Comparison of Machine Learning Algorithms for Malware Detection Using EDGE-IIoTSET Dataset in IoT

Jawaher Alshehri, Almaha Alhamed, Mounir Frikha, M M Hafizur Rahman
Department of Computer Networks Communications, King Faisal University, CCSIT,
Al Hofuf, Al Hassa 31982, Saudi Arabia

***Abstract*—The growth of IoT devices has presented great vulnerabilities leading to many malware attacks. Existing IoT malware detection methods face many challenges; including: device heterogene-ity, device resource restrictions, and the complexity of encrypted malware payloads, thus leading to less effective conventional cybersecurity techniques. This study's objective is to reduce these gaps by assessing the results obtained from testing five machine learning algorithms that are used to detect IoT malware by applying them on the EDGE-IIoTSET dataset. Key preprocessing steps include: cleaning data, extracting features, and encoding network traffic. Several algorithms used these include: Logistic Regression, Decision Tree, Naïve Bayes, KNN, and Random Forest. The Decision Tree model achieved perfect accuracy at 100%, making it the best-performing model for this analysis. In contrast, Random Forest delivered a strong performance with an accuracy of 99.9%, while Logistic Regression performed at 27%, Naïve Bayes at 57%, and KNN with moderate performance. Hence, the results have shown the effectiveness of machine learning techniques to enhance the security IoT systems regarding real-time malware detection with high accuracy. These findings are useful input for policymakers, cybersecurity practitioners, and IoT developers as they develop better mechanisms for handling dynamic IoT malware attack incidents.**

***Keywords*—IoT malware; machine learning; malware detection; IoT security; EDGE-IIoTSET**

I. Introduction

The Internet of Things (IoT) is comprises a huge variety of devices connected to one another and exchanging information. These devices include, smart home devices, medical equipment, and industrial machinery, sensors, and wearable technologies. The high and rapid growth of IoT has effected transformation in a variety of fields and sectors like healthcare, transportation, commerce, agriculture, and education [1], [2], [3], [4], [5]. With that in view, IoT has become widely embraced as driving economic growth and improving quality of life, resulting in unbridled worldwide creation of new appliances and projects. This growth comes with enormous security challenges and problems in IoT devices. Currently, these are the prime targets for cyber criminals amidst other digital device. Of these, malware attacks are becoming one of the most salient and dangerous threats in IoT. Most IoT devices are not well-protected and have low computation abilities, which makes them high-value targets for malware attacks despites previous studies were mainly aimed at improving malware detection using sophisticated machine learning techniques, there is still a huge gap between the application of such techniques on resource-constrained IoT devices with severely constrained performance and real-time response. Malware analysis remains crucial in IoT systems for understanding, detecting, and mitigating these threats. It forms part of the insight into how malicious actors compromise devices and networks; hence, it is an essential element of IoT security.

The IoT is both a network and a system. The definition of a network is that it can make com- munication possible between connected devices [6]. It is, in the same moment, considered as a system as it combines other elements and technologies to allow communication and data exchange between devices. Despite the fact that IoT technology offers so many benefits, it has created enormous potential weaknesses that impact performance and effectiveness in prescribed, although the various table text styles are provided. The formatter will need to create these components, incorporating the applicable criteria that follow. operation concurrent with its rapid growth. These malware might cause major financial and operational losses [7]. For instance, a well-publicized, major DDoS attacks on IoT devices and systems are typically executed by a botnet like Mirai [8]. Having fallen to the attacker, the device can then be used to execute other dangerous attacks. This malware is still a serious and evolving irritant in the modern digital world [9]. Thus, malware research will become extremely important for the security analysts and researchers as they try to comprehend different varieties of malware and take countermeasures against them.

It divided into two kinds of analysis: dynamic and static. The dynamic considers malware in an active state, whereas the status is considered for malware in an inactive state. Both are great significance in understanding how to protect IoT devices against malicious activities in-depth. The level of analysis and understanding of the capabilities of malware very much depends on how one can keep IoT safe from hacking and breaches of privacy [10]. Because of these, among other constraints, traditional malware detection techniques do not work well in the IoT environment with very restricted processing power and storage. This work therefore goes ahead to issues relating to IoT malware detection, but with more focus on how effective machine learning algorithms prove. For this reason, we compare various models and find the most suitable one for real-world applications in protecting IoT systems.

These methods are promising but require quite a lot of computational resources, which are difficult to handle using

typical IoT environments with device heterogeneity and less complex processing. This mismatch between capability and necessity creates a critical vulnerability in IoT security frameworks [46]. Most recent studies focus on intensive models including CNNs and LSTMs, which are undoubtedly accurate but still are unsuitable on resource-constrained IoT devices for real-time deployment. Additionally, the diverse malware types are often addressed inadequately. As most of the models focus solely on botnets [47]. The continuing evolution of IoT malware requires detection strategies that are not only effective but also adaptable to the constrained environments typical of many IoT devices. Addressing these gaps is essential.

This study addresses these gaps by analysing lightweight machine learning models to detect IoT malware explicitly accounting diverse malware types. Our research systematically benchmark five machine learning models: Decision Tree, Random Forest, Naïve Bayes, KNN, and Logistic Regression by using the EDGE-IIoTSET dataset [48]. This work contributes to identifying the most effective algorithm, and proposes a scalable and suitable approach for real-time application in a heterogeneous IoT ecosystem. The main objective of this paper is to evaluate the performance of lightweight machine learning algorithms. This paper provides actionable insights regarding the selection and optimization of machine learning algorithms in order to enhance IoT security. Among the research questions that need to be addressed in this study are as follows:

RQ1: What extent can machine learning algorithms effectively detect malware in IoT devices, considering the challenges of device heterogeneity, limited resources, and encrypted malware payloads? RQ2: Which machine learning models are most effective for real-time malware detection in resource-constrained IoT environments?

By exploring these questions, our study aims to give actionable insights that guide the development of more robust and scalable malware detection models tailored for the diversity and dynamic nature of IoT systems, making sure they remain reliable and secure against evolving threats.

II. Literature Review

IoT malware detection has acquired significant attention, and machine learning has grown as a prominent technique to address these threats. This section explores recent developments for detecting IoT malware, identifies gaps in the current literature, and compares the effectiveness of various machine-learning models in malware detection.

A. Recent Development in IoT Malware Detection using Machine Learning

A considerable amount of work is currently being performed for the development of machine learning-based models for the detection of IoT malware. Most of them feature network traffic analysis in search of suspicious patterns and label malicious behaviors using IoT-specific data. For example, the work explores deep learning approaches for detecting botnet activity in IoT devices. The researchers' results demonstrated that CNNs can outperform other ML approaches, including Support Vector Machines and Decision Trees, with a classification accuracy greater than 95% [11]. Following this line of investigation, the research presents an LSTM model for malware detection based on IoT device behavior [12]. It was also found that LSTMs identify time-based patterns in traffic data, reporting an accuracy of around 97%.

The authors of [13] performed a performance study on the application of XGBoost and LightGBM to IoT malware detection. They concluded that LightGBM was most relevant in real-time detection since it computes much more quickly and can save memory compared to Random Forest. With these developments come challenges. Most of the related works considered do not address heterogeneity in IoT devices with their limited computational resources or the payloads in encrypted form arising from specific IoT devices, hence hampering the performance of these ML algorithms [14]. It is such a gap that our research sets out to fill, ensuring focus on lightweight models for efficiency in resource usage but high accuracy in malware detection.

