https://github.com/dad-repository/cidad.git), URL (accessed on 30 October 2022)) a complete and labelled CoAP-IoT dataset based on the reproduction of certain traffic with a sufficient number of samples, specific feature extraction, and mixed anomalous scenarios.

An incident that happened at the Centre for Information and Communications Technology Research (CITIC) [8] served as the inspiration for the scenario design in this article. The temperature sensors initially failed to deliver accurate data to the cooling system due to a software issue that resulted from an erroneous sensor configuration in the data centre. The system failed due to a hardware breakdown brought on by changes in the device temperatures.

In addition, to validate CIDAD for traffic anomaly detection on CoAP-IoT networks, five shallow machine learning models were applied. Indeed, ML cannot work with raw data in suitable conditions; thus, as a preliminary stage, it was necessary to extract and

analyse the representative features and remove redundant information. Consequently, the data had to be pre-processed to be perfectly interpreted by the ML algorithms.

1.2. Contributions of the Work

The main objective of this work is to present a new CoAP-IoT dataset, perfectly labelled, easy to follow and manipulate, and designed for traffic anomaly detection in IoT environments, and present its validation through the application of supervised learning algorithms. The main contributions of this research are:

- An exhaustive overview of existing IoT datasets and their validation using machine learning algorithms for anomaly detection.
- Design and implementation of a semi-synthetic IoT environment to program request and respond COAP messages, introducing alterations of the default behaviour of the CoAP packets to produce anomalies in the traffic.
- An annotated, versatile, easy to follow, and manipulated CoAP-IoT dataset that facilitates tests with supervised learning algorithms.
- Analysis and feature extraction of IoT dataset wireless sensor networks.
- Implementation and data optimisation for binning, handling unbalanced data, feature extraction, and hyperparameter tuning in real environments.
- Application of diverse machine learning techniques and their evaluation using different traditional metrics to validate the selected dataset.

1.3. Paper Organisation

This work is structured as follows: Section 2 presents the most relevant IoT datasets and how they have been used in detecting traffic anomalies using machine learning techniques. Section 3 identifies the architectures used for both the physical and virtual settings. The structures implemented for the exchange of messages are presented in Section 4. An overview of the CIDAD is detailed in Section 5. Section 6 exposes the machine learning techniques used. Section 7 presents the preprocessing of data as well as the feature engineering, to make the data perfectly understood by classifiers. For model implementation, CIDAD was divided into training and test sets. The results of the training split are discussed in Section 8. The assessment of the test set is detailed in Section 9. Finally, a discussion and future research directions are presented in Section 10.

2. Related Work

The emerging use of IoT networks and their relative novelty, along with their security threats, has led to the recent endeavour of building specialised IoT datasets.

This section presents some of the most relevant published public datasets, which were developed specifically in IoT sensor networks with sufficient traces and are perfectly labelled, designed for IDSs using supervised machine learning. As well as this, examples of their implemention for traffic anomaly detection using machine learning are shown.

- N-BaIoT [9]: This dataset is composed of captured traffic from two networks, the first one corresponding to a video surveillance network with internet protocol (IP) cameras with eight different types of attacks that interrupt the availability and integrity of the video links. The second one comes from an IoT network with three computers and nine IoT nodes, one of which is affected with the Mirai botnet malware. This dataset was employed to detect botnet attacks in the IoT by extracting snapshots of network behaviour and using deep autoencoders. In addition, using the mentioned dataset, Kitsune [10] devised a system for network intrusion detection, which learns how to detect online attacks on local networks without supervision using autoencoders. Furthermore, Abbasi [11] exploited the dataset for classification using logistic regression (LR) and artificial neural networks (ANN).
- CCD-INID-V1: Liu et al. [12] created a new dataset to identify traffic anomalies in IoT networks. In addition, they developed a hybrid intrusion detection method that incorporated an embedded model for feature selection and a convolutional neural

network (CNN) for attack classification. The considered method has two models: RCNN, where random forest is combined with a CNN, and XCNN, where extreme gradient boosting (XGBoost) is combined with a CNN. They compared the accuracy of their method with classical ML techniques, such as support vector machine (SVM), K-nearest neighbours (KNN), LR, and naive Bayes (NB). To validate their experiments, they also compared their proposed dataset with the N_BaIoT dataset [9] and the DoH20 dataset [13] to investigate the performance of these learning-based security models and compare their method against conventional ML techniques.

- Doshi et al.: They developed some classifiers to automatically identify attacks in IoT traffic. They demonstrated that considering IoT traffic information in the feature selection process can lead to a high accuracy distributed denial of service (DDoS) detection [14]. A pipeline was assembled to operate on network middleboxes and corresponding devices that may be part of an ongoing botnet. Different classifiers for attack detection were implemented, such as KNN, support vector machine with linear kernel (LSVM), decision tree (DT), random forest (RF) and neural networks (NN).
- All eyes on you: Pahl et al. [15] developed an IoT microservice anomaly detection, where they created and published two separate datasets. The datasets monitored connections between seven different virtual state layer (VSL) service types of light controllers, movement sensors, thermostats, solar batteries, washing machines, door locks, and user smartphones. Additionally, employing this dataset, Hasan et al. [16] proposed an ML system using LR, SVM, DT, RF, and ANN. The proposed models based on ML could recognise and conserve the system when it is in an abnormal state. In training and test stages, the best accuracy results were obtained with RF and ANN..
- Anthi et al.: They presented a dataset based on eight commercially popular IoT nodes as an example of a smart home [17]. They collected benign data for three weeks and malicious data for two weeks from the IoT testbed in five scenarios with six IoT attack categories. They also proposed a three-layer IDS considering NB, Bayesian networks, J48, Zero R, OneR, simple logistic, SVM, multi-layer perceptron (MLP), and RF, generating potential intrusion detection systems.
- SCADA IIoT: For an industrial IoT (IIoT) environment, Lemay et al. presented a synthetic labelled dataset for supervisory control and data acquisition (SCADA) networks [18]. The dataset includes both malicious and non-malicious Modbus traffic packet captures, introducing five types of malicious activities. The exact number of master terminal units (MTUs) and remote terminal units (RTUs) was varied to illustrate the impact of different implementations on real traffic. Detection of traffic anomalies was carried out by Duque et al. in [19], where four different ML algorithms were applied to detect anomalies, e.g., k-means clustering, KNN, SVM, RF, and KNN, achieving the best performance with the SVM model.
- IIoT MODBUS: Another approach in IIoT was developed by Frazao et al. [20], who employed a small automation testbed using MODBUS/TCP. Using both the MODBUS and TCP protocols, the testbed emulates a cyber-physical system process that is managed by a SCADA system. They analysed several attacks in the testbed: ”ping flooding”, ”TCP SYN flooding”, and ”MODBUS query flooding -read holding registers”, which target the programmable logic controller (PLC). In order to validate the dataset, four classifiers were implemented, kNN, SVM, DT, and RF, with the DT classifier achieving the best results.
- UNSW: A group of researchers at the University of New South Wales, Sydney (UNSW) [21], analysed the network traffic of IoT devices to learn their behaviour, implementing the internet engineering task force manufacturer usage description framework (MUD). They presented a labelled dataset with a range of volumetric attacks, such as denial of service (DoS), reflective TCP/UDP/internet control message protocol (ICMP) flooding, and address resolution protocol (ARP) spoofing to IoT devices. They also developed a machine learning system employing two outlier detection techniques, boundary detection and Markov Chain, to determine whether an IoT device is involved in a

volumetric attack or not and the flow that participates in the attack. IoT monitoring was performed via a combination of coarse-grained (per device) and fine-grained (per flow) software-defined networking (SDN) telemetry for each IoT device. The process was executed for various time scales, identifying deviations from expected traffic patterns in device flows defined by the device MUD profile.

- BoT-IoT: [22] designed by the Cyber Range Lab of the Australian Center for Cyber Security (ACCS), is a dataset that generates a mix of normal and botnet traffic through a simulated IoT sensor test bench. The dataset comprises a simulated network of IoT services, which contains some temperature, pressure, and humidity IoT sensors that are connected to the public IoT centre through MQ telemetry transport (MQTT) messages. The dataset contains operating system (OS) and services analyses, DoS, DDoS, data exfiltration, and keylogging attacks. Based on the protocol used, the DDoS and DoS attacks are further organised. To develop their work, the authors applied three ML algorithms, SVM, a recurrent neural network (RNN), and long-short term memory recurrent neural network (LSTM-RNN). Additionally, other authors have exploited the BoT-IoT to develop their experiments, for example, Susilo et al. [23] built an anomaly detection system applying RF, SVM, MLP, and CNN, while Alsamiri-imi et al. [24] implemented NB, RF, MLP, iterative dichotomiser 3 (ID3), adaptative boosting (AdaBoost), quadratic discriminant analysis (QDA), and KNN.
- IoT-NID: The IoT Network Intrusion Dataset [25] was developed by The Hacking and Countermeasure Research Lab. They created various types of network attacks on the Internet of Things environment including Mirai, man in the middle (MITM), DoS, and scanning, among others, which were connected to the same wireless network. The dataset consists of 42 raw network packet files at different time points, using typical smart home devices and combining some laptops and smartphones.
- IoT-23: The Aposemat IoT-23 dataset [26] published by Stratosphere Laboratory, AIC group, FEL, and CTU University, offers a large record consisting of 23 captures of different IoT network traffic: twenty from infected IoT devices and three from real IoT device network traffic, over HTTP, domain name system (DNS), dynamic host configuration protocol (DHCP), Telnet, secure sockets layer (SSL), and internet relay chat (IRC) protocols. Malicious traffic captures were performed for long periods, including various attack scenarios, such as Torii, Mirai, and Gafgyt. To detect traffic anomalies, this dataset was also employed by Thamaraiselvi et al. [27], applying RF, NB, SVM, and DT.

  Furthermore, Aversano et al. [28] evaluated abnormal IoT traffic with a deep learning technique that can distinguish between legitimate and illegitimate IoT traffic as well as diverse forms of anomalies. They integrated five different datasets, including the BoT-IoT [22] presented by Sivanathan et al. [29] for normal traffic coming from various devices; the IoT network intrusion dataset [25], a part of the IoT-23 [26], which mainly involves malicious traffic; and a small automation testbed [20] for industrial control systems.
- ToN-IoT: Moustafa at the Cyber Range Labs of UNSW Canberra [30] created an MQTT dataset, named ToN-IoT, that comprised heterogeneous data sources collected from a realistic and large-scale network, where more than ten IoT and IIoT sensors, including weather and industrial control sensors, were employed to capture telemetry data, presenting both regular and abnormal events. The IoT nodes (for instance, green gas IoT and industrial IoT actuators) communicate using MQTT, and they publish and subscribe to different topics, namely temperature and humidity. Sarhan et al. [31] intended to standardise the techniques to apply them to any dataset. Six ML models, deep feed forward (DFF), CNN, recurrent neural network (RNN), DT, LR, and NB, and three feature extraction algorithms, principal component analysis (PCA), linear discriminant analysis (LDA), and automatic encoder, were applied on three reference datasets, and among them was the ToN-IoT [30].

- MedBIoT: This is another semi-synthetic dataset related to IoT botnet, focused on the detection of botnet attacks such as Mirai, BashLite, and Torii [32]. Developed in the Department of Software Science at Tallinn University of Technology, this dataset was obtained from a network with more than 80 devices, including normal and malicious traffic. The scale extension enables capturing malware spreading patterns that cannot be seen in small-sized networks, thus providing a more realistic environment. KNN, SVM, DT, and RF classification models were implemented, verifying the applicability of the proposed dataset.
- DAD [33]: This is a labelled IoT dataset containing a reproduction of certain real-world behaviours seen from the network and used for the detection of traffic anomalies in IoT sensor networks. The scenario involves four InRows, with four temperature sensors each, connected to a broker that establishes connections through MQTT messages. The dataset presents three types of anomalies, duplication, interception, and modification, on the MQTT protocol payload spread over 5 days. Later, this was validated in [34] applying LR, RF, NB, AdaBoost, and SVM, demonstrating its applications in anomaly detection, where the best classification results were obtained for RF and Adaboost models.
- MQTTset: Vaccari et al. presented MQTTset [35], a dataset which was constructed from capturing traffic from a diverse IoT network that simulated a smart home/office/building environment. The testbed employed the MQTT protocol at the application layer, introducing denial-of-service attacks, flood attacks, MQTT publish floods, SlowITe, malformed data, and brute force authentication. Intrusion detection systems using ML were implemented using DT, RF, MLP, NN, and Gaussian NB on balanced and unbalanced data. The authors demonstrated the ability of intrusion detection systems in IoT-MQTT environments to identify malicious behaviour.
- IoT-RPL: Dhifallah et al. at Laboratoire Hatem Bettaher (IRESCOMATH) [36] proposed a new IoT-RPL dataset. They built a real-time IoT network infrastructure, following a simulation/emulation approach where IoT nodes were simulated with Cooja. The IoT-RPL dataset presents significant data link (6LoWPAN and IEEE 802.15.4), network (IPv6 and ICMPv6), transport (UDP), and application (CoAP) protocols. Four parasite signal distribution nodes (hello food attack, blackhole attack, decreased rank attack, and version number attack) were included in the dataset. For intrusion detection, five machine learning models (SVM, DT, RF, KNN, and MLP) were compared, achieving accuracy rates of 99% in most cases.
- IoT-Flock: Ghazanfar et al. [37] built a real-time IoT smart home system dataset using an IoT traffic generator tool that could produce normal and abnormal IoT traffic over a real-time network by means of just a physical machine. The given dataset contains four types of environmental monitoring sensors (temperature, humidity, light, and motion) and communicates using MQTT protocol through a wireless local area network (WLAN). In order to show the utility of the dataset for creating an IoT security solution, it was used with three common ML models (NB, RF, and KNN), achieving high detection rates.
- SDN-IoT: Bhayo et al. [38] developed a secure IoT framework based on a software-defined network. This framework is capable of detecting vulnerabilities in IoT devices and abnormal traffic created by IoT devices taking into account the number of IP sessions and analysing IP payloads. In this way, it can easily detect DDoS attacks in the network by studying different parameters. The authors presented IoT nodes in different scenarios. A large amount of traffic is generated from an exposed node, and abnormal traffic is detected and notified by the SDN controller. Given the results reported, the proposed framework detection accuracy for DDoS attacks is high, from 98% to 100%. These results were obtained in the early stage of the attacks and have a low false-positive rate. Despite that, due to the novelty of the proposed framework, it has not been tested for ML IDS.

- IoTID20 [39]: The testbed for the IoTID20 dataset is a combination of IoT devices and interconnecting structures. The attacks are carried out on two devices that act as victims. The dataset contains eight types of attacks, belonging to DoS, Mirai, MITM, and scan. The ML models used for anomaly detection are SVM, GaussianNB, LDA, LR, DT, RF, and ensemble classifiers. The best detection rates achieved were with RF and ensemble classifiers.
- MedBIoT [32]: Developed at the Department of Software Sciences at Tallinn University of Technology, this is another semi-synthetic IoT dataset obtained from a network with eighty emulated devices and three physical devices. The malicious behaviour of the dataset is generated by the deployment of three prominent botnet malware within the controlled environment: Mirai, BashLite, and Torii. In order to verify the suitability of the IoT behavioural dataset, kNN, SVM, DT, and RF classification models were implemented.
- X-IIoTID [40]: This dataset is implemented using the Brown-IIoTbed [41] testbed developed at the University of New South Wales (UNSW). It consists of a variety of M2M, machine to human (M2H), and human to machine (H2M) connectivity protocols, sensors, actuators, various mobile and IT devices, means of access, application programming interfaces (API), and states that are implemented at the three levels of an IIoT system. It contains a wide variety of attacks on different layers of the system. The data include normal and abnormal traffic, captured at various hours of the day for four non-continuous months. For dataset validation, DT, NB, kNN, SVM, LR, DNN, and GRU algorithms were used. The best results were achieved with DT in all cases, achieving an accuracy of 99.54% for binary classification.
- CIC IoT Dataset 2022 [42]: Simulating smart home activity, this dataset was created from 60 devices in an isolated IoT sensor network at a laboratory in the Canadian Institute for Cybersecurity (CIC). The IoT network includes WiFi, ZigBee, and Z-Wave devices. In the experiments, six different scenarios were investigated, containing two different types of attacks: flood denial-of-service attack and real time streaming protocol (RTSP) brute-force attack. The IDSs were developed using 12 different Gaussian classifiers, NB, DT, LDA, AdaBoost, Ridge, Perceptron, passive-agressive, XGBoost, kNN, RF, LSVC, and SGD, obtaining the best results with XGBoost (98.6% accuracy) and the tree-based classifiers, DT and RF (98.5% accuracy), but with a lower prediction time on the tested dataset. They finally decided on an RF classifier because it is computationally efficient, does not overfit, and the performance can be improved with data and feature selection.

To summarise, Table 1 outlines the mentioned datasets, emphasising the type of IoT network used, the anomalies they contain, and some works that validate their use in detecting anomalies using ML. Most of the described IoT datasets are based on smart home environments or video surveillance cameras in non-specific environments. Three datasets handle environments or traces on IIoT platforms, and only a few of them use dedicated protocols. Some of them introduce networks of wireless sensors, such as systems for measuring temperature, humidity, movement, etc., but none of the aforementioned datasets provide networks of lightweight temperature sensors in a data centre with an acceptable amount of traces in the IoT, with a mixture of anomalies, based on a CoAP protocol. Consequently, we found the need to generate a dataset that would adjust to the requirements of the described environment. Thereby, the design, implementation, and application of ML for anomaly detection to a dataset under the described conditions is presented in this article.

Table 1. IoT dataset review summary considering applications in ML-IDS.

Dataset	IoT Network	Anomaly	ML Validation
N-BaIoT [9]	IP cameras, PCs, and IoT devices	Mirai botnet and BASHLITE malware	Deep autoencoders [9]. Encoders [10]. LR and ANN [11]. RCNN, XCNN [12].
CCD-INID-V1 [12]	Smart sensors in an IoT network	ARP poisoning, ARP DoS, UDP Flood, Hydra Bruteforce with Asterisk protocol, SlowLoris	RCNN, XCNN, KNN, NB, LR, SVM [12].
Doshi et al. [14]	Smartphones and laptops	DDoS, Mirai	KNN, LVSM, DT, RF, NN [14]
All eyes on you [15]	Smart home and user smartphones	Anomalous timestamps, DoS	grid-based, k-mean [15]. LR, SVM, DT, RF, ANN [16]
Anthi [17]	Smart home	DoS	NB, Bayesian Network, J48, Zero R, OneR, Simple Logistic, SVM, MLP, RF [17].
SCADA IIoT [18]	Modbus traffic packet captures	Metasploit, fingerprinting, unauthorised packets	SVM, RF, KNN, k-means [19].
IIoT MODBUS [20],	Small automation testbed using MODBUS/TCP	Ping flooding, TCP SYN flooding, and MODBUS Query flooding.	KNN, SVM, DT, RF [20]. Autoencoder [28]
UNSW [21]	Smart home	DoS, reflective TCP/UDP/ICMP flooding, and ARP spoofing	Boundary detection and Markov Chain [21].
BoT-IoT [22]	Smart Home (MQTT)	DoS, DDoS, data exfiltration, keylogging	SVM, RNN and LSTM-RNN [22]. Autoencoder [28]. RF, SVM, MLP, CNN [23]. NB, RF, MLP, ID3, AdaBoost, QDA, KNN [24].
IoT-NID [25]	Smart home, laptops, and smartphones	Mirai, MITM, DoS, scanning, etc	Autoencoder [28]
IoT-23 [26]	Smart home (real IoT devices)	Mirai, Torii, and Gafgyt	RF, NB, SVM, DT [27], autoencoder [28]
ToN-IoT [30]	Weather and industrial control sensors (MQTT)	Several normal and cyber attack events from IoT networks	DFF, CNN, RNN, DT, LR, NB [30]. Autoencoder [31]
MedBIoT [32]	IoT network (i.e., fans, locks, light bulbs and switches).	Mirai, BashLite, Torii	KNN, SVM, DT, RF [32].
DAD [33]	Temperature IoT sensors (MQTT)	Duplication, interception, and modification of packets.	LR, RF, NB, AdaBoost, SVM [33].
MQTTset [35]	Temperature, humidity, motion sensors (MQTT)	Flooding DoS, MQTT Publish flood, SlowITe, malformed data, and brute force authentication	DT, RF, MLP, NN, and Gaussian NB [35].
IoT-RPL dataset [36]	Simulated nodes	Hello food, blackhole, decreased rank, and version number	SVM, DT, RF, KNN, and MLP [36]
IoT-Flock [37]	Smart home (MQTT/CoAP)	MQTT Packet Crafting Attack, MQTT Publish Flood, CoAP Memory Leak and Attack CoAP Segmentation Fault Attack	NB, RF, and KNN [37]
SDN-IoT framework [38]	Simulated network nodes	DDoS	Still not implemented
IoTID20 [39]	smart home	Flood Syn, HTTP and UDP, ARP Spoofing, brute force	SVM, Gaussian NB, LDA, LR, DT, RF, and ensemble clasifiers

Table 1. *Cont.*

Dataset	IoT Network	Anomaly	ML Validation
MedBIoT [32]	IoT devices	Mirai, BashLite, Torii	KNN, SVM, DT, and RF.
X-IIoTID [40]	IIoT	Weaponisation, exploitation, lateral movement, exfiltration, tampering, crypto-ransomware, ransom denial of service	DT, NB, KNN, SVM, LR, DNN, and GRU
CIC IoT Dataset 2022 [42]	WSN	Flooding and brute force RTSP	NB, DT, LDA, AdaBoost, Ridge, Perceptron, Passive-Agressive, XGBoost, KNN, RF, LSVC, and SGD

3. Scenario

In order to obtain a dataset that can be used to detect traffic anomalies, a virtual scenario based on a real data centre temperature sensor network has been designed and implemented. First, this section presents the structure and actual operation of the sensor network from the real data centre. This includes the interaction, distribution, and localisation of the sensor devices. Secondly, the implementation of the virtual scenario and the elements required for the construction of an IoT sensor network are described.

3.1. Physical Architecture

To build a real environmental dataset, data were obtained by modelling observations from authentic temperature sensors in the Centre for Information and Communications Technology Research (CITIC) Data Center [8]. The data centre structure consists of three elements with sensors: the racks, the power strips (PDU), and the refrigeration machines (InRow). In the proposed scenario, only the sensors of the InRows will be considered. The other sensors in the data centre maintain static temperature values; therefore, they are not susceptible to analysis. Each InRow has control over four sensors: (a) unit entering fluid temperature (TFEU), which measures the temperature of the input fluid to the InRow unit, (b) unit leaving fluid temperature (TFSU), which measures the temperature of the fluid leaving the InRow back to the outside cooler, (c) unit return air temperature (TAR) unit, which measures the temperature of the air entering the InRow air conditioning unit of the closed hot corridor, and (d) unit supply air temperature (TAS), which measures the temperature of the air coming out of the InRow into the cold corridor.

Based on how the CoAP protocol works, a client node can command each InRow node sending a CoAP packet. Every sensor is identified by a uniform resource identifier (URI) address (ex. www.inrow15.gal/1) , where the resource indicates the InRow, and the number that follows the slash corresponds to its sensor: (1) TAS, (2) TAR, (3) TFEU, and (4) TFSU. Then, when performing virtualisation, there will be a virtual machine for each InRow, with the host machine acting as a client.

3.2. Virtual Scenario Description

To generate the dataset, a virtual infrastructure was deployed using the ESXi 6.5 hypervisor. Through the vSwitch capacity provided by this hypervisor, an isolated Ethernet network, called the IoT network, hwas configured, so that the infrastructure is not interfered with by external or unwanted traffic. Five virtual machines with the Ubuntu 20.04 operating system were deployed in this network, four of which represent the four CoAP servers, called inRow, which in turn have four sensors each, while the fifth virtual machine implements the CoAP client functions and therefore implements the messages sent to each of the four sensors of each of the four CoAP servers, thus performing the function of a system monitor. Figure 1 shows the scenario with different virtual machines that contain a client and four CoAP servers.

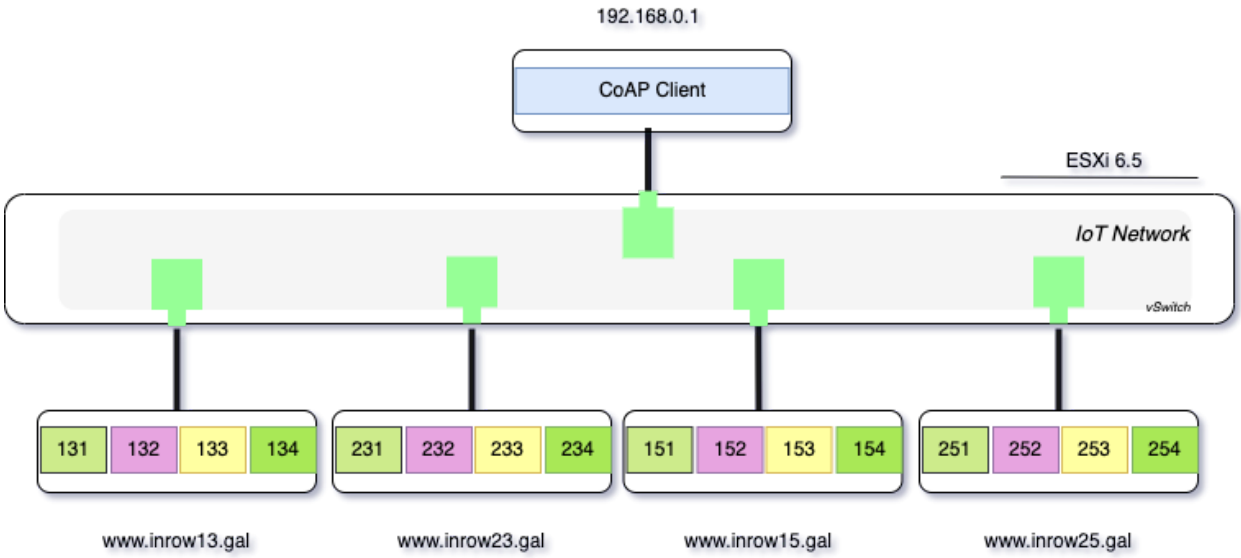

Figure 1. Virtual scenario, including a CoAP client and four CoAP servers (InRows).

The CoAP client message, sent by the client virtual machine, contains a CoAP packet built using the Scapy for Python tool [43], where the IP and UDP protocols are concatenated and the token value is randomly generated. Finally, the CoAP packet is sent asynchronously using the send method from the Scapy library. In each of the four virtual machines that simulate the operation of the InRows, a server CoAP that listens for messages from the CoAP client was implemented, using the sniff method of the Scapy library and performing a filtering process. For each message received, a reply is sent, which includes the temperature value for each node. The temperature values were calculated based on the time series implemented in [33].

Each of the samples generated by the sensors is sent every five minutes to the client through a CoAP message that contains its node identifier as a clientID (using URI-Host and URI-Path values). The capture of the traffic exchange between the nodes is executed on the client. The connection information is stored to be latter tagged in the dataset via Scapy. Most abnormal packets are flagged on the server, except those that belong to the interception anomaly.

4. Dataset Generation

The CIDAD was generated from a seven day monitoring of the situation that occurs daily in a simulated environment. Five of the seven days present anomalies injected in one of the InRows on the payload of the CoAP message. Traffic labelling is performed at the packet level, where abnormal traffic is statistically different from normal traffic. Three different scenarios of anomalies are presented in the dataset, where the behaviour of the node can be affected as follows:

- Interception: the duplication anomaly consists of arbitrarily removing packages.
- Modification: this anomaly consists of modifying the temperature of the sensors to a very low value. This causes the system to stop cooling, resulting in overheating of the devices.
- Duplication:this anomaly sends more tokens than the packets initially planned.

Dataset Setup

One of the most promising protocols for small devices is CoAP. It is intended to be used for web transfer with restricted devices and networks, such as scenarios with power constraints or dissipating lines. It is created to communicate machine to machine applications, according to a request–response model. It is capable of discovering both services and

resources. Moreover, it takes into account very important web ideas, such as the unified resource identification (URI) and media types. For this reason, it can communicate with HTTP; however, at the same time, it allows for multicast communications, very slight overhead, and plainness for limited scenarios. CoAP employs UDP at the transport layer, so it communicates with unreliable and connectionless datagrams [2]. The UDP port is 5683. The structure used in this scenario for the exchange of messages in each of the roles used by every host is described below:

- Client: The client sends CoAP request messages to InRows (servers). These messages are of type NON (non-confirmable) and do not require a confirmation ACK message (the network will be less overloaded and the devices will consume fewer resources when processing and sending these messages). The request sent by the client has the following structure:
  - Version: the value of this field will be 1 (01).
  - Type: since the messages will be of type NON, the value will be 1 (01).
  - Code: the messages that are sent from the client requesting a resource from the server will be of the GET method. Therefore, the value of the code field will be 1 (0.01).
  - Message ID: this is calculated according to the message sent; the message id increases according to each message and follows a chronological order.
  - Token: each token is a unique and randomly generated field of the message. The size of this token is 4 bytes, the minimum size recommended.
  - Options: the options used are URI-Host and URI-Path, able to correctly identify the resource on the corresponding server. We will also add the Accept option with value 0, which indicates that the format of resource representation that the client accepts is plain text (text/plain).
- Server: the InRows will act as servers, sending back responses to the client. The scenario includes four servers: InRow (1) www.inrow13.gal, (2) www.inrow15.gal, (3) www.inrow23.gal, and (4) www.inrow25.gal. Each InRow has four sensors, whose resource identifier or URI-Path in CoAP will be 1, 2, 3, and 4. The responses sent by the server have the following structure:
  - Version: the same as the original message (01).
  - Type: the same type as the original message (type NON, 01).
  - Code: the response code is 2.05 Content, which includes a representation of the requested resource.
  - Message ID: this is generated randomly for each reply message.
  - Token: the same as the request message.
  - Options: the URI-Host and URI-Path options are the same as the request. The Content-Format option is also added, which specifies the representation format of the object included in the payload field of the message (in our case, the value will be 0, text/plain).
  - Payload: this field includes a representation of the object, which is the temperature measured by the sensor in this specific scenario.

The Python code shown below indicates how the client constructs the message for each of the InRow nodes:

```
for server,ip in self.servers:
    for sensor in self.sensors:
        coapPackage = CoAP(ver=1, type=1,
        code=1, msg_id =
        self.create_msgID,
        token = self.create_token,
        options = [(''Uri-Host'', server),
        ''Uri-Path'',sensor),
        (''Accept'', b''\x00'')],
```

```
paymark = b‘‘\xff’’)
packet = (IP(dst=ip)
/ UDP(sport=5683, dport=5683)
/ coapPackage)
```

Next, the code shown below indicates the reply code from the server to the client. The client has an IP address “192.168.0.1”:

```
if req.haslayer(CoAP) and
req.getlayer(IP).dst == ‘‘192.168.0.1’’:

    respCoap = (CoAP(ver=coap.ver,
    type=coap.type, code=69,
    msg_id = self.create_msgID(),
    token=coap.token,
    options = [coap.options[0],
    coap.options[1],
    (‘‘Content-Format’’, b‘‘\x00’’)],
    paymark=coap.paymark) /
    Raw(load=
    self.calculate_replyText(coap)))

    respIP = (IP(dst=ip.src) /
    UDP(sport=udp.dport,
    dport=udp.sport) / respCoap)
```

5. Dataset Analysis

The anomaly traffic is measured from InRow 13 with IP source 192.168.0.2 during one week, as shown in Table 2.

Table 2. Dataset anomalies.

Day	Modification	Interception	Duplication
Monday 27 July 2020			
Tuesday 28 July 2020		✓	
Wednesday 29 July 2020			✓
Thursday 30 July 2020	✓		
Friday 31 July 2020	✓	✓	✓
Saturday 1 August 2020	✓	✓	✓
Sunday 2 August 2020			

All the anomalies presented are performed on the temperature message packets. During the interception anomaly, the data text line packets from the InRow 13 sensors are not sent to the client. For the modification anomaly, all the temperature measurements are adjusted to the same value, placing them at 11.56 ^{o}C regardless of which sensor they belong to. When a duplication anomaly occurs, additional packets are sent to the client.

As aforementioned, the sensor sends data to the client every 5 min. The CoAP data text line packets correspond to the messages from the sensors. Therefore, under normal conditions, there should be 288 daily CoAP data text line packets. Anomalous duplication and modification packets must be reflected in the number of labelled daily packets. In the case of interception, the packets are not shipped from the host; thus, these do not exist in the dataset and cannot be labelled as an anomaly. However, the request messages sent by the client, in this case, are labelled as anomalies.

5.1. General Dataset Description

Taking the type of communication used in the proposed scenario into account, a series of important features are selected in the sending of CoAP messages, such as the amount

of source and destination bytes, the number of source and destination packets sent, the number of packets for each protocol, and the amount of abnormal and normal packets per day.

The CIDAD has a total of 88,238 packets, and recalling that the CoAP protocol goes over the UDP, 73% of all the traffic corresponds to UDP packets. Detailed information is presented in Table 3. It contains ARP, IP, and IPv6 traffic. However, the number of packets sent in IPv6 represents only 0.95% of the total packets and 26% of the ARP traffic. All IPv6 packets correspond to ICMPv6 packets, while IPv4 packets contain UDP and ICMP datagrams. All the anomaly packets belong to UDP protocol. Thus, 71.62% of UDP packets are normal, while just 1.20% of all the traffic is labelled as an anomaly. It is important to note that the packets from the sensor that are intercepted do not appear in the dataset because they do not exist anymore. As a consequence, the request packets sent by the client that have no response are labelled as anomalies.

Table 3. Dataset statistics.

Day	Src Bytes	Dst Bytes	Src Packets	Dst Packets	ARP Protocol	IP Protocol	IPv6 Protocol	UDP Protocol	Normal Packets	Anomaly Packets
Monday	490,344	331,776	7040	4608	2312	9216	120	9216	11,648	0
Tuesday	529,942	331,818	7737	4609	3105	9123	118	9120	12,250	96
Wednesday	537,736	331,818	7836	4609	3105	9219	121	9216	12,349	96
Thursday	516,302	330,666	7464	4593	2702	9235	120	9232	11,961	96
Friday	633,196	331,902	9442	4611	4707	9225	121	9216	13,669	384
Saturday	633,056	331,902	9440	4611	4707	9225	119	9216	13,667	384
Sunday	489,796	331,776	7030	4608	2304	9216	118	9216	11,638	0
Total	3,830,372	2,321,658	55,989	32,249	22,942	64,459	837	64,432	87,182	1056

Knowing how CoAP carries out the exchange of messages and assuming the request/response connection, in normal instances, there should be homogeneity in the number of packets sent from each IP address source, as well as the number of bytes sent. Therefore, alterations in data uniformity are an indication of an anomaly.

The four sensors belonging to the same InRow share the same IP address. Due to the type of architecture presented, the sensors of the IoT network only send packets to the client and do not create connections between them. A large portion of packets, about 32,249, corresponds to requests made by the client. All abnormal traffic is involved with InRow 13 with the IP address 192.168.0.2, over IPv4.

The lack of homogeneity in the packets sent by the sensors represents abnormal network behaviour. The frame length determines the number of bytes sent per packet. Taking into account the proposed scenario and the homogeneity that must be presented, given the nature of the CoAP protocol, changes in the density function of the frame length imply anomalies in the packets sent.

5.2. UDP/CoAP Description

The CIDAD presents the UDP protocol at the transport layer and the CoAP at the application layer. The CoAP makes use of two message types, requests and responses, using a simple, binary-based header format. In the dataset, there are two types of CoAP messages; the packets marked as CoAP represent those requests made by the client and the CoAP data text line are the responses sent by each of the sensors. Therefore, the messages with the temperature data from the sensors will be housed in the payload of the CoAP data text line messages.

Table 4 shows the number of packets for each type of CoAP message. As mentioned above, all anomaly packets are over UDP messages. The packets labelled as anomalies in the CoAP protocol on Tuesdays, Fridays, and Saturdays correspond to packets sent by the client that have not received a response and correspond to an interception anomaly. This anomaly is also reflected in the absence of a data text line CoAP. The modification and

duplication anomalies are labelled over the CoAP data text line. In the case of Wednesday (modification anomaly), the number of packets remains unchanged, while on Thursday (duplication anomaly), there is a considerable increase in the number of packets. In cases where there is a mixture of anomalies (Friday and Saturday), the number of packets and their distribution is not an accurate indicator of the presence of anomalies.

Table 4. UDP protocol composition.

	CoAP		CoAP Data Text Line	
Day	**Normal**	**Anomaly**	**Normal**	**Anomaly**
Monday	4608	0	4608	0
Tuesday	4512	96	4512	0
Wednesday	4608	0	4512	96
Thursday	4592	0	4544	96
Friday	4512	96	4320	288
Saturday	4512	96	4320	288
Sunday	4608	0	4608	0
Total	31,952	288	31,424	768

As a consequence of the simplicity of the structure handled by the CoAP protocol, the network presents a low, uniform, and homogeneous exchange of packets. This fact makes it necessary to find other types of features that allow to model network behaviour as normal or abnormal.

All InRow sensors have the same IP address, but the URI-host field allows to determine the InRow to which it belongs, and the URI-path field allows to identify the sensor. The network has a total of 16 sensors and each sensor sent 288 packets per day to the client. The absence or presence of additional packets also gives us an initial indicator of the type of anomaly the InRow presents that day. The first evidence of removed packets in this kind of lightweight environment could be the lack of uniformity in traffic. Indeed, the packet reduction presented on Tuesday at InRow 13 is a strong indicator of the presence of an interception anomaly, in contrast to the excess of packets on Thursday, indicating a duplication anomaly. In the case of days in which there is a mixture of anomalies, such as Friday and Saturday, the presence of duplication and interception anomalies will not affect the homogeneity in the number of packets sent per day, but the number of anomalous packets will increase.

6. Machine Learning Techniques

The aim of a machine-learning-supervised approach is to train an algorithm using a dataset for which we know the result. From this data, the algorithm "learns" and can then in the future make decisions about values for which the outcome is not known [44]. There are many machine learning methods, some are more or less flexible, and shallow models are simpler since they have a relatively small estimation range. The implementation of the algorithms was executed using the scikit-learn tools for classification in Python [45].

During the training phase, there is a process called hyperparameter optimisation or tuning. It consists of trying several models with different combinations of values and then comparing the model performance to choose the best one according to a predefined metric on a cross-validation (cv) set. The principle of grid search is exhaustive searching [46].

To perform the experiment, five machine learning models were selected in order to validate the dataset to be employed to detect traffic anomalies in CoAP-IoT networks: logistic regression, naive Bayes, random forest, AdaBoost, and support vector machine. Initially, logistic regression with the liblinear solver, tuning $L1$, $L2$, and C parameters to address overfitting, was applied.

To check or visualise the performance of a classification problem, one of the most important evaluation metrics is the AUC (area under the curve) ROC (receiver operating characteristics) curve. The ROC is a probability curve and the AUC represents the degree

or measure of separability. It tells us to what extent the model is capable of distinguishing between classes. The higher the AUC, the better the model is at predicting 0 classes as 0 and 1 class as 1. A mathematical explanation of the metric can be found in [47].

The type of classification performed on experiments is binary; thus, the Bernoulli naive Bayes was employed, using a single parameter, α, for optimisation. For random forest classification, the parameters considered were the n_estimators and the max_features. The n_estimators parameter suggests the number of trees established in the RF model. The $max\_features$ parameter can take the values of $sqrt$ and $log2$. It suggests the number of features to deal with when searching for the best split. The fourth model selected was AdaBoost, using the decision tree classifier as a base estimator. The hyperparameters selected for tuning AdaBoost, considering a trade-off between them, were n_estimators and learning _rate. Finally, a crucial hyperparameter for a linear support vector machine (SVM) model is the regularisation penalty, C, which can severely affect the resulting shape of the regions for each class.

7. Data Preparation

This work was developed by adapting the the cross-industry standard process for data mining (CRISP-DM) [48]. The first step in a machine learning pipeline involves all the techniques adopted to clean the dataset, reduce their dimensionality, and remove noise. The algorithm performance can be adversely affected when using raw datasets. Some features can determine the general behaviour of the sample, while others simply do not provide additional information. Therefore, it is important to have a clear view of a dataset and reduce or select the number of relevant features.

After the analysis carried out in Section 5, the following features were initially considered:

```
frame.len,frame.time_epoch,ip.dst,
ip.id, ip.len, ip.src, ipv6.dst,
ipv6.src, ipv6.flow, udp.length,
udp.time_delta, udp.time_relative,
udp.checksum_d,  temperature, weekday,
protocol_IoT, coap_sensor, label
```

The implementation of the selected machine learning algorithms requires numerical arrays as input data, so it is necessary to transform categorical variables into numerical ones. This process is known as discretisation. In this approach, the features are encoded using a "dummy" encoding scheme. This creates a binary column for each category and returns a sparse matrix or a dense array (depending on the sparse parameter). After mapping symbolic attributes to numeric values, if there is significant variance, scaling of the feature values is required. Feature scaling is achieved through mean normalisation.

In this case, the dimensionality of the input dataset is high and so is the complexity of every related machine learning algorithm. Feature selection is the process in which each characteristic is evaluated to determine the ones that effect the outcome within the dataset. Its purpose is to reduce high-dimensional data and maintain or improve precision. The recursive feature elimination (RFE) algorithm excludes properties and calculates their performance recurrently [49]. For the estimation of the RFE in CIDAD, a DT classifier with a cv k-fold of 5 divisions was used as the base model. The feature importance ranking is shown in Figure 2, and each colour represents each of the iterations.

The best accuracy was achieved adopting nine features: temperature, hora_cos, hora_sin, ip.dst_192.168.0.2, ip.src_192.168.0.2, weekday_Friday, weekday_Saturday, and weekday_Thursday.

Unbalanced data are a common problem in network intrusion systems, where the important cases requiring detection represent a very small portion of the data. This situation is clearly presented here (see Table 3), where abnormal packets represent 1.2% of the total packets. When learning extremely unbalanced data, there is a significant probability that a selected sample contains few or even none of the minority class. All of this results in an algorithm with poor performance in predicting the minority class. An alternative to

mitigate misclassifications in machine learning algorithms is using stratified k-fold cv. This technique generates partitions in the data, maintaining the balance of the samples between the classes during the training of multiple models. Another mechanism that permits a reduction in the imbalance is the use of sampling techniques. The synthetic minority oversampling technique (SMOTE) is a function that allows handling cases of unbalanced classification by randomly subsampling the majority class. Instead of oversampling the minority class with existing samples, it creates instances of synthetic minority classes to increase the current data and try to avoid overfitting [50,51].

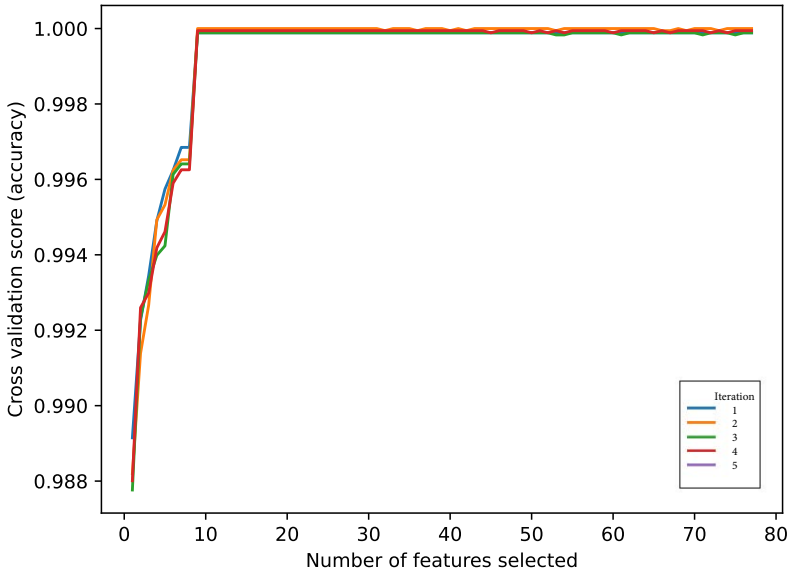

Figure 2. Accuracy versus variable importance.

8. Models Training

As a good practice, the CIDAD was split into training and test sets, with a training subset of 80%. For the evaluation of the models, a nested stratified k-fold cv was chosen with an internal and an external loop. Both loops were adjusted with k = 5. Additionally, to guarantee balanced samples, SMOTE was applied in the internal loop using five k-neighbour. The minority class is over-sampled at about a ratio of 0.4, and the majority class is under-sampled at a ratio of 0.5. This section outlines the process of hyperparameter tuning on the machine learning models over the remaining training data, computing ROC AUC scores. The results presented correspond to the average value of the inner cv, and the best parameters will be chosen to be used in the test stage.

The first model to consider was logistic regression used with a liblinear solver. The penalty hyperparameter was adjusted to $L1$ and $L2$. To tune the C parameter, a sweep over the values of 0.001, 0.1, 1, 100, and 1000 was carried out. The best model was obtained with an $L2$ penalty and a C value of 0.1 and an ROC AUC score of 0.984. The global mean over all iterations was 0.9845, with a standard deviation of 1.013×10^{-3}. Normally, a better operation is expected at high values of C; however, as can be seen in Figure 3, the variation in the parameter C does not significantly affect the performance of the model with an appropriate choice of penalty.

The tuned hyperparameters that offer the best results for LR, in all cases, are a C equal to 0.1 and a penalty $L2$, as shown in Table 5. This table presents the parameters that obtain the best average score in each internal cv and its standard deviation, among all the possible combinations tested. The parameters selected here are those used in the evaluation of the model in the test split.

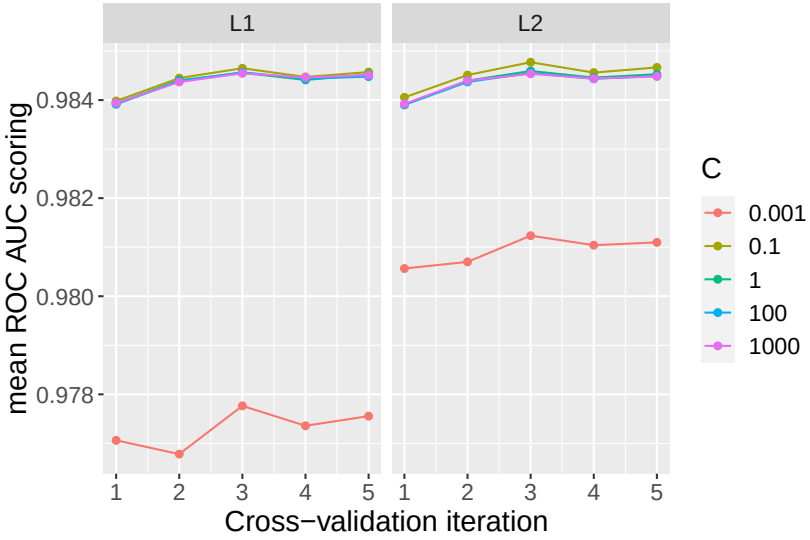

Figure 3. Logistic regression mean score by iteration.

Table 5. Best logistic regression model by iteration.

cv	*C*	Penalty	Mean	SD
1	0.1	l2	0.9841	1.28×10^{-3}
2	0.1	l2	0.9845	0.66×10^{-3}
3	0.1	l2	0.9848	1.01×10^{-3}
4	0.1	l2	0.9846	1.27×10^{-3}
5	0.1	l2	0.9847	0.82×10^{-3}

To identify the optimal composition of hyperparameters for the Bernoulli naive Bayes model, a sweep for the values 0.01, 0.1, 0.5, 1, and 10 in the parameter α was performed. The overall mean ROC AUC value was 0.9398, with a standard deviation of 7.35×10^{-3}. The best individual score for this model was 0.9596, with α equal to 10. Figure 4 shows the average results obtained over the selected values of α in the inner iterations. As a logistic regression, naive Bayes does not present a linear relationship between values of α in each iteration.

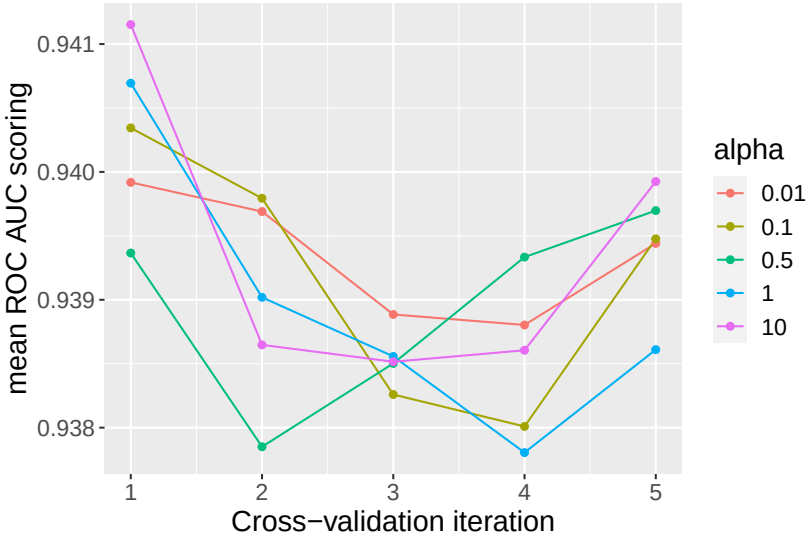

Figure 4. Naive Bayes mean scoring by iteration.

An average of the values obtained in the internal iteration in each of the parameter combinations is calculated for the selection of the parameters that will be used in the model. The best mean in the inner cv is adopted and defines the parameter that will be used in the test split. Table 6 presents the results of this process for the NB model.

Table 6. Best naive Bayes model by iteration.

cv	α	Mean	SD
1	10.0	0.9412	9.39×10^{-3}
2	0.1	0.9398	4.90×10^{-3}
3	0.01	0.9389	8.72×10^{-3}
4	0.5	0.9393	6.93×10^{-3}
5	10.0	0.9399	6.81×10^{-3}

To perform RF, the hyperparameters were tuned using *sqrt* and *log*2 as max_features, and tested with 100, 200, and 300 trees. This model obtains a score of 1 on average over all iterations. Figure 5 presents the average AUC ROC results by iteration. The best performance found among all the models at the training stage is obtained by random forest.

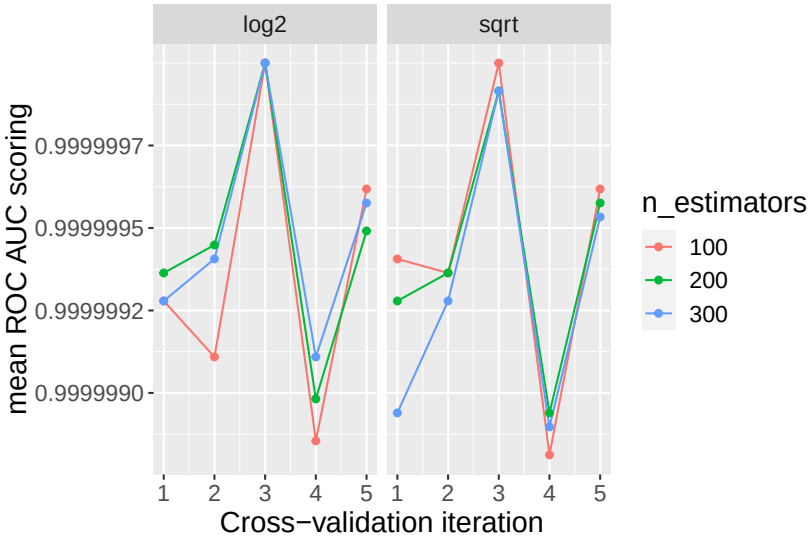

Figure 5. Random forest mean score by iteration.

In tree-based models, the number of trees and features can affect the time and performance of the model during its execution. For this dataset, many combinations of parameters presented perfect scores; however, those that achieved the same performance with fewer trees were selected. Table 7 shows the chosen parameter combinations that perform perfectly for each inner iteration.

Table 7. Best random forest model by iteration.

cv	max_features	n_estimators	Mean	SD
1	sqrt	100	1	9.89×10^{-7}
2	log2	200	1	7.54×10^{-7}
3	sqrt	100	1	0
4	log2	300	1	1.39×10^{-6}
5	log2	100	1	6.62×10^{-7}

On the other hand, the AdaBoost classifier employs a DT base estimator initialised with a max_ depth = 1. In this model, tuning is performed on the values of 500, 1000, and

2000 trees. The learning rate values tested were 0.001, 0.1, 0.5, and 1. The overall mean AUC ROC obtained was 0.99, with a standard deviation of 4.04×10^{-4}.

AdaBoost's performance can be improved by increasing the learning rate parameter and the number of selected trees, as shown in Figure 6. Even so, in the ideal scenario, high accuracy values are obtained with low learning rates and less trees in the classifier.

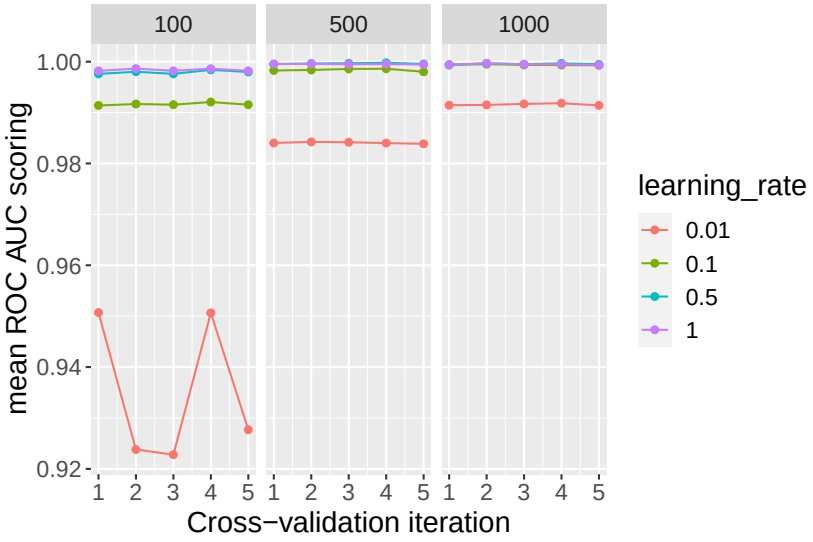

Figure 6. AdaBoost mean score by iteration.

Table 8 presents the parameters that achieve the highest average in internal cv for the AdaBoost model. The best parameter frequently obtained is a learning rate of 0.5 using 500 n_estimators.

Table 8. Best AdaBoost model by iteration.

cv	Learning Rate	n_estimators	Mean	SD
1	1	500	0.99	4.60×10^{-4}
2	0.5	1000	0.99	3.96×10^{-4}
3	0.5	500	0.99	3.87×10^{-4}
4	0.5	500	0.99	3.38×10^{-4}
5	0.5	500	0.99	4.39×10^{-4}

Finally, in SVM, the penalty parameter *C* defines how much error is bearable and can regulate the trade-off between the decision boundary and the misclassification term. The parameter *C* was varied between 0.1, 1, 5, and 10. The linear SVM achieved a mean ROC AUC score of 0.98, with a standard deviation of 1.04×10^{-3}.

Figure 7 shows the averages obtained in each of the internal iterations for the SVM model. For this dataset, the variations in the parameter *C* do not improve the performance of the model. As can be seen in Table 9, the configurations of the parameter *C* that obtain the best means are 10 and 1, with an ROC AUC value of 0.98.

Table 9. Best support vector machine model by iteration.

cv	*C*	Mean	SD
1	10	0.9838	1.39×10^{-3}
2	1	0.9843	7.22×10^{-4}
3	10	0.9845	1.04×10^{-3}
4	1	0.9844	1.24×10^{-3}
5	1	0.9844	8.34×10^{-4}

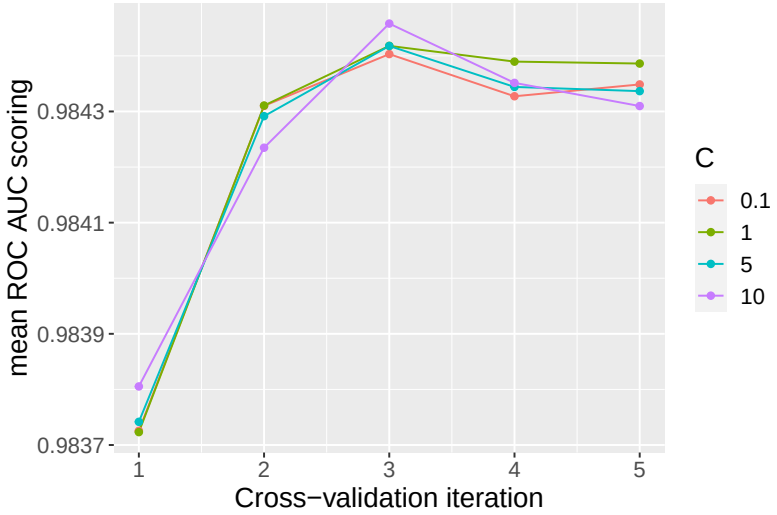

Figure 7. Support vector machine mean score by iteration.

9. Results and Model Comparison

A classification task can be evaluated in many different ways to accomplish specific objectives. For binary classification problems, the assessment of the optimal performance of the algorithm can be defined based on a confusion matrix. A confusion matrix compares the values of the current target with those predicted by the machine learning model. In practice, accuracy is the most widely used evaluation metric either for binary or multi-class classification problems. Through accuracy, the quality of the produced approach is evaluated based on the proportion of correct predictions over total observations. Instead of accuracy, F1 is also a good discriminator and performs better than the latter in optimising binary classification assignments. In contrast, precision and recall present a single evaluation task (positive or negative class, respectively). This section presents the results of four traditional metrics for all the shallow machine learning models selected: (a) accuracy, (b) precision, (c) recall, and (d) F1. The definitions of the metrics are thoroughly explained in [47]. Furthermore, Cohen's kappa statistic is evaluated. In order to achieve the best model, a comparison between the performance of the algorithms is made.

The evaluation of the models uses the remaining 20% of the dataset. The hyperparameters and configurations used in the test stage are the same used in the training split. Figure 8 shows the results of the average values in inner iterations in each of the outer iterations.

Figure 8a presents the mean accuracy values for each model. Even though the naive Bayes classifier exhibits the lowest performance for this metric, it is still an excellent result, with a detection rate above 90%. In other words, all the classifiers for the selected configurations perform a correct prediction on the total number of evaluated observations. The best results are achieved by the models based on decision trees, obtaining a perfect result in some iterations.

Precision refers to the ratio of positive instances that are correctly predicted to the total observations in the positive class. Figure 8b illustrates the results obtained for this metric. Decision-tree-based classifiers achieved values of 1 in most iterations. On the contrary, the rest of the models obtained very low values of below 50%.

Since we are interested in the detection of network traffic anomalies, the classification of abnormal packets as normal can imply a risk for the system, and therefore it is important to minimise false negatives. The recall metric, illustrated in Figure 8c, provides an indication of how well the model correctly identifies positives out of the total number of true positives. For this metric, NB obtains deficient values, contrary to SVM, LR, AdaBoost, and RF, which also obtain very satisfactory results.

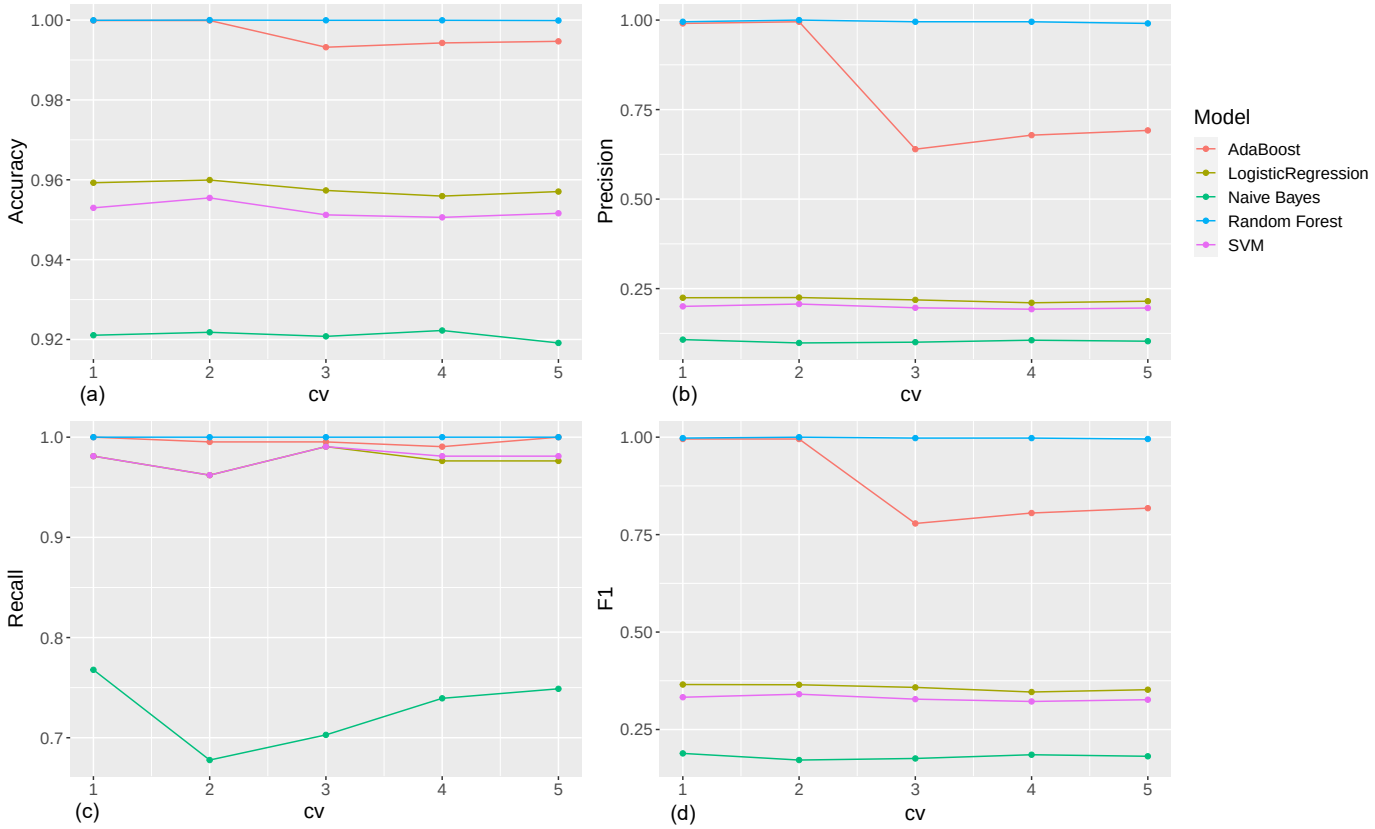

Figure 8. Metrics by iteration: (**a**) accuracy, (**b**) precision, (**c**) recall, and (**d**) F1.

The F1 metric defines a harmonic mean between precision and recall; thus, low values in any of these metrics will directly affect the values of F1, as shown in Figure 8.

The global results obtained for each model and their standard deviation are presented in Table 10. The naive Bayes classifier presents the lowest performance in all metrics. Despite having excellent results in accuracy, NB cannot be declared as a reliable classifier. The linear family models, SVM and LR, give similar results. Likewise, decision-tree-based classifiers obtain similar results and are found to be the best models, with RF being an optimal classifier for this dataset.

Table 10. Best scoring model by iteration.

	NB	AdaBoost	LR	RF	SVM
Mean accuracy	0.9210	0.9964	0.9579	0.9999	0.9524
Mean precision	0.1030	0.7991	0.2187	0.9953	0.1984
Mean recall	0.7272	0.9962	0.977	1	0.9791
Mean F1	0.1805	0.8785	0.3573	0.9976	0.3299
Std. accuracy	1.1980×10^{-3}	3.2419×10^{-3}	1.6584×10^{-3}	4.0069×10^{-3}	1.9381×10^{-3}
Std. precision	3.8186×10^{-3}	0.1779	6.2834×10^{-3}	3.3197×10^{-3}	5.5056×10^{-3}
Std. recall	3.6408×10^{-2}	3.9638×10^{-3}	1.0289×10^{-3}	0	1.0395×10^{-2}
Std. F1	6.9215×10^{-3}	0.10756	8.2403×10^{-3}	1.6677×10^{-3}	7.1099×10^{-3}

Cohen's kappa statistic is a very useful evaluation metric when dealing with imbalanced data. It is a metric often used to evaluate the agreement between two qualifiers and can also be used to assess the performance of a classification model. In the experiments, the NB classifier obtained a kappa score of 0.11, meaning that the classification could have been obtained by a random guess. The LR and SVM classifiers have a poor score of around

0.34. On the other hand, RF and AdaBoost presented kappa statistics of 0.86 and 0.99, respectively, guaranteeing an excellent classification performance.

10. Discussion

After the recent advances in IoT networks present in daily life and the difficult task of incorporating secure systems, intrusion detection systems have been a versatile tool to certify their integrity and confidentiality.

Preventing or avoiding network failures using anomaly-based IDSs requires accessible datasets to analyse and detect anomalous behaviour in network traffic. The CoAP protocol is one of the most widespread protocols in the IoT environment. However, after reviewing the literature, it was not possible to find datasets fulfilling our requirements that were developed for this protocol to allow determining its scope and limitations.

This article presents a versatile, realistic, easy to handle, and fully labelled dataset designed for use in intrusion detection systems in CoAP-IoT environments. In addition, a specific analysis of the most relevant characteristics of the dataset is undertaken. It contains a total of 88,238 packets, among which we find the use of ARP, IP, and IPv6 protocols. A total of 64,459 packets correspond to UDP, and all the anomalies are performed over the CoAP data. This means all anomaly packets belong to UDP/CoAP and represent 1.64% of the UDP packets and 1.2% of the total packets presented in the dataset.

To validate the dataset, we selected five widely used shallow machine learning algorithms, such as logistic regression, naive Bayes, random forest, and linear support vector machine. Three of them base their behaviour on statistical linear functions (logistic regression, naive Bayes, and SVM) and the other two on are based on decision trees.

The data entered into the classifier had to be appropriate so that it was perfectly understood by the classifier. Initially, data pre-processing using discretisation techniques was performed. Then, RFE for feature eliminations was carried out. The SMOTE data balancing technique, along with stratified k-fold cross-validation, assured the presence of the minority class, while the hyper parameterisation of the model looked for the configuration that presented the best results for the classifier.

To evaluate machine learning algorithms for classification, the cv train–test split procedure was used. In the training stage, the AUC ROC score was considered. The best values achieved were 0.97 for LR and SVM, 0.96 for NB, and 1 for RF and Adaboost, in different configurations of parameter combinations.

For the test stage, as traditional for ML models, four measures were analysed to determine the quality of the classifier: accuracy, precision, recall, and F1. However, due to the type of scenario in which the study was developed, i.e., detection of traffic anomalies, the most relevant aspect is the adequate detection of positives. This is the reason why recall is so important in this context, since it measures the ability of an algorithm to predict a positive outcome when the actual outcome is positive.

The obtained results confirm the tree-based classifiers as the best for this dataset, reaching almost perfect values (close to 1) and also presenting the lowest standard deviations, indicating the stability of the classifier. As expected, models of the same nature obtained very similar results. Cohen's kappa supplied a more objective description of the model performance, confirming RF and AdaBoost as the best classifiers. These results demonstrate that this dataset can be used for anomaly traffic detection in IoT-CoAP environments.

In conclusion, we present the first application of ML algorithms on the proposed dataset for anomaly detection in IoT networks. In future work, we will consider other datasets in order to verify the obtained results. On the other hand, this research focuses on the application of shallow machine learning models. However, in the future, the research can be expanded to the application of deep learning algorithms on the proposed dataset.

Author Contributions: Conceptualisation, L.V. and V.C.; methodology, D.F. and V.C.; software, L.V., A.P., V.C. and D.F.; validation, V.C. and D.F.; formal analysis, L.V., A.P., V.C. and D.F.; investigation, L.V., A.P., V.C. and D.F.; resources, L.V., V.C., D.F. and A.P.; writing—original draft preparation, L.V., V.C. and D.F.; writing—review and editing, L.V., V.C. and D.F.; supervision, V.C.; project administration, L.V. All authors have read and agreed to the published version of the manuscript.

Funding: This project was funded by the Accreditation, Structuring, and Improvement of Consolidated Research Units and Singular Centers (ED431G/01), funded by Vocational Training of the Xunta de Galicia endowed with EU FEDER funds and Spanish Ministry of Science and Innovation, via the project PID2019-111388GB-I00.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All datasets are publicly available.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Donta, P.K.; Srirama, S.N.; Amgoth, T.; Annavarapu, C.S.R. Survey on recent advances in IoT application layer protocols and machine learning scope for research directions. *Digit. Commun. Netw.* **2021**, *8*, 727–744. [CrossRef]
2. RFC 7252 Constrained Application Protocol. Available online: https://coap.technology/ (accessed on 17 October 2022).
3. Rahman, R.A.; Shah, B. Security analysis of IoT protocols: A focus in CoAP. In Proceedings of the 2016 3rd MEC International Conference on Big Data and Smart City (ICBDSC), Muscat, Oman, 15–16 March 2016; pp. 1–7. [CrossRef]
4. Fahim, M.; Sillitti, A. Anomaly Detection, Analysis and Prediction Techniques in IoT Environment: A Systematic Literature Review. *IEEE Access* **2019**, *7*, 81664–81681. [CrossRef]
5. Shafiq, M.; Thakre, K.; Krishna, K.R.; Robert, N.J.; Kuruppath, A.; Kumar, D. Continuous quality control evaluation during manufacturing using supervised learning algorithm for Industry 4.0. *Int. J. Adv. Manuf. Technol.* **2023**, 1–10. [CrossRef]
6. Bhuyan, M.H.; Bhattacharyya, D.K.; Kalita, J.K. Towards Generating Real-life Datasets for Network Intrusion Detection. *Int. J. Netw. Secur.* **2015**, *17*, 683–701.
7. Chen, H.; Xiong, Y.; Li, S.; Song, Z.; Hu, Z.; Liu, F. Multi-Sensor Data Driven with PARAFAC-IPSO-PNN for Identification of Mechanical Nonstationary Multi-Fault Mode. *Machines* **2022**, *10*, 155. [CrossRef]
8. Centro de Investigación en Tecnoloxías da Información e as Comunicacións de Galicia. Available online: https://www.citic-research.org/ (accessed on 30 October 2022).
9. Meidan, Y.; Bohadana, M.; Mathov, Y.; Mirsky, Y.; Shabtai, A.; Breitenbacher, D.; Elovici, Y. N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders. *IEEE Pervasive Comput.* **2018**, *17*, 12–22. [CrossRef]
10. Mirsky, Y.; Doitshman, T.; Elovici, Y.; Shabtai, A. Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection. *arXiv* **2018**, arXiv:1802.09089.
11. Abbasi, F.; Naderan, M.; Alavi, S.E. Anomaly detection in Internet of Things using feature selection and classification based on Logistic Regression and Artificial Neural Network on N-BaIoT dataset. In Proceedings of the 2021 5th International Conference on Internet of Things and Applications (IoT), Isfahan, Iran, 19–20 May 2021; pp. 1–7. [CrossRef]
12. Liu, Z.; Thapa, N.; Shaver, A.; Roy, K.; Siddula, M.; Yuan, X.; Yu, A. Using Embedded Feature Selection and CNN for Classification on CCD-INID-V1—A New IoT Dataset. *Sensors* **2021**, *21*, 4834. [CrossRef]
13. MontazeriShatoori, M.; Davidson, L.; Kaur, G.; Lashkari, A.H. Detection of DoH Tunnels using Time-series Classification of Encrypted Traffic. In Proceedings of the 2020 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), Calgary, AB, Canada, 17–22 August 2020; pp. 63–70.
14. Doshi, R.; Apthorpe, N.; Feamster, N. Machine Learning DDoS Detection for Consumer Internet of Things Devices. In Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW), San Francisco, CA, USA, 24 May 2018; pp. 29–35.
15. Pahl, M.; Aubet, F. All Eyes on You: Distributed Multi-Dimensional IoT Microservice Anomaly Detection. In Proceedings of the 2018 14th International Conference on Network and Service Management (CNSM), Rome, Italy, 5–9 November 2018; pp. 72–80.
16. Hasan, M.; Islam, M.M.; Zarif, M.I.I.; Hashem, M. Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. *Internet Things* **2019**, *7*, 100059. [CrossRef]
17. Anthi, E.; Williams, L.; Słowińska, M.; Theodorakopoulos, G.; Burnap, P. A Supervised Intrusion Detection System for Smart Home IoT Devices. *IEEE Internet Things J.* **2019**, *6*, 9042–9053. [CrossRef]
18. Lemay, A.; Fernandez, J.M. Providing SCADA Network Data Sets for Intrusion Detection Research. In Proceedings of the 9th Workshop on Cyber Security Experimentation and Test (CSET 16), Austin, TX, USA, 8 August 2016; USENIX Association: Austin, TX, USA, 2016.

19. Duque, S.; Kanoor, S.; Fraunholz, D.; Schotten, H.D. Evaluation of Machine Learning-based Anomaly Detection Algorithms on an Industrial Modbus/TCP Data Set. In Proceedings of the 13th International Conference on Availability, Reliability and Security, Hamburg, Germany, 27–30 August 2018.
20. Frazao, I.; Abreu, P.H.; Cruz, T.; Araújo, H.; Simoes, P. Denial of Service Attacks: Detecting the Frailties of Machine Learning Algorithms in the Classification Process. In Proceedings of the Critical Information Infrastructures Security, CRITIS 2018, Kaunas, Lithuania, 24–26 September 2018; Luiijf, E., Žutautaitė, I., Hämmerli, B.M., Eds.; Springer International Publishing: Cham, Switzerland, 2019; pp. 230–235.
21. Hamza, A.; Gharakheili, H.H.; Benson, T.A.; Sivaraman, V. Detecting Volumetric Attacks on LoT Devices via SDN-Based Monitoring of MUD Activity. In Proceedings of the 2019 ACM Symposium on SDN Research, SOSR '19, San Jose, CA, USA, 3–4 April 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 36–48. [CrossRef]
22. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [CrossRef]
23. Susilo, B.; Sari, R.F. Intrusion Detection in IoT Networks Using Deep Learning Algorithm. *Information* **2020**, *11*, 279. [CrossRef]
24. Alsamiri, J.; Alsubhi, K. Internet of Things Cyber Attacks Detection using Machine Learning. *Int. J. Adv. Comput. Sci. Appl.* **2019**, *10*. [CrossRef]
25. Kang, H.; Ahn, D.H.; Lee, G.M.; Yoo, J.D.; Park, K.H.; Kim, H.K. *IoT Network Intrusion Dataset*; IEEE: Piscataway, NJ, USA, 2019. [CrossRef]
26. Parmisano, A.; Garcia, S.; Erquiaga, M.J. Stratosphere Laboratory. A Labeled Dataset with Malicious and Benign IoT Network Traffic. Available online: https://https://www.stratosphereips.org/datasets-iot23 (accessed on 17 October 2022).
27. Thamaraiselvi, D.; Mary, S. Attack and Anomaly Detection in IoT Networks using Machine Learning. *Int. J. Comput. Sci. Mob. Comput.* **2020**, *9*, 95–103. [CrossRef]
28. Aversano, L.; Bernardi, M.; Cimitile, M.; Pecori, R.; Veltri, L. Effective Anomaly Detection Using Deep Learning in IoT Systems. *Wirel. Commun. Mob. Comput.* **2021**, *2021*, 9054336. [CrossRef]
29. Sivanathan, A.; Gharakheili, H.H.; Loi, F.; Radford, A.; Wijenayake, C.; Vishwanath, A.; Sivaraman, V. Classifying IoT Devices in Smart Environments Using Network Traffic Characteristics. *IEEE Trans. Mob. Comput.* **2019**, *18*, 1745–1759. [CrossRef]
30. Moustafa, N. New Generations of Internet of Things Datasets for Cybersecurity Applications based Machine Learning: TON_IoT Datasets. 2019. Available online: http://handle.unsw.edu.au/1959.4/resource/collection/resdatac_921/1 (accessed on 30 October 2022).
31. Sarhan, M.; Layeghy, S.; Moustafa, N.; Gallagher, M.; Portmann, M. Feature Extraction for Machine Learning-based Intrusion Detection in IoT Networks. *Digit. Commun. Netw.* **2022**. [CrossRef]
32. Guerra-Manzanares, A.; Medina-Galindo, J.; Bahsi, H.; Nõmm, S. MedBIoT: Generation of an IoT Botnet Dataset in a Medium-sized IoT Network. In Proceedings of the 6th International Conference on Information Systems Security and Privacy—ICISSP, Valletta, Malta, 25–27 February 2020; SciTePress: Setúbal, Portugal, 2020; pp. 207–218. [CrossRef]
33. Vigoya, L.; Fernandez, D.; Carneiro, V.; Cacheda, F. Annotated Dataset for Anomaly Detection in a Data Center with IoT Sensors. *Sensors* **2020**, *20*, 3745. [CrossRef]
34. Vigoya, L.; Fernandez, D.; Carneiro, V.; Nóvoa, F.J. IoT Dataset Validation Using Machine Learning Techniques for Traffic Anomaly Detection. *Electronics* **2021**, *10*, 2857. [CrossRef]
35. Vaccari, I.; Chiola, G.; Aiello, M.; Mongelli, M.; Cambiaso, E. MQTTset, a New Dataset for Machine Learning Techniques on MQTT. *Sensors* **2020**, *20*, 6578. [CrossRef]
36. Dhifallah, W.; Tarhouni, M.; Moulahi, T.; Zidi, S. A Novel Realistic Dataset for Intrusion Detection in IoT based on Machine Learning. In Proceedings of the 2021 International Symposium on Networks, Computers and Communications (ISNCC), Dubai, United Arab Emirates, 31 October–2 November 2021; pp. 1–6. [CrossRef]
37. Ghazanfar, S.; Hussain, F.; Rehman, A.U.; Fayyaz, U.U.; Shahzad, F.; Shah, G.A. IoT-Flock: An Open-source Framework for IoT Traffic Generation. In Proceedings of the 2020 International Conference on Emerging Trends in Smart Technologies (ICETST), Karachi, Pakistan, 26–27 March 2020; pp. 1–6. [CrossRef]
38. Bhayo, J.; Jafaq, R.; Ahmed, A.; Hameed, S.; Shah, S.A. A Time-Efficient Approach Toward DDoS Attack Detection in IoT Network Using SDN. *IEEE Internet Things J.* **2022**, *9*, 3612–3630. [CrossRef]
39. Ullah, I.; Mahmoud, Q.H. A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks. In Proceedings of the Advances in Artificial Intelligence, Canadian AI 2020, Ottawa, ON, Canada, 13–15 May 2020; Goutte, C., Zhu, X., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 508–520.
40. Al-Hawawreh, M.; Sitnikova, E.; Aboutorab, N. X-IIoTID: A Connectivity-Agnostic and Device-Agnostic Intrusion Data Set for Industrial Internet of Things. *IEEE Internet Things J.* **2022**, *9*, 3962–3977. [CrossRef]
41. Al-Hawawreh, M.; Sitnikova, E. Developing a Security Testbed for Industrial Internet of Things. *IEEE Internet Things J.* **2021**, *8*, 5558–5573. [CrossRef]
42. Dadkhah, S.; Mahdikhani, H.; Danso, P.K.; Zohourian, A.; Truong, K.A.; Ghorbani, A.A. Towards the Development of a Realistic Multidimensional IoT Profiling Dataset. In Proceedings of the 2022 19th Annual International Conference on Privacy, Security & Trust (PST), Fredericton, NB, Canada, 22–24 August 2022; pp. 1–11. [CrossRef]
43. Scapy. Available online: https://scapy.net/ (accessed on 30 October 2022).
44. Irizarry, R.A. *Introduction to Data Science*; Chapman and Hall/CRC: Boca Raton, FL, USA,2019. [CrossRef]

45. Scikit-Learn Machine Learning in Python. Available online: https://scikit-learn.org/stable/ (accessed on 12 March 2023).
46. Wu, J.; Chen, X.Y.; Zhang, H.; Xiong, L.D.; Lei, H.; Deng, S.H. Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimizationb. *J. Electron. Sci. Technol.* **2019**, *17*, 26–40. [CrossRef]
47. Hossin, M.; Sulaiman, M.N. A review on evaluation metrics for data classification evaluations. *Int. J. Data Min. Knowl. Manag. Process* **2015**, *5*, 1.
48. Chapman, P.; Clinton, J.; Kerber, R.; Khabaza, T.; Reinartz, T.; Shearer, C.R.; Wirth, R. *CRISP-DM 1.0: Step-by-Step Data Mining Guide*; SPSS Inc.: Chicago, IL, USA, 2000.
49. Ustebay, S.; Turgut, Z.; Aydin, M.A. Intrusion Detection System with Recursive Feature Elimination by Using Random Forest and Deep Learning Classifier. In Proceedings of the 2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT), Ankara, Turkey, 3–4 December 2018; pp. 71–76. [CrossRef]
50. Chen, C.; Breiman, L. *Using Random Forest to Learn Imbalanced Data*; University of California: Berkeley, CA, USA, 2004.
51. Chawla, N.V.; Bowyer, K.W.; Hall, L.O.; Kegelmeyer, W.P. SMOTE: Synthetic Minority over-Sampling Technique. *J. Artif. Int. Res.* **2002**, *16*, 321–357. [CrossRef]