A Data-Centric Anomaly-Based Detection System for Interactive Machine Learning Setups

Joseph Bugeja[a] and Jan A. Persson[b]
Internet of Things and People Research Center, Department of Computer Science and Media Technology, Malmö University, Malmö, Sweden

Keywords: Anomaly Detection, Interactive Machine Learning, Internet of Things, Virtual Sensors, Intrusion Detection, Poisoning Attack, IoT Security.

Abstract: A major concern in the use of Internet of Things (IoT) technologies in general is their reliability in the presence of security threats and cyberattacks. Particularly, there is a growing recognition that IoT environments featuring virtual sensing and interactive machine learning may be subject to additional vulnerabilities when compared to traditional networks and classical batch learning settings. Partly, this is as adversaries could more easily manipulate the user feedback channel with malicious content. To this end, we propose a data-centric anomaly-based detection system, based on machine learning, that facilitates the process of identifying anomalies, particularly those related to poisoning integrity attacks targeting the user feedback channel of interactive machine learning setups. We demonstrate the capabilities of the proposed system in a case study involving a smart campus setup consisting of different smart devices, namely, a smart camera, a climate sensmitter, smart lighting, a smart phone, and a user feedback channel over which users could furnish labels to improve detection of correct system states, namely, activity types happening inside a room. Our results indicate that anomalies targeting the user feedback channel can be accurately detected at 98% using the Random Forest classifier.

1 INTRODUCTION

Over the past few years, the Internet of Things (IoT) has transformed environments in homes, buildings, cities, and more, connecting them to the Internet. With a forecast of about 29.4 billion connected devices in 2030[1] and the global IoT market revenue estimated to cross USD 1 trillion landmark by 2024[2], the IoT is one of the fastest-growing fields in computing (Al-Garadi et al., 2020). Concurrently, increased adoption of IoT technology has introduced new security challenges.

IoT devices are often left unattended and are mostly connected via wireless networks. The lack of a human oversight process over IoT devices, and hence over data collection processes exposes organizations to new security threats and a broad attack surface that adversaries can exploit (Goldblum et al., 2022). Machine learning (ML) is being utilized and will most likely continue to be employed in IoT systems. This adds to security challenges, especially if ML models need to be updated in an online learning setting. It is even more challenging if, like in interactive ML, this online learning is done by regular IoT system users rather than IoT device suppliers.

Interactive ML setups tend to be prone to integrity attacks. This is as outsiders can manipulate training datasets by injecting invalid or malicious data (Kebande et al., 2020). A poisoning attack, or more specifically, a poisoning integrity attack (Jagielski et al., 2018), is a threat to the integrity of a system designed to adversely affect the operation of a system. According to a poll of industry practitioners done in 2020 (Siva Kumar et al., 2020), organizations reported that they are far more concerned about poisoning threats than other adversarial ML threats. These attacks have been exploited in the real-world. For example, the manipulation of Microsoft's Tay chatbot (Wolf et al., 2017) is an example demonstrating the

[a] https://orcid.org/0000-0003-0546-072X
[b] https://orcid.org/0000-0002-9471-8405
[1] https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide [Accessed on 19-September-2022].
[2] https://www.researchandmarkets.com/reports/5464819 [Accessed on 19-September-2022].

Bugeja, J. and Persson, J.
A Data-Centric Anomaly-Based Detection System for Interactive Machine Learning Setups.
DOI: 10.5220/0011560100003318
In *Proceedings of the 18th International Conference on Web Information Systems and Technologies (WEBIST 2022)*, pages 182-189
ISBN: 978-989-758-613-2; ISSN: 2184-3252

exploitability of datasets, particularly, when involving a user input channel for ML training purposes. While reactive security mechanisms, such as intrusion detection systems (IDS), can help detect such attacks, they have received little attention in the context of the IoT. Indeed, the majority of the available IDSs tend to be designed for conventional ICT infrastructure or wireless sensor networks, but not for the IoT (Anthi et al., 2018).

In this paper, we propose a data-centric anomaly-based IDS based on ML to detect anomalies associated with integrity attacks targeting interactive online learning scenarios in the IoT. We focus on training-only attacks that affect the user feedback channel. Specifically, we focus on a representative type of poisoning integrity attack, known as a label-flipping attack (Biggio et al., 2012). A label-flipping attack is a type of adversarial attack that exploits classification algorithms by corrupting their training data with small perturbations. Thus, the main goal of this attack type is to fool target systems into misclassifying benign inputs as malicious ones, or vice versa. Unlike most previous research, which has focused on network traffic, we focus on application layer data. Applying anomaly detection on the application layer can detect intrusions that may be missed if only lower layers of network traffic are analyzed (Meyer-Berg et al., 2020). As an example, a manipulated thermostat may show no irregularities on lower layers, e.g., on the network layer, whereas actual temperature readings, e.g., as captured in an activity log, might indicate an anomaly. An attack on smart thermostats was demonstrated by security researchers at DefCon 24[3], where they uploaded a proof-of-concept ransomware to a smart thermostat, allowing them to manipulate the temperature until the homeowner paid a ransom; potentially evidence of such an attack could have been captured at the application layer in the form of anomalous temperature readings. We interpret an anomaly as an intrusion (Khraisat and Alazab, 2021), which represents any significant deviation between an observed behavior and the learned ML model.

Our proposed data-centric anomaly-based detection system is demonstrated in a case study consisting of a smart campus setup that involves: a smart camera, a climate sensmitter, smart lighting, a smart phone, and a user feedback channel, over which users can provide feedback to the training process. For creating this setup, we leverage a concept known as the Dynamic Intelligent Virtual Sensor (DIVS) (Tegen et al., 2019). The DIVS essentially extends the notion of a virtual sensor, which is typically used in devices with a fixed set of sensors, to a dynamic setting with heterogeneous sensors. Through the application of supervised ML algorithms trained on application layer data, we demonstrate that anomalies targeting the user feedback channel of interactive ML setups can be accurately detected at 98% using the Random Forest classifier.

[3]https://defcon.org/html/defcon-24/dc-24-news.html [Accessed on 19-September-2022].

2 RELATED WORK

Approaches to building anomaly detectors for IDSs can be broadly categorized as design-centric and data-centric (MR et al., 2021).

Design-centric approaches make use of physical relationships, captured as invariants, among a system's components (MR et al., 2021). This means that, if an invariant exists for a system, it can be used as a basis for detecting anomalies in the system's behavior. However, design-centric approaches tend to be based on the assumption that the system itself is a closed environment, such as a private home, in which all components are known. In data-centric approaches, such relationships among system components are learned and modelled through the application of ML and computational intelligence techniques, namely, supervised, unsupervised, and hybrid (semi-supervised) algorithms (Alsoufi et al., 2021)(MR et al., 2021)(Albulayhi et al., 2021). This also means that they can better cater to open or semi-open environments such as a building or campus.

Given their ability to automatically learn the dynamics and strategies deployed in a system and the dynamic and heterogeneous nature of an IoT system, we focus on data-centric approaches for developing our anomaly detector. Another advantage of the data-centric approaches is that they could be used to better detect new attacks, such as zero-day attacks, and also need fewer human interventions. Moreover, we focus on the supervised learning approach to anomaly detection (Lin et al., 2015). Supervised learning involves the collection and analysis of every input variable and an output variable, and an algorithm to learn the normal user behaviour from the input to the output (Khraisat and Alazab, 2021).

There have been several similar works done in IoT domains. The following are some of the most recent notable works on IDSs that have used a data-centric approach to anomaly detection in the IoT context.

Liu et al. (Liu et al., 2018) proposed a light probe routing mechanism for detecting On-Off attacks caused by malicious network nodes in an industrial IoT site. An On-Off attack in this context means a malicious network node could target the IoT network when it is active (on) and perform normally when it is in an inactive (off) state.

Diro and Chilamkurti (Diro and Chilamkurti, 2018) proposed a deep learning model to detect distributed attacks in a social IoT network where they compared the performance of the deep model with a shallow neural network using the NSL-KDD (Diro and Chilamkurti, 2018) dataset. This work's primary focus was to detect four classes (normal, DoS, probe, and R2LU2R) of attacks and anomalies. Their system achieved an accuracy of 98.27% for the deep neural network model and an accuracy of 96.75% for the shallow neural network model.

Kozik et al. (Kozik et al., 2018) introduced an attack detection technique that used the extreme learning machine (ELM) method in the Apache Spark cloud architecture. This work focused on three main cases in IoT systems – scanning, command and control, and infected host – and attained accuracy levels of 99%, 76%, and 95%, respectively.

Pajouh et al. (Pajouh et al., 2019) proposed a two-stage dimension reduction and classification model to detect anomalies, namely U2R and R2L attacks, in IoT backbone networks. They used principal component analysis and linear discriminate analysis feature extraction methods to reduce features of the dataset and then used NB and KNN to identify anomalies at an 84.82% identification rate.

Hasan et al. (Hasan et al., 2019) evaluated the performance of five ML algorithms (Logistic Regression, Support Vector Machine, Decision Tree, Random Forest, and Artificial Neural Network) for detecting attacks, namely, Denial of Service, Data Type Probing, Malicious Control, Malicious Operation, Scan, Spying, and Wrong Setup, in the IoT context. One of their findings is that their system obtained 99.4% test accuracy for Decision Tree, Random Forest, and ANN. However, the Random Forest classifier performed the best when evaluated with other performance metrics.

While all of the above works rely on ML and ensemble techniques for anomaly detection, our proposal is different. First, in our approach, we do not rely on an existing dataset but instead create our own dataset based on actual data as captured from our IoT test lab (i.e., smart campus setup) and a handcrafted data simulator. The simulator allows for introducing additional data, allowing for performing controlled experiments on larger datasets. Second, in comparison to some of the existing work that rely on a single method for detecting anomalies, we apply multiple methods – Logistic Regression, Gaussian Naive Bayes, K-Nearest Neighbors, Decision Tree, and Random Forest – to identify the most accurate detection method. Third, we focus on an interactive ML setup. This setup offers a unique setting for studying some of the state-of-the-art attacks that exploit the user feedback channel. Consequently, we also study an attack, a label-flipping attack, that has been relatively understudied in previous works, especially those concentrating on the topic of anomaly detection suited for IoT-based setups.

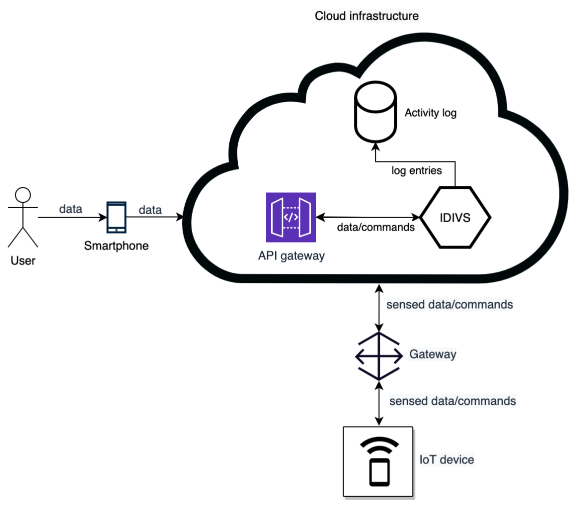

Figure 1: Conceptual representation of the DIVS system structure.

3 SYSTEM MODEL

A system model describes the components and functionality of the system being studied. We assume our system to be one that exhibits the characteristics of an interactive ML setup. For this reason, we assume our system to be an instance of the DIVS. The DIVS is a logical entity (software) that produces a sensor-like output in real time (Tegen et al., 2019). It capitalizes on interactive ML to make use of people's presence in the environment. This is done with the intent of improving the accuracy of the underlying ML models. Figure 1 is a graphical depiction of this model.

At its core, the DIVS reads sensor-like data from the environment and users. The environment, e.g., the smart campus, can feature heterogeneous IoT devices, ranging from constrained devices like single-sensor devices to resourceful devices like smart cameras. Users can be in the form of service users or interactive service users. The main difference between the two user types is that service users rely on the DIVS output to gain insights and perform actions, whereas interactive service users extend the functionality of the service users, allowing them to furnish input (e.g., in the form of labels) to the system to help it learn and adapt.

A core use case of the DIVS is to help detect the type of activity happening in a room. This activity recognition task can be performed by using feedback from interactive users in an interactive ML approach. These users are provided with a smartphone and an accompanying mobile application. Through the mobile application, they could then furnish a label indicating the activity being performed (e.g., "silent", "convo/meeting", and "gathering"); while detecting activities can be done automatically through the use of computer vision techniques, human input can help improve the accuracy of the detection processes.

We assume the activities, i.e., the actions performed by the users and those that are externally generated from the environment (e.g., temperature change), of the DIVS are captured in an activity log. This log identifies, among other things, the date, time, device, and type of messages being exchanged, particularly, between the interactive service users and IoT devices. The message type can be either a data message (e.g., an activity type) or a command message (e.g., an actuation value). Other information, particularly that related to network and system level events, is captured in other log files; these log files are out of scope for this study. Furthermore, we assume that the IoT devices communicate via a central gateway/router device. This communication model, while being the typical for IoT systems, also allows for easier analysis of log files for anomaly detection purposes.

4 THREAT MODEL

A threat model describes the potential ways an adversary can compromise a system. Formally, we assume a sample-label pair of the normal dataset to be: $\alpha=\{(x_1,y_1),(x_2,y_2),(x_3,y_3),\ldots,(x_N,y_N)\}$, where, $x_i, i=1,\ldots,N$ is the sample, and $y_i, i=1,\ldots,N$ is the label corresponding to the sample. The correct labels are assumed to be provided by interactive service users, who tend to have knowledge of the current state of the environment. Labels are provided during the ML training process, which might occur in an online fashion. Based on the formalization of the label-flipping attack proposed by Liu et al. (Liu et al., 2021), we assume that the attack takes α and transforms it into: $\beta=\{(x_1,\overline{y_1}),(x_2,\overline{y_2}),(x_3,\overline{y_3}),\ldots,(x_N,\overline{y_N})\}$, where, $x_i, i=1,\ldots,N$ is the sample, and $\overline{y_i}, i=1,\ldots,N$ is the label corresponding to the sample after label-flipping. We assume that α is transformed to β by the function θ, i.e., $\theta:\alpha\rightarrow\beta$. Moreover, we assume the set of all labels of the normal dataset Y and

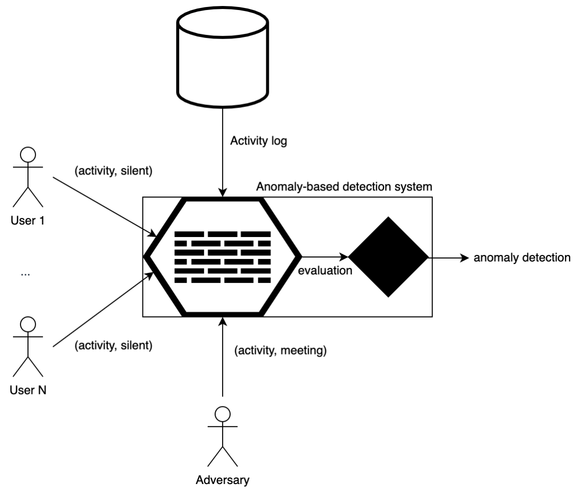

Figure 2: Threat model for the DIVS system enhanced with anomaly detection.

the set of all the labels of the malicious dataset $\overline{Y}$ respectively expressed as: $Y=\{(y_1,y_2,y_3,\ldots,y_N)\}$ and $\overline{Y}=\{(\overline{y_1},\overline{y_2},\overline{y_3},\ldots,\overline{y_N})\}$, where, $\overline{Y}=Y$ but $y_i\neq\overline{y_i}, i=1,\ldots,N$. This implies that malicious labels must be present in the normal dataset.

We assume that the adversary's goal is to corrupt the learning model generated in the training phase, so that predictions on new data will be modified in the test phase. This, in our case, translates to the system not being able to accurately detect an activity type being performed. Moreover, we assume that the attack is a gray-box attack, meaning that the adversary has partial knowledge about the system, i.e., the DIVS setup. In our case, we assume that the adversary has at least some knowledge on the features values of the system, specifically, the labels and the types of activities. Finally, when it comes to the capability of the adversary, we assume that an adversary can inject poison points into the training set. However, we assume that an adversary cannot manipulate the activity log; in practice, this requirement is often implemented through access control mechanisms.

5 EXPERIMENTAL SETUP

To conduct the experiment, we created a smart campus setup. This setup is an instantiation of the system model proposed earlier in Section 3. Specifically, it consisted of the following devices: a smart camera, a climate sensmitter, smart lighting, and a smart phone. These devices are interconnected with the users and services deployed over the cloud using the Message Queue Telemetry Transport (MQTT) protocol; internally, this translates to the exchange of data/commands between the mentioned entities. MQTT is a lightweight IoT-oriented publish-subscribe protocol and has been selected as it is widely used in IoT systems (Roldán-Gómez et al., 2021). Also, the setup featured a user feedback channel, over which users could provide online feedback to the DIVS' learning/training modules. The DIVS, in our case, was configured to help detect activity types and occupancy rate of a room. For the purposes of our experiment, we focus our analysis work on activity types. Nonetheless, both data messages are represented and captured in the activity log.

In the following sections, we describe the different phases of the experiment.

5.1 Data Collection and Preprocessing Phase

To generate the data, we collected activity data (i.e., normal data) from the deployed DIVS setup. Furthermore, based on the specification of the collected data, we generated additional data using a hand-crafted simulator. This simulator was developed using the Python programming language libraries – Panda and NumPy. Panda was used for generating the tabular data, whereas NumPy was used for generating numerical data. The final raw dataset consisted of a total of 600 records (N = 600) representing data generated for the entire month of November 2021. The dataset is composed of data extracted from a network using MQTT. A representation of the collected activities for the month of November is displayed in Figure 3. The final code was deployed on the cloud using Datalore[4].

The collected raw data was transformed as part of the preprocessing phase. Data preprocessing consists of cleaning of data, visualization of data, feature engineering, and vectorization steps. The raw dataset contained both categorical and numerical data. Categorical data, such as activity types, was converted

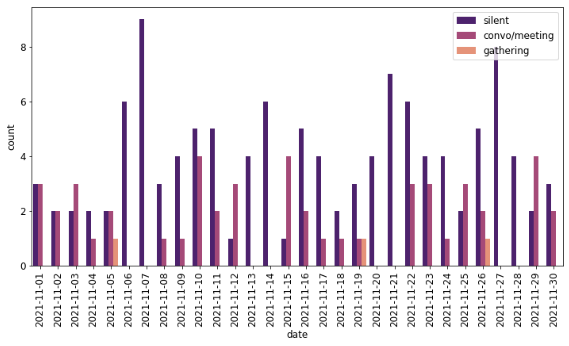

Figure 3: Type of activities happening in November 2021.

into vectors using label encoding. Numerical data, such as sensor and actuator values, was removed and hence not used for model development later. This is because it was not related to the user feedback process, let alone the activity recognition task. Finally, to the dataset, which contains normal patterns, we added the feature anomaly. A summary of all the dataset features is provided in Table 1.

By generalizing on the labeling function proposed by Pathak et al. (Pathak et al., 2021) for labeling anomalies, we assume that the feature anomaly is set by the function $\gamma(l)$ as indicated in Equation 1. The function $\gamma(l)$ is $\{0,1\}$ with 0 representing $l = N$, and N representing application data coming in on a normal day, and 1 the contrary. We assume A to represent abnormal data, i.e., irregular input data coming in as a result of an adversary calling θ.

$$\gamma(l) = \begin{cases} 0, \text{if } l = N \\ 1, \text{if } (l \neq N) \wedge (l = A) \end{cases} \tag{1}$$

The end result of the data collection and preprocessing phase is a dataset consisting of feature vectors.

5.2 Model Development and Validation Phase

A major goal of the ML process is to find an algorithm that most accurately predicts future values based on a set of features. Accordingly, we split our preprocessed data into training and test datasets. Effectively, we split the feature vectors into an 80-20 ratio representing, training and test data, respectively. The data was split using random sampling.

As part of the training set, we selected a random sample of activities from the dataset that correspond to the user feedback process. Then, to simulate a label-flipping attack, we poisoned those records via θ. For simplicity, as part of the interactive ML training process, we set $\gamma(l)$ = 1 when: (a) there was a gathering or a convo/meeting being held before 08:00 and after 19:00; and (b) when the update was done by a principal, i.e., an interactive service user, other than User1 and User2. We flag these as anomalies as the smart campus tends to be vacated during the specified time period and updates tend to occur by those two users. In total, 35% of the training dataset was poisoned[5]. An overview of the anomalies and normal behaviour registered for the month of November

[4]https://datalore.jetbrains.com [Accessed on 19-September-2022].

[5]The processed dataset is available here: https://bit.ly/3Uld9nL

Table 1: Feature description.

Feature	Description	Data type
timestamp	Date and time the activity occurred	discrete
principal	Entity performing the activity	nominal
device	Source IoT device used to perform the activity	nominal
activity	Type of activity performed	nominal
message	Indicates whether the activity is a command or data	nominal
attribute	Physical or virtual feature of the environment or system	nominal
value	Content of the attribute	nominal
anomaly	Indicates whether the activity represents an anomaly	binary

2021, after running the label-flipping attack, is displayed in Figure 4.

Next, we applied five ML algorithms: Logistic Regression, Gaussian Naive Bayes, K-Nearest Neighbors, Decision Tree, and Random Forest; for training the anomaly-based detection system. All the features identified in Table 1 were used as input for training the aforementioned ML algorithms. During the training phase, the time component of the timestamp was dynamically extracted and used, instead of the entire timestamp. The timestamp feature is not considered in its entirety as it has minimal correlation to the dataset's predictor variable normality and also because the target model was not intended to be a time series model but a classification model.

Finally, during the model validation phase, different evaluation metrics were used in the comparison of performance. Namely, the metrics are: AUC (Area Under Curve) of ROC (Receiver Operating Characteristics), accuracy, and F1-score. AUC-ROC is an evaluation metric that calculates the rank correlation between predictions and targets. Accuracy is an evaluation metric that measures how many observations, both positive and negative, were correctly classified. The F1-score is an evaluation metric that combines precision and recall into one metric by calculating the harmonic mean between those two.

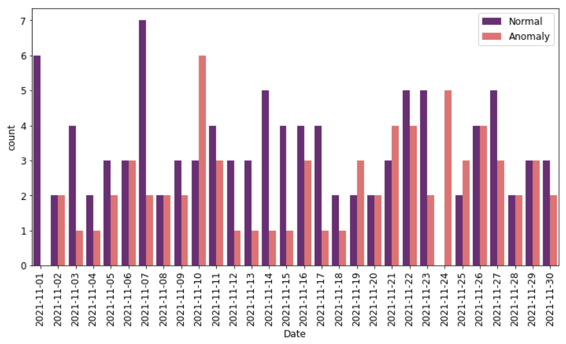

Figure 4: Normal and anomalous activity types being registered during the month of November 2021.

6 RESULTS AND DISCUSSION

The results obtained from our experiment demonstrate that label-flipping attacks were successfully detected, to different extents, by the proposed data-centric anomaly-based detection system. In Table 2, we summarize the evaluation metrics (AUC-ROC score, accuracy, and F1-score) for each of the different ML algorithms. The results indicate that the Random Forest was the algorithm with the highest AUC-ROC score, accuracy, and F1-score.

In Figure 5, we present the ROC curve obtained for the five ML classification models. The ROC curve is a chart showing the performance of a classification model at all classification thresholds. In our case, it indicates that the best predictions are those attained when using the Random Forest algorithm. A similar finding was reported by Husan et al. (Hasan et al., 2019), who in their proposed system for attack and anomaly detection in IoT sites, empirically found that Random Forest also offers the best overall performance in comparison to the other ML classifiers they evaluated for detecting cyberattacks on IoT networks. The corresponding confusion matrix for the Random Forest classifier is presented in Figure 6. This chart illustrates the performance of that classification model on a set of test data for which the true values are known.

The obtained results demonstrate that without the need to have any hard-coded rules (which may also be challenging to write), e.g., as required by IDS solutions such as Snort (Roesch et al., 1999) that tend to rely on pattern matching detection, we were able to detect anomalies pertaining to label-flipping attacks. Even though the specified anomalies could have been detected using simple programming rules, we were able to identify them automatically by harnessing supervised ML. This is beneficial, particularly for future use cases where such rules may be more challenging to develop. Furthermore, even though there are more specific existing proposals for defending against poisoning techniques (e.g., methods from robust statis-

Table 2: Measuring the anomaly detection performance using the metrics: AUC-ROC, accuracy, and F1-Score.

ML classifier	AUC-ROC	Accuracy	F1-Score
Logistic Regression	0.465368	0.488372	0.656250
Gaussian Naive Bayes	0.909091	0.860465	0.863636
K-Nearest Neighbors	0.833333	0.697674	0.628571
Decision Tree	0.976190	0.488372	0.000000
Random Forest	**1.000000**	**0.976744**	**0.976744**

tics such as Huber regression (Huber, 1992)), our aim was to demonstrate how these attacks and potentially other attacks can be detected using our proposal. With the proposed system, it also becomes easier to extend the model to detect other, perhaps more complex, attack types that exploit the user feedback process. In practice, though, there are limitations to our approach.

A first limitation of our approach is that while the accuracy obtained is noteworthy, the current model implementation requires retraining similar to batch learning. This is necessary in the beginning, particularly to avoid the cold start problem. However, this might be challenging to do for some IoT setups, such as certain types of smart buildings, where activities might be happening continuously. While training the model with the different classifiers took only 2.6s, other algorithms may need to be explored to allow for partial retraining of the model. Another limitation is the dataset size. The dataset consisted of only 600 records. This may be significantly smaller than what one would expect in a real-world setting. However, this is still large enough to demonstrate that the proposed approach is capable of detecting anomalies with good accuracy given only a small training set. A final limitation concerns the validation method used. The data was divided into training and testing sets via a train/test split method. Consequently, there might have been some classifier overfitting. While we reduced that risk by repeating the train/test split with various random state values, cross-validation could have been implemented instead. Cross-validation automatically splits the dataset into k folds (subsets) and performs the training phase on k-1 folds and tests on the remaining fold.

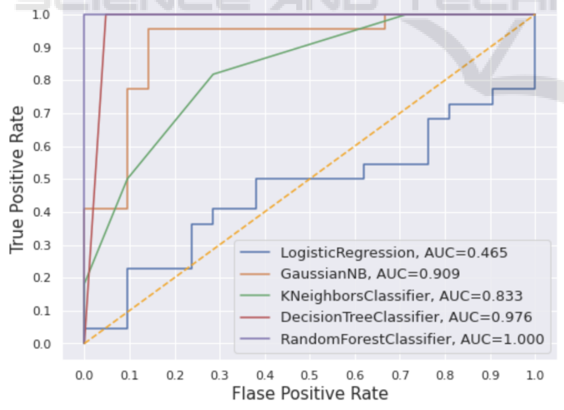

Figure 5: ROC Curve of: Logistic Regression, Gaussian Naive Bayes, K-Nearest Neighbors, Decision Tree, and Random Forest.

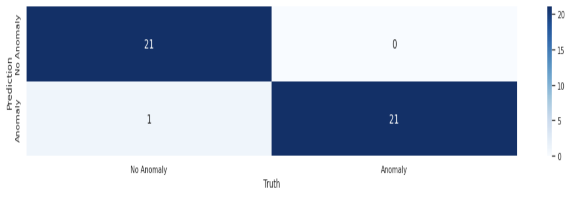

Figure 6: Confusion matrix of Random Forest.

7 CONCLUSION AND FUTURE WORK

A major concern in the use of IoT technologies in general is their reliability in the presence of security threats and cyberattacks. In this paper, we presented a data-centric anomaly-based detection system intended for interactive ML setups. By evaluating the system in a case study consisting of an exemplary smart campus setup that communicates via the MQTT protocol, we observed that of the five evaluated ML classifiers, the Random Forest classifier was the most accurate, at 98%, in detecting anomalies. An advantage of the presented system is that it requires no hard-coded rules, thus making it suitable for detecting new types of cyberattacks that were not initially identified by the domain experts.

Although our results are promising, there are still various elements and parameters that we need to consider to fully develop the system. As part of our future work, we plan to combine the proposed approach with Complex Event Processing to help collect and analyze different IoT data streams in real time for cyberattacks. Here, we also intend to perform more experiments to further validate or improve the proposed approach. Moreover, we plan to extend the system to be able to detect other attack types, for example, low-level attacks, e.g., network protocol Denial-of-Service attacks, as well as high-level attacks, e.g., command injection attacks. Possibly, implementing this requires the analysis of network data in addition to application data. Finally, we plan to develop interfaces that can help notify the building administrators when the system detects anomalies in real time. This process can further help improve the accuracy of the system by having an expert user decide if a flagged anomaly is a false positive or not. Eventually, such a system could be possibly integrated into another mechanism to be able to also react to attacks by blocking them in real time.

ACKNOWLEDGEMENTS

This work has been carried out within the research profile "Internet of Things and People", funded by the Knowledge Foundation and Malmö University.

REFERENCES

Al-Garadi, M. A. et al. (2020). A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. *IEEE Communications Surveys Tutorials*, 22(3):1646–1685.

Albulayhi, K. et al. (2021). IoT Intrusion Detection Taxonomy, Reference Architecture, and Analyses. *Sensors*, 21(19):6432.

Alsoufi, M. A. et al. (2021). Anomaly-based intrusion detection systems in iot using deep learning: A systematic literature review. *Applied Sciences*, 11(18):8383.

Anthi, E., Williams, L., and Burnap, P. (2018). Pulse: an adaptive intrusion detection for the internet of things. *IET Conference Proceedings*.

Biggio, B., Nelson, B., and Laskov, P. (2012). Poisoning attacks against support vector machines. *arXiv preprint arXiv:1206.6389*.

Diro, A. A. and Chilamkurti, N. (2018). Distributed attack detection scheme using deep learning approach for Internet of Things. *Future Generation Computer Systems*, 82:761–768.

Goldblum, M. et al. (2022). Dataset Security for Machine Learning: Data Poisoning, Backdoor Attacks, and Defenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Hasan, M. et al. (2019). Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches. *Internet of Things*, 7:100059.

Huber, P. J. (1992). Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer.

Jagielski, M. et al. (2018). Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 19–35.

Kebande, V. R., Bugeja, J., and Persson, J. A. (2020). Internet of threats introspection in dynamic intelligent virtual sensing. *arXiv preprint arXiv:2006.11801*.

Khraisat, A. and Alazab, A. (2021). A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. *Cybersecurity*, 4(1):18.

Kozik, R. et al. (2018). A scalable distributed machine learning approach for attack detection in edge computing environments. *Journal of Parallel and Distributed Computing*, 119:18–26.

Lin, W.-C., Ke, S.-W., and Tsai, C.-F. (2015). CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. *Knowledge-Based Systems*, 78:13–21.

Liu, W. et al. (2021). D2MIF: A Malicious Model Detection Mechanism for Federated Learning Empowered Artificial Intelligence of Things. *IEEE Internet of Things Journal*, pages 1–1.

Liu, X. et al. (2018). Defending ON–OFF Attacks Using Light Probing Messages in Smart Sensors for Industrial Communication Systems. *IEEE Transactions on Industrial Informatics*, 14(9):3801–3811.

Meyer-Berg, A. et al. (2020). IoT Dataset Generation Framework for Evaluating Anomaly Detection Mechanisms. In *Proceedings of the 15th International Conference on Availability, Reliability and Security*, ARES '20. Association for Computing Machinery.

MR, G. R., Ahmed, C. M., and Mathur, A. (2021). Machine learning for intrusion detection in industrial control systems: challenges and lessons from experimental evaluation. *Cybersecurity*, 4(1):1–12.

Pajouh, H. H. et al. (2019). A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks. *IEEE Transactions on Emerging Topics in Computing*, 7(2):314–323.

Pathak, A. K. et al. (2021). Anomaly Detection using Machine Learning to Discover Sensor Tampering in IoT Systems. In *ICC 2021 - IEEE International Conference on Communications*, pages 1–6.

Roesch, M. et al. (1999). Snort: Lightweight intrusion detection for networks. In *Lisa*, volume 99, pages 229–238.

Roldán-Gómez et al. (2021). Attack Pattern Recognition in the Internet of Things using Complex Event Processing and Machine Learning. In *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pages 1919–1926.

Siva Kumar, R. S. et al. (2020). Adversarial Machine Learning-Industry Perspectives. In *2020 IEEE Security and Privacy Workshops (SPW)*, pages 69–75.

Tegen, A. et al. (2019). Collaborative Sensing with Interactive Learning using Dynamic Intelligent Virtual Sensors. *Sensors*, 19(3).

Wolf, M., Miller, K., and Grodzinsky, F. (2017). Why We Should Have Seen That Coming: Comments on Microsoft's Tay "Experiment," and Wider Implications. *The ORBIT Journal*, 1(2):1–12.