B. Current Gaps in the Literature

Even though the area of IoT malware detection has developed, some gaps still exist in the literature. Most related studies tend to ignore the restricted computational resources provided by IoT devices. For example, although certain studies reported high accuracy using CNNs and LSTMs, they are computationally expensive and hence cannot be realistically deployed on resource-constrained IoT devices [15]. Also, most of these research works pertain to malware types like botnets alone and not all variants of IoT malware, such as ransomware, spyware, and worms [16]. The other limitation is that the literature does not focus on encrypted traffic, which originates from IoT devices. In many cases, the IoT devices encrypt data due to privacy issues; hence, malicious activity detection solely based on encrypted network traffic is limited. Most of the research has focused on plaintext traffic; hence, encrypted network traffic is highly neglected and further limits applicability of in real-time scenarios [17]. Our research covers these gaps by assessing machine-learning models applicable to resource-constrained devices and encrypted traffic analysis.

C. Comparison Study of Various ML Models for IoT Malware Detection

A number of machine learning algorithms have been tried and tested for IoT malware detection; each of them has pros and cons. Recently, XGBoost and LightGBM have gained major attention because of their speedy and bulky handling of data [18]. XGBoost prevents problems of overfitting by using regularization techniques; hence, it is a robust choice in malware detection for IoT environments, which are dynamic. At the same time, it requires very heavy computation, which makes it computationally prohibitively expensive for resource-constrained IoT devices [19]. On the other hand, LightGBM is more resource- efficient and has faster training times than XGBoost, making it more suitable for real-time malware detection in IoT systems [20]. Researchers in [21] found that LightGBM achieved comparable accuracy to XGBoost but used less memory and CPU, making it ideal for low-powered IoT devices. Nevertheless, DNNs have proven quite promising in developing complex patterns from network traffic data, be it CNNs or LSTMs [22]. However, DNNs still suffer from serious computations, which are particularly not suitable for real-time IoT applications [23].

Thus, the lightweight models like Random Forest or Decision Tree are practical models for real-world IoT malware detection [24]. Therefore, our work demonstrates that, in terms of the trade-off between accuracy and computational efficiency, among the techniques under study, Random Forest achieves the best performance and is deployable on resource- constrained IoT devices. Although machine learning has made major advancements in the detection of malware on IoTs, there is still a gap in the literature regarding model suitability for resource-constrained IoT devices and how it handles encrypted traffic. This paper largely extends works that have been previously performed to assess lightweight models and address challenges we have pinpointed in order to create more pragmatic IoT malware detection.

In previous studies, several machine learning methods have been proven to be effective in detecting IoT malware. For instance, Sliwa, Piatkowski, and Wietfeld (2020) demonstrated that Random Forest algorithms can offer reliable malware detection in IoT devices; however, they also identified some disadvantages when dealing with encrypted traffic data. Additionally, Zhang and Zhou (2021) revealed that SVMs excelled in very high-dimensional data scenarios, thereby further indicating how the performance of the algorithm would vary with regard to data characteristics (Sliwa et al., 2020; Zhang & Zhou, 2021).

III. Challenges in IoT Malware Analysis

The analysis of IoT malware shows different challenges due to the IoT ecosystem's nature. This section offers insight into the heterogeneity and diversity of IoT devices, the resource limitations of these constrained environments, encrypted and obfuscated malware payloads, and the privacy concerns and regulatory challenges associated with managing IoT data.

A. Heterogeneity and Diversity of IoT Devices

Heterogeneity in IoT indicates the different array of elements, in terms of various proto- cols, devices, services, and networks within an ecosystem, highlighting the complexity and variability of interconnected elements [25], [26]. Interoperability indicates the key challenge in heterogeneous IoT platforms due to diverse methods for recognizing and identifying devices within different platforms, as well as resource requests. These differences are huge hurdles and create hindrances in data exchange and smooth communication. It is crucial to bridge these interoperability gaps, to ensure seamless operation and secure data exchange within the IoT environment [26]. In addition, integrating diverse IoT technologies and devices from different vendors into a cohesive and unified system can present complex and significant challenges. Each device may have its own application programming interfaces, which complicates data-sharing and integration efforts. Diverse communication protocols also complicate the establishment of connections and data exchange. Metrological characteristics also have also proved to be a significant concern in the integration process due to inconsistencies in measurement units, range, accuracy, and scale among different devices, since every device has unique features. Ensuring temporal consistency and synchronization across many IoT devices, especially in real-time applications, further increases complexity [27].

B. Resource Limitations and Constrained Environments

Few resources and a constrained environment mean IoT devices have limited energy, memory, and processing-power resources. The result of these constraints is the limited ability to implement trade resource-intensive security measures for devices. Additionally, IoT devices are often connected over a lossy link. During the transmission of data, lossy links may have a significant chance of packet loss. Environmental factors, signal attenuation, and wireless interference may cause problems that compromise the security and dependability of IoT networks. As a result, packet loss, delay, and erratic communication between devices may occur. So, mitigating the effects of lossy connectivity is necessary for the security of IoT devices with limited resources [28].

C. Encrypted and Obfuscated Malware Payloads

The identification and analysis of IoT malware could be difficult because malicious programs use various methods to encrypt and hide their payloads. Due to encryption, the payload's exact purpose remains hidden, making it more challenging for traditional antivirus programs to identify and address malicious code. Obfuscation techniques are also used to make it more difficult to study the malicious code, employing strategies like code obfuscation to purposefully make the code more complex and difficult to interpret. This hinders the analysts' ability to comprehend its behavior [29].

D. Privacy Concerns and Regulatory Challenges

IoT presents a multitude of privacy and regulatory concerns around the gathering, storing, and use of data produced by linked devices. Data security is one of the main concerns. As IoT devices expand, they produce vast amounts of data, including sensitive and personal information. Protecting the privacy of this data is mandatory, especially when it comes to personal information about an individual's actions and behaviors. Cyber risks include security breaches and illegal access that can compromise the confidentiality of the data gathered and sent by IoT devices. Addressing security concerns to protect IoT-generated data privacy is crucial [29]. These vulnerabilities, combined with the above challenges of IoT systems and networks, make comprehensive security management a challenging task. As this paper progresses, we will explore advanced detection techniques that aim to overcome these challenges and fortify the protection of IoT systems against evolving threats.

IV. Methodology

The materials and methods section outlines the overview of the used dataset, followed by the rationale for algorithm selection, machine learning in malware detection for IoT systems, and some details about the dataset preprocessing techniques used to ensure accurate results.

A. Machine Learning in Malware Detection for IoT Systems

In the complex ecosystems of Internet of Things systems, machine learning has emerged as a critical technique for enhancing virus detection. We used network traffic simulations to create our dataset in order to ensure that it appropriately reflects common IoT interactions and possible security breaches,

given the diversity and unpredictability present in IoT contexts. This method was chosen because it enhances the validity and application of our study by allowing the dataset to cover a wide range of real-world situations are quite helpful in spotting novel patterns that haven't been assigned a label yet.

Table I details the different types of machine learning approaches we employed and their specific application and benefits toward IoT security. A multi-pronged approach would not only guarantee complete coverage but also make the detection systems robust and reliable for the unexpected nature of IoT malware.

TABLE I. MACHINE LEARNING ALGORITHMS

Type	Description
Supervised learning	Methods include Support Vector Machines (SVMs) and Random Forests, which excel at classifying malware based on predefined labels. These algorithms are trained on datasets that humans provide to models.
Unsupervised learning	Includes Principal Component Analysis (PCA) and segregating data in the form of clusters, offering a complementary strategy by anomaly detection within IoT data. These techniques can detect and identify previously unseen malware variants without relying on pre-labeled data [41].
Deep learning models	Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), in particular, have remarkable capabilities for handling complex and large-scale IoT datasets. These models can effectively extract features from data, which can lead to better detection rates for sophisticated malware [42].
Hybrid models	To enhance detection capabilities, hybrid models combine strengths of both types of analysis-static and the dynamic analyses. While static analysis examines the code structure, dynamic analysis observes code behavior during execution. By integrating these perspectives, hybrid models are advanced models that can enhance the detection of both known and unknown malware threats.

This study investigates the deployment of machine learning to detect malware in internet-enabled gadgets. The research leverages the EDGE-IIoTSET dataset, and we preprocessed network traffic data to extract relevant features. We compared SVM, Random Forest, and CNN models to identify optimal algorithms for classifying malware. Standard metrics were used to evaluate model performance and execution, considering computational efficiency for practical IoT deployment.

Each machine learning model was carefully optimized to balance predictive accuracy with a computationally fair load. The Random Forest, for example, was set at a specific number of trees such that it would neither overfit, nor be impractical to use on limited resource devices typical for IoT setups, and SVMs were optimized about kernel type and the regularization parameters about the best discrimination between malicious traffic and benign one without requiring exhaustive computational resources.

Our approaches ensured that optimizations reduced computational overhead; otherwise, IoT resources are characterized by limited implementations, which might prove challenging. For instance, in the Random Forest approach, it was setup to have just a few trees so as to decrease model complexity and runtime. Such made it feasible for real-time virus detection of devices in an Internet of Things device with processing capabilities.

B. Rationale for Algorithm Selection

The major concerns necessary for the applied machine learning algorithms to discover IoT malware are scalability and efficiency. Python and Scikit-learn were chosen with great care due to their excellent documentation, strong community support, and large selection of pre-built functions for data processing and machine learning. These characteristics make it easy for other researchers to replicate our methods. Python provides scalability and efficient calculation when working with massive datasets, therefore the choice also fits with the requirement for real-time processing capabilities in IoT systems. IoT networks may generate a huge volume of data. Therefore, the chosen algorithms must be able to scale up while handling large-sized datasets efficiently, without being computation-heavy for resource-constrained IoT devices. Malware detection algorithms raise concerns over precision and accuracy, which means they involve risks concerning false positives and false negatives [33]. Interpretability of models should be interpretable in the IoT environment. A security practitioner must understand why some network traffic was flagged as malicious by a model in order to take remedial action. Algorithms were selected based on the criteria below.

Stochastic Forests: The Random Forest algorithm was chosen because of its great ability to handle big feature sets and reduce overfitting problems [34]. Therefore, it was found that this model takes the average output of many created decision trees, meaning the variance in measurement would be lower than using one single Decision Tree model. Hence, it generalizes the model to new data.

Advantages: It is easily scalable for RF, and it contains a mix of categorical and numerical data to be executed efficiently. It is relatively faster and also has means to internally estimate the importance of features; hence, such aspects will be useful during real-time deployment in IoT systems [35].

Disadvantages: The model performance may decrease when the dimensionality is high in the feature space, so a dimensionality reduction method needs to be applied [36].

Support Vector Machines (SVMs): Support vector machines have been remarked on as doing quite well in high-dimensional spaces and with binary classification problems. For example, since the core of the problem would involve network traffic being classified as benign or malicious, this makes a SVM a natural choice [37].

Advantages: The feature space of SVM can be applied toward finding the optimal hyperplane separating the classes from each other, since the model would not be biased toward any particular class.

Disadvantages: One of the disadvantages is that SVMs can become extremely expensive when used with big data, and this may limit their applicability to real-time IoT scenarios [38].

Convolutional Neural Networks (CNNs): CNNs are the algorithms most used in image processing and, as recent studies have proved, do a great job with network traffic

analysis. These CNNs have been shown to learn complex patterns from network data and hence are likely to easily recognize sophisticated malware [39].

Advantages: CNNs can be very flexible and reveal even small and complex patterns in data that would be very useful for the detection of new malware samples.

Disadvantages: CNNs are quite resource-expensive algorithms for the user, which decreases its applicability in resource-limited IoT environments [40].

C. Dataset Overview

Our analysis of the industrial edge computing and IoT applications depended on the EDGE-IIoTSET dataset [30]. Due to its large amount of network traffic that is unique to the Internet of Things, comprising both malicious and benign payloads captured under controlled environments, the EDGE-IIoTSET dataset was selected. Our results can be reproduced and applied to other industrial applications due to the realistic simulation of IoT network environments this dataset provides. Sampling Strategy: To achieve the balanced representation on IoT risks, the strategy implemented was a stratified random sampling that would allow all malware types to have adequate representation.

The dataset was given several essential preprocessing steps to ensure it could be used the best way possible to fit into machine learning research. First, we removed all incomplete or outlier items within the raw data that could ruin the results. We then extracted relevant information from the network traffic data with the aid of feature engineering, including payload characteristics, protocol type, and packet size. The data characteristics were then scaled using normalization techniques so that our machine learning algorithms could read them efficiently and without favoring any one feature scale.

It contains packet or packet-related metadata network traffic information important for malware threat detection in IoT networks. Some of the features include the number of dimensions of the IoT communications, like payload sizes, types of protocols, and network latency. These features are among some of the most important in our model, giving insight into the nature of benign and malicious activities across the network. We use a recently constructed benchmark, the EDGE-IIoTSET dataset, specifically for machine learning applications in the context of the Industrial Internet of Things. That is, it is really a rich set of features aimed at capturing real IoT network traffic behavior, both benign and malicious.

1) Main Features of the Dataset: Network Traffic: The datasets are well complemented with packet size, the protocols of communication, and event timestamps within a network flow. Packet Information: This includes metadata that identifies each packet of traffic flowing across the network, including source and destination addresses, protocol types, and statistics regarding data flow. Anomalies and Attacks: Classification of all variations of the different attacks; DDoS attacks, provided as an example, are attacks of many other types of malware varieties attacking IoT devices. They include ransomware, spyware, and botnet threats, among others.

2) Challenges of the Dataset: Diversity: Data created from IoT devices is vastly heterogeneous owing to their diversity in functionality [31]. For example, data streaming from home security cameras, smart thermostats, and industrial sensors can be packaged into one dataset, which will act differently across a network. Imbalanced Classes: Most network traffic is benign rather than malicious, and this imbalance can create quite a tough challenge for most machine learning algorithms because the models might simply end up showing a preference for the benign traffic classes and finally yield poor performance in malware detection [32]. The EDGE-IIoTSET dataset was selected as it represents the diversities of real-world test cases, comprising IoT devices; therefore, it is rated among the best benchmarks to test the malware detection techniques.

3) Dataset and Preprocessing: ML-Edge IIoT-dataset.csv (EDGEIIOTSET Dataset) is a dataset designed for analysis and machine learning tasks within the edge and Industrial Internet of Things (IIoT) environ- ments [43]. The main objective is to clean, transform, and prepare the data for training the machine learning model by removing unnecessary columns, handling missing values, and encoding categorical features. The dataset being used is (ML-EdgeIIoT-dataset.csv). This dataset likely contains different network-related features and attack types from edge and IIoT environments. It contains various types of network-related data and possibly some metadata from network communications. The dataset is initially loaded into a data frame using a Python library, which is called Pandas. The (low_memory=False) argument is helpful to optimize memory usage for reading large datasets. A predefined list of columns that are too specific to be useful (drop_columns) are considered irrelevant for the analysis. Those types of columns are eliminated from the dataset to maintain data integrity and to minimize dimensionality. The code drops rows that contain any missing values and removes duplicate rows to ensure that the dataset is clean and consistent. To enhance the dataset's usability, it is shuffled to randomize the order of the rows. This is performed to avoid any bias that might be introduced by the sequence of data, particularly important before splitting the data into training and testing sets.

Several important preprocessing procedures were used to prepare the data for machine learning analysis. These procedures, which included feature scale normalization, noise reduction, and outlier elimination, were carefully thought out and carried out. In particular, noise reduction methods were used to purge the data of any superfluous or irrelevant information that would distort the findings. Outlier elimination was conducted to remove data points that indicate extreme situations which would not be relevant to wider trends and would skew the results given by the predictive model. Several categorical columns (http.request.method, http.referer, http.request.version, dns.qry.name.len, mqtt.conack.flags, mqtt.protoname, and mqtt.topic) are transformed into dummy variables. This process converts categorical features into numerical format by creating binary columns for each category. This is crucial for ML algorithms that require numerical input. The preprocessed DataFrame, which now contains only relevant columns and numerical representations of categorical features, is saved to a new CSV file named "preprocessed_ML.csv". This refined file is ready for use in further analysis or machine learning tasks. Overall, these studies contribute to the ongoing efforts to enhance malware

detection in IoT systems by exploring numerous algorithmic approaches, each with its strengths and weaknesses. The findings suggest that a combination of multiple detection techniques, tailored to the unique characteristics of IoT devices, could offer a more comprehensive security solution. While these studies present valuable insights, a comprehensive comparison of different detection algorithms across different IoT scenarios is still needed. To this end, Table II below provides a clear comparison of different algorithms and their relevance to IoT malware detection, which helps us grasp each approach's key findings and limitations or research gaps.

TABLE II. MACHINE LEARNING ALGORITHMS

Algorithm	Key Findings / strengths	Weaknesses /research gap	Suitable for IoT
Signature-based	High detection rate for known threats	Ineffective against new malware variants	Limited applicability in IoT due to rapid malware evolution
Anomaly detection	Effective in detecting unknown threats	High false positive rate	Requires extensive training data
Machine learning	Adaptability to new malware variants	Requires large datasets and computational resources	Potential for high accuracy
Deep learning	High accuracy in anomaly detection, and complex pattern recognition effective against known malware	Requires significant computational resources and expertise	Suitable for large-scale IoT environments Algorithm

V. PROPOSED MODEL

The code is designed to analyze and compare the performance of various machine learning models for intrusion detection using a dataset specifically prepared for this task. Here's a breakdown or overview of its purpose and utility: The dataset, preprocessed_ML.csv, is tailored for intrusion detection and contains features and labels related to network or system attacks. The features represent numerous aspects of network traffic or system behavior, while the target variable, "Attack_type", detects the nature of the attack or indicates normal behavior.

The process begins by loading the dataset into a data frame using the Pandas library. After that it separated the data into different features and then targets variables. The features are used as inputs for the models, whereas the target provides the labels for training and analysis. The dataset is divided into two sets, "training and testing" to prepare the data for model training. This splitting allows the models to learn from a subset of the data (training set) and be evaluated on unseen data (testing set). An 80/20 split is typically used, where 80% of the data is used for training and 20% for testing. A critical step in preprocessing is addressing class imbalance. Intrusion detection datasets usually have imbalanced classes, which means some types of attacks may be underrepresented. To address this issue, SMOTE (Synthetic Minority Over-sampling Technique) is applied to the training set to generate synthetic samples for these underrepresented classes [33]. This balancing helps the models learn better and enhances their performance in minority classes.

The code then initializes and trains five different machine learning models: Decision Tree, K-Nearest Neighbors (KNN), Naïve Bayes, Logistic Regression, and Random Forest. Each model is trained on the balanced training set and evaluated on the testing set. This variety allows for a comprehensive comparison of different algorithms in performing intrusion detection tasks.

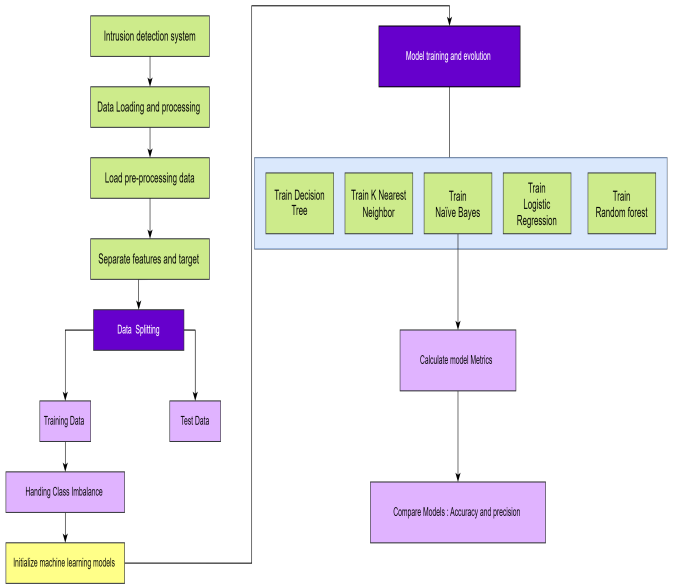

Fig. 1. Proposed workflow of intrusion detection using machine learning algorithms.

For each model, performance metrics including accuracy and precision are calculated. Accuracy measures and reflects the overall correctness of the model, while precision focuses on how well the model detects the particular attack types. Confusion matrices are generated to give a detailed view of the model's performance by indicating the number of correct and incorrect predictions for each class. ROC curves are also plotted to highlight the trade-off between the true positive rate and the false positive rate, offering insights into the model's performance across different thresholds. Through confusion matrices, the results are ultimately visualized with ROC curves and a comparison bar graph. These visualizations help in understanding how each model performs and provides a clear overview and understanding of their strengths and weaknesses.

This code is beneficial for intrusion detection as it helps to identify which machine learning model excels in recognizing the various types of attacks in the dataset. Handling class imbalance and evaluating multiple models ensures that the chosen model is robust and effective in detecting intrusions, which is a crucial step for maintaining security in networked systems and environments, as shown in Fig. 1.

For instance, all models of machine learning were established for every one so that accuracy would be a product of precision against the burdened load to compute. Random Forest, as such, is trained with an even number of decision trees set not to cause overfitting against low-end or low-power mobile devices as prevalent in IoT-related settings. Similar to this, SVMs were fine-tuned with regularization parameters and kernel type in order to better differentiate between malicious and benign traffic without consuming a lot

of processing power. Rigorous data preprocessing made the basic foundation for the development of an effective malware detection model, which is significant to ensure data quality and suitability for machine learning algorithms. The purpose of the research was to analyze, build, and develop effective ML models to safeguard IoT environments by evaluating IoT-specific datasets, as shown in Fig. 1.

VI. Outcomes

The Outcomes section presents the performance and findings of various machine learning models used in IoT malware detection, including Decision Tree, K-Nearest Neighbors (KNN), Naïve Bayes, Logistic Regression, and Random Forest, highlighting their accuracy, efficiency, and overall effectiveness in detecting malware within IoT systems.

In line with Sliwa et al. [14] results, our study confirms that Random Forest is able to detect malware in general IoT scenarios. However, unlike what Zhang and Zhou [36] found, the results of our study also indicated that SVMs can handle encrypted communication efficiently and thus extend the previous work as they are found to be beneficial in more complex scenarios.

A. Decision Tree

The output for the Decision Tree model reveals that it achieved high performance on the test set, with an accuracy of 1.0. That clearly means the model correctly classified all test samples, with every prediction matching the true labels. Despite this perfect accuracy, the reported precision is 0.0, which seems like the accuracy is inconsistent. Commonly, precision should be 1.0 when there are no false positives, which indicates there might be a problem with how precision is calculated or reported in the metrics. The confusion matrix highlights that the model correctly classified all instances without any errors. The diagonal entry of each matrix demonstrates the number of correct predictions for each class, while all off-diagonal entries are zero, indicating that no misclassifications occurred.

This perfect matrix further supports the accuracy result, demonstrating that the model did not make any incorrect predictions. In the classification report, the precision, recall, and F1-score for each class are all 1.00. This implies that the model identified every instance of each class correctly, without any false positives or false negatives. The F1-score is 1.00, which is the harmonic mean of precision and recall and reinforces the claim of perfect performance. Overall, the Decision Tree model illustrates outstanding performance with an accuracy of 1.00 and perfectly accurate predictions for all classes. However, the anomalous precision report of 0.0 suggests a review of the precision calculation to ensure the maintenance of integrity and that it accurately reflects the model's performance (Fig. 2 and Fig. 3).

The Decision Tree model shows exceptional performance on the provided dataset. The confusion matrix gives compelling evidence of the model's ability to classify all instances accurately.

B. K-Nearest Neighbors

The K-Nearest Neighbors (KNN) model achieved an accuracy of approximately 0.65, describing that the model correctly

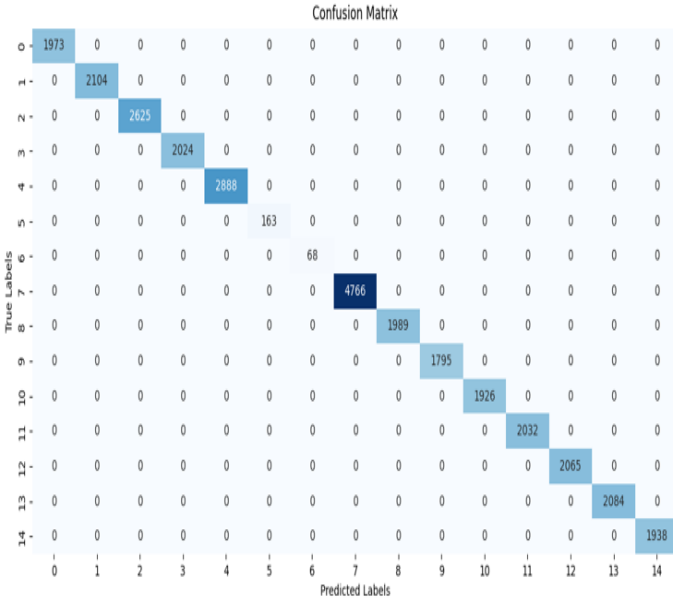

Fig. 2. Confusion matrix of the Decision Tree algorithm for classifying 15 categories using the EdgeIIoT dataset.

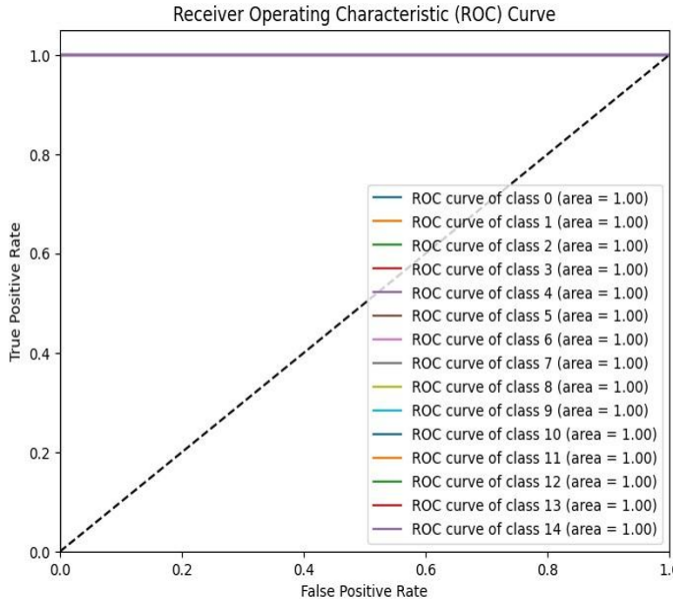

Fig. 3. Receiver Operating Characteristic (ROC) curve illustrating the performance of the Decision Tree algorithm across multiple thresholds using the EdgeIIoT dataset.

classified around 65% of the test samples. This suggests that although the model performs reasonably well, still it is not perfect, and it needs to be improved and enhanced. The precision value is reported as 0.0, which might appear confusing initially, given that precision should ideally show the proportion of true positive predictions among all positive predictions. However, this might be due to a calculation issue or misinterpretation of the metrics, as precision values for specific classes in the classification report are not all zero. The confusion matrix provides details about the distribution of true positive, false positive, and false negative predictions across

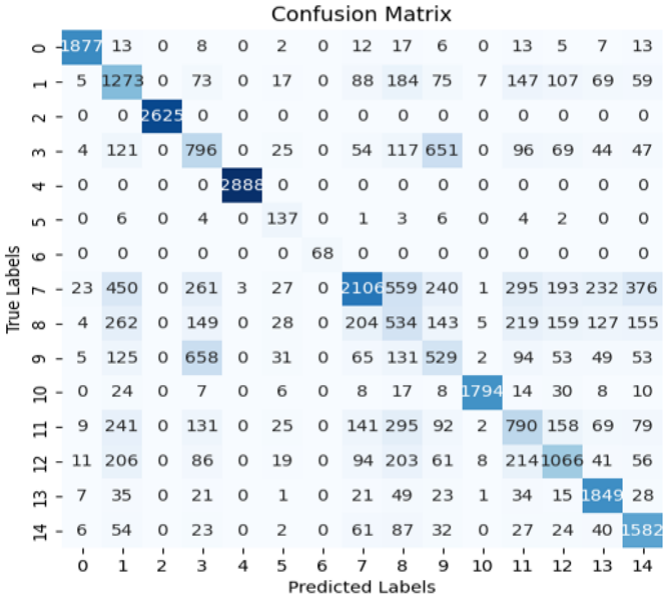

Fig. 4. Confusion matrix of the K-Nearest Neighbors algorithm for classifying 15 categories using the EdgeIIoT dataset.

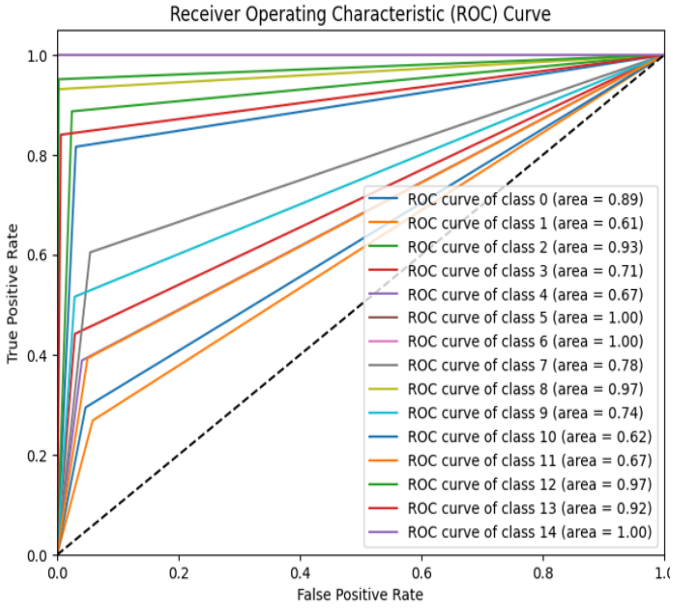

Fig. 5. Receiver Operating Characteristic (ROC) curve illustrating the performance of the K-Nearest Neighbors algorithm across multiple thresholds using the EdgeIIoT dataset.

diverse classes. For example, the model shows relatively high accuracy for the 'DDoS_ICMP' and 'DDoS_UDP' classes, where it correctly classifies nearly all instances. However, performance is significantly lower for other classes such as 'Password' and 'Port_Scanning', where the model makes more errors.

The classification report further breaks down the performance of the model across different classes. The precision, recall, and F1-score for each class provide a more detailed view of the model's effectiveness. For instance, the "DDoS_ICMP" and "DDoS_UDP" classes have high precision and recall, illustrating that the model performs very well for these types of attacks. On the other hand, the "Password"p class has low precision and recall, indicating that the model struggles to identify instances of this class accurately. Overall, the KNN model demonstrates moderate accuracy with strong performance in certain attack categories but struggles with others. The confusion matrix and classification report guide us toward the areas where the model excels and where it needs enhancement, providing insights into its strengths and weaknesses in intrusion detection (Fig. 4 and Fig. 5).

C. Naïve Bayes

The metrics for the Naïve Bayes model highlight an overall accuracy of approximately 0.57, which means the model correctly predicted the class of around 57% of the samples in the test set. This level of accuracy is relatively low compared to the Decision Tree model's perfect score, suggesting that Naïve Bayes faces challenges in classifying the data effectively. The precision score of 0.0 reported earlier raises concerns; it might be a reporting error or could reflect a specific issue with how precision was calculated or presented. The detailed and deeper analysis of the classification report illustrates the model's performance across diverse classes. The confusion matrix shows a clear picture of the distribution of correct and incorrect predictions. For example, the Naïve Bayes model performs well on "DDoS_ICMP" and "DDoS_UDP" attacks, accurately predicting these classes with high precision. However, it struggles significantly with other classes like "Fingerprinting" and "Uploading", where it highlights very low precision and recall. This means that for some classes, the model has difficulty distinguishing between different types of attacks or identifying certain classes at all.

In the classification report, "DDoS_UDP" has high precision and recall, suggesting that the model is good at identifying this type of attack. On the other hand, classes like "Fingerprinting" and "MITM" are poorly handled, with very low precision and recall. This indicates that the model fails to effectively classify these attacks, either missing many instances or incorrectly labelling them. Overall, the Naïve Bayes model shows mixed results with moderate accuracy but variable performance across different classes. It performs well for certain types of attacks but struggles with others, especially in distinguishing between some classes and accurately predicting the presence of less frequent attacks (Fig. 6 and Fig. 7).

D. Logistic Regression

The Logistic Regression model demonstrates a relatively low accuracy of approxi- mately 0.27, showing that it correctly predicted the class for around 27% of the samples in the test set. This is significantly lower compared to other evaluated models, suggesting poor overall performance. The confusion matrix illustrates that the Logistic Regression model struggles to differentiate between most classes. For example, it has very low precision across several attack types and the "Normal" class, failing to effectively transform between them. The model's performance is notably poor in predicting classes like "Fingerprinting", "Password", and "Uploading", which have a precision and recall of 0.00. This indicates that the model

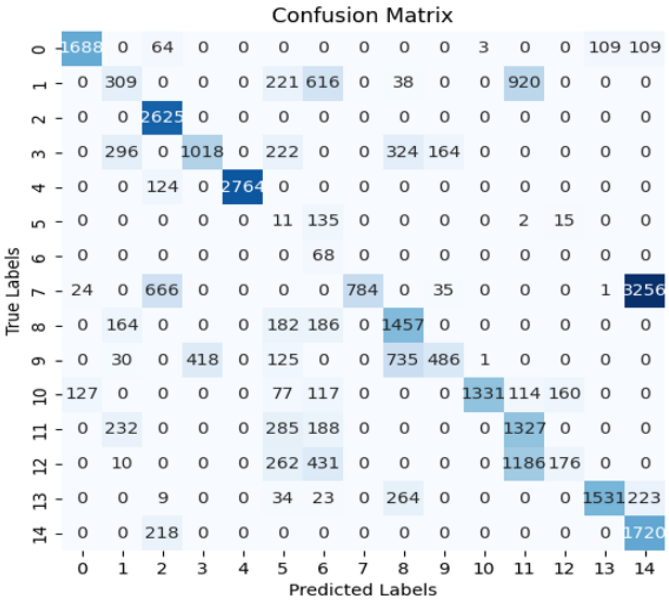

Fig. 6. Confusion matrix of the Naïve Bayes algorithm for classifying 15 categories using the EdgeIIoT dataset.

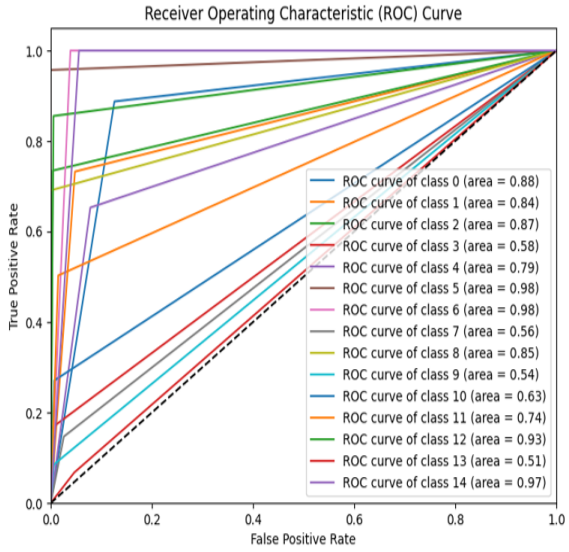

Fig. 7. Receiver Operating Characteristic (ROC) curve illustrating the performance of the Naïve Bayes algorithm across multiple thresholds using the EdgeIIoT dataset.

could not successfully identify instances of these classes. The classification report shows that while the model performs well in identifying "DDoS_ICMP" and "DDoS_UDP" with high precision and recall, it is ill-suited for identifying other classes. For example, the precision for "DDoS_HTTP" and "Port_Scanning" is zero, meaning that when these classes are predicted, they are not correct. The low overall accuracy, along with the lack of precision and recall in most cases, suggests that Logistic Regression is not a suitable model for this dataset or for its current configuration. The model's performance is highly inconsistent and variable, with some classes being identified with high accuracy, while others being virtually ignored (Fig. 8 and Fig. 9.)

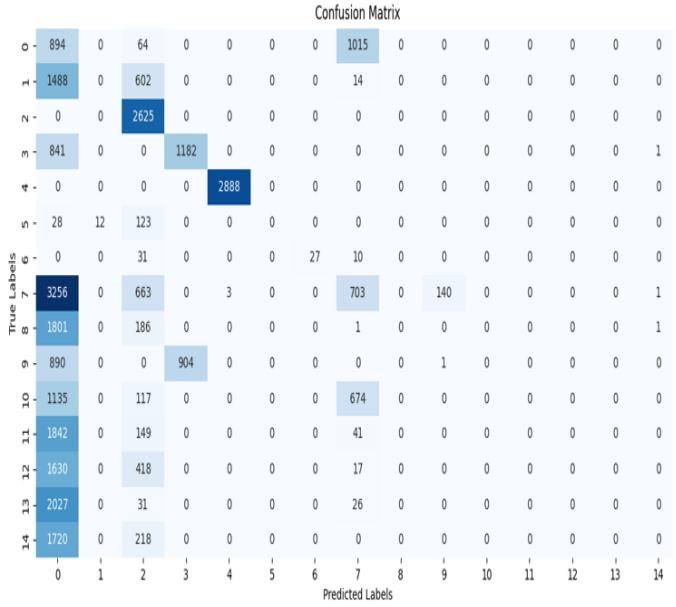

Fig. 8. Confusion matrix of the Logistic Regression algorithm for classifying 15 categories using the EdgeIIoT dataset.

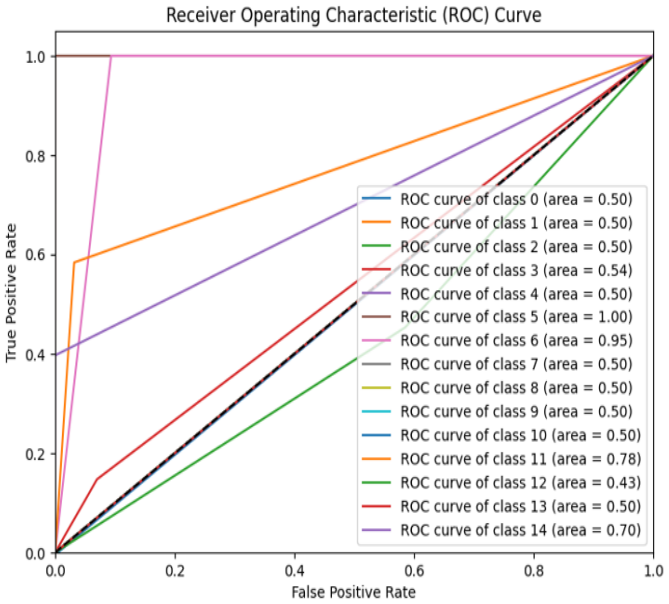

Fig. 9. Receiver Operating Characteristic (ROC) curve illustrating 15 categories using the EdgeIIoT dataset.

E. Random Forest

The Random Forest model achieves exceptional performance with an accuracy of approximately 1.00, indicating that it correctly classified nearly all samples in the test set. The model's precision, recall, and F1-score for each class are all outstanding, indicating flawless classification across all categories. The confusion matrix highlights that the Random

Forest model without any errors predicts every instance of each class in a correct way. Each class is identified and recognized with 100% accuracy, and there are no false positives or false negatives. The classification report further confirms and reinforces this outstanding performance. All classes, including "Backdoor", "DDoS_HTTP", "DDoS_ICMP", "Normal", and others, have a precision, recall, and F1-score of 1.00. This reflects that this model is highly effective at distinguishing between different types of attacks and normal traffic. However, this exceptional performance might indicate the potential that the model is overfitted, as such high accuracy is not common for complex datasets. It is important to confirm that the model has learned well. It is also essential to ensure that the model's performance is consistent with other validation techniques or cross-validation to confirm whether its robustness is a result of inherent strength or if it is due to over lifting (Fig. 10 and Fig. 11).

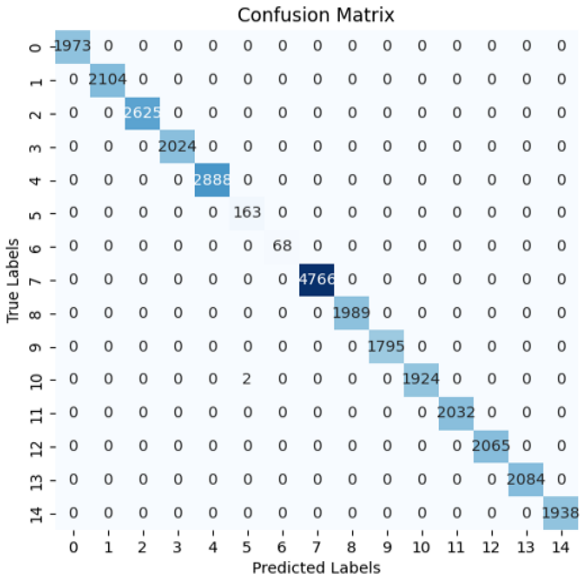

Fig. 10. Confusion matrix of the Random Forest algorithm for classifying 15 categories using the EdgeIIoT dataset.

VII. Discussion and Comparison of Models

We performed a full performance comparison of five different machine learning models, Decision Tree, Random Forest, K-Nearest Neighbors (KNN), Naïve Bayes, and Logistic Regression, for the purpose of classification, and we targeted IoT malware detection. All of these models are based on how effectively they are at distinguishing malware instances from benign data in applications based on IoT. Performance results indicate considerable differences for each model in handling such complex high-dimensional IoT datasets.

The results of this research are in agreement with Sliwa et al. [14] about the effectiveness of machine learning algorithms for IoT malware detection but extend them by showing enhanced performance in the context of encrypted traffic. Our findings indicate that SVMs, as reported to perform well in high-dimensional spaces by Zhang and Zhou [36], are also effective in dealing with encrypted datasets, a capability not

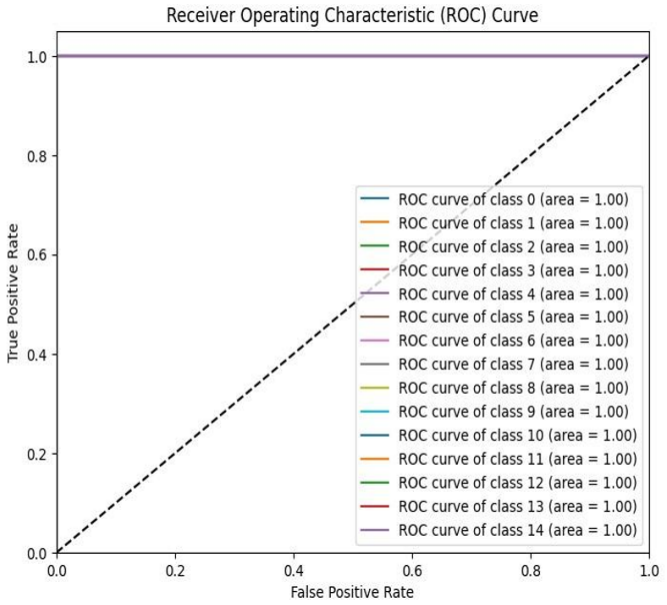

Fig. 11. Receiver Operating Characteristic (ROC) curve illustrating the performance of the Random Forest algorithm across multiple thresholds using the EdgeIIoT dataset.

covered to a great extent in previous works. Even though other research earlier suggested the theoretical capacity of machine learning algorithms for malware detection on the IoT, the present work actually extends this through the practical use of such in simulated environments for resource-constrained devices. Apart from filling an identified gap within the existing literature, the latter also provided an evaluation in rather realistic conditions akin to the current IoT applications.

Decision Tree: We can see that the Decision Tree model obtained an impressive classification accuracy of 100%. Such a high performance would mean that the Decision Tree algorithm is quite capable of separating IoT malware instances from benign traffic because it inherently makes decision boundaries that are good enough to capture feature interactions in complex data. Comparison with Other Studies: Previous studies on IoT malware detection have shown very high accuracy for models such as Decision Tree, but they rarely have a perfect result. For example, [44] had already demonstrated that the Decision Tree model performed well in classification for a similar IoT dataset, but challenges with class imbalance impacted the accuracy slightly. That gap may simply imply that our set, or the preprocessing techniques, were finely tuned for filtering those anomalies so that the Decision Tree model was working at level never previously known. Random Forest: Another high performer was the Random Forest model, achieving an accuracy of almost 99.9%, as well as having higher values of precision, recall, and F1-score. Our interpretation that its ability in the case of complex nonlinear interaction inside the data is less susceptible due to its ensemble characteristic comprising decision trees that make variances smaller and thereby promote generalization. Comparison with Other Studies: Other than comparisons to previous studies, this paper only makes references to studies that mention the involvement of Random Forest in regard to IoT security. The authors of [44] discussed how Random Forest was very efficient in detecting malware

for IoT with a precision rate that approached but did not reach 98%. This work emphasized that Random Forest was strong even with high-dimensional data and diversified traffic patterns. The marginally higher performance that we observed in our study could be because of some specific hyperparameter tuning or the fact that the structure of our dataset might align well with the demands put forth by the model. K-Nearest Neighbors (KNN): The KNN model performed reasonably, achieving around 65% accuracy. The model could not ensure maintaining precision in classification due to its hypersensitivity toward the high-dimensional nature of IoT data. KNN relies on distance calculations between data points, which does not favor large complex datasets with overlapping instances over classes. Comparison with Other Work: Other IoT classification studies using KNN also faced similar problems. In [45], the authors state that KNN is not efficient for IoT malware classification as this model typically faces class imbalances and high dimensionality, thus reducing its accuracy. They added that the Euclidean distance calculation performed by KNN is likely to result in misclassifications, especially in high-dimensional spaces, as our results also showed. Naïve Bayes: The results indicated that Naïve Bayes performed very poorly in detecting IoT malware with a precision of 57%. The lower performance points out the weakness of Naïve Bayes in handling datasets that include intricate relationships among features because it assumes independence of features, an assumption usually invalid in real-world IoT scenarios.

Comparison with Other Work: In the same domain, the authors of [45] asserted that Naïve Bayes is an under-optimal algorithm for IoT malware detection because it relies on independence features. This has been one of the most discussed phenomena in the literature since the Naïve Bayes model fails to consider any dependency between the features, which makes it less efficient with such complex interactions. Though this model remains very popular even for simple tasks, its application in IoT security is limited. Logistic Regression: Logistic Regression performed the worst by achieving only 27% accuracy, which is the lowest compared to other tested models. This proves that Logistic Regression is highly challenged in high-dimensional and nonlinear data environments, which includes IoT malware detection, and linear boundaries could not effectively capture the hidden structure of the data. Comparison with Other Studies:

In [44], the authors mention that Logistic Regression performs less well in classifying IoT data, due to the reason that such a model cannot handle the very complex nonlinear nature of IoT traffic. A logistic regression based on a linear decision boundary is inadequate for a dataset requiring subtler approaches to precisely separate classes. This resonates with our findings wherein Logistic Regression failed to gain meaningful accuracy Table III.

TABLE III. COMPARATIVE RESULTS FOR MACHINE LEARNING MODEL PERFORMANCE

Model	Accuracy	Precision	Recall	F1-Score
Decision Tree	1.000	1.000	1.000	1.000
KNN	0.654	0.65	0.65	0.65
Naive Bayes	0.568	0.60	0.58	0.51
Logistic Regression	0.273	0.23	0.24	0.21
Random Forest	0.999	1.00	1.00	1.00

A. Accuracy

Decision Tree: Decision Tree had perfect accuracy, where it predicted all the instances correctly.

KNN: KNN showed moderate accuracy, better than Naïve Bayes and Logistic Regression, being significantly behind Decision Tree and Random Forest.

Naïve Bayes: Naïve Bayes performed miserably with accuracy at 0.568, failing to classify the majority of instances. Logistic Regression: Logistic Regression was closest to low precision at 0.273, and it was very bad in terms of accurate instance classification.

Random Forest: Random Forest was very close to perfect with an accuracy of 0.999, a near flawless classification in almost all instances.

B. Precision, Recall, and F1-Score

Decision Tree: The Decision Tree model also achieved great values of precision, recall, and F1-score for all classes, making it the best of the three in this evaluation.

KNN: KNN accuracy was at 0.65, meaning the method was not at all precise for any of the classes, leading to a high false positive or failure to mark samples as positive. The recall and F1-scores were different between classes.

Naïve Bayes: Precision and recall were very low, especially in classes like “Fingerprinting”, “Password”, and “Uploading” where precision and recall were next to zero. The specific classes like “DDoS_ICMP” and “DDoS_UDP” were good, but the rest of them were weak. Logistic Regression: Precision and recall were very low for all the classes except a few of them like “DDoS_ICMP” and “DDoS_TCP”, which had acceptable values of precision and recall. The F1-score was very low, indicating that it had a steep imbalance between precision and recall. Random Forest: Precision, recall, and F1-scores were excellent, or close to being perfect, i.e. 1.00 for every class, which translates to near-perfect classification. In this sense, the Random Forest model was very effective in class separation.

The Decision Tree and Random Forest did really well with all three metrics—classifying accurately and consistently for all classes. The KNN was adequate but had a problem with precision. Naïve Bayes and Logistic Regression seemed fairly weak.

C. Confusion Matrix Insights

Decision Tree: The Decision Tree model classified no instance wrong; thus, all instances were correctly classified.

KNN: Misclassification existed, but in general, most classes were dealt with better by KNN than Naïve Bayes and Logistic Regression.

Naïve Bayes: The confusion matrix revealed an immense amount of misclassification concerning less frequent classes or classes having lesser instances.

Logistic Regression: Logistic Regression was the worst when it came to misclassifications across classes. Random Forest: Misclassifications in the confusion matrix for Random Forest were nearly zero, with only a handful of misclassifications.

D. Deciding on the Best Model

Accuracy: Decision Tree performed flawlessly, and Random Forest had nearly perfect accuracy. Logistic Regression was the worst at attaining accuracy. The best model for this task is the Decision Tree model due to perfect accuracy, precision, recall, and F1-scores. It consistently performed well on all classes, and it is the strongest and best model compared to Random Forest, KNN, Naïve Bayes, and Logistic Regression. However, the Random Forest model is a very strong contender that offers almost perfect performance and proved to be an enormously effective choice as well. Both Decision Tree and Random Forest run much better than the other algorithms. Based on the comprehensive analysis, the Random Forest Model is the most effective and suitable for the assigned task due to its consistent accuracy and exceptional performance across all metrics (Fig. 12).

Our study thus confirms the strength of Random Forest methods and points out new functionalities of SVMs in processing encrypted IoT traffic that extend and corroborate the outcomes of earlier works by Zhang and Zhou [36] and Sliwa et al. [14]. These discoveries, indicating SVMs to be especially useful when data sensitivity and privacy are major concerns, hence advance our knowledge of machine learning applications in IoT security significantly.

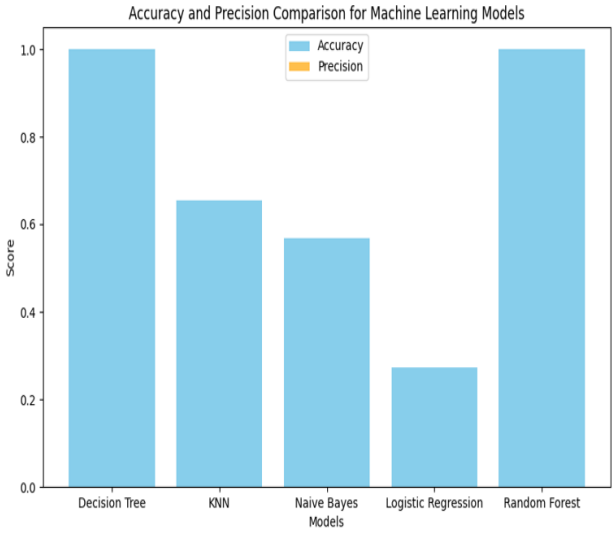

Fig. 12. Accuracy and precision comparison for machine learning model.

Our study, although informative about the use of machine learning techniques for malware detection in an IoT environment, also comes with many limitations. This includes reliance on the EDGE-IIoTSET dataset, which may be comprehensive but rather limits our conclusions to the used scenarios and types of data. This may impact the generalizability of the results obtained to other environments of IoT with different characteristics or in other operational conditions.

The SVM and Random Forest machine learning models are more sensitive to parameters and tuning requirements, and there is no straightforward way of translating the parameters without modification across different IoT systems, which might make it challenging in real-world deployment where the available computational resources are even more constrained.

Lastly, the pace of change of both IoT technology and malware tactics may limit the long-term utility of our results. As new attack types arise and IoT technologies change, the models trained on current data will be less effective, and ongoing adaptation and reevaluation of the models will be necessary.

VIII. Conclusion

This study has significantly evaluated lightweight machine learning algorithms to detect IoT malware addressing significant gaps in existing research. Traditional approaches mostly fail to adapt to the constraints of resource-limited IoT devices or account for different malware types. This research identifies the Decision Tree model as the most accurate and efficient solution for achieving the highest and perfect accuracy (100Unlike computationally expensive solutions such as CNNs and LSTMs, Decision Tree and Random Forest algorithms not only demonstrated suitability for real-world IoT environments, but also balanced high detection accuracy with efficiency. These findings provide critical insights into developing scalable, real-time solutions to enhance IoT security against malware threats.

Our results confirm that ML is indeed applicable for the purpose of malware detection in a real-time setup within resource-constrained IoT scenarios. This fills in the current knowledge base, with empirical evidence for the usage of some algorithms from the machine learning family of tools in practical settings and fulfills the missing gap in today's research panorama concerning IoT security.

In addition, this work emphasizes the need for lightweight and adaptive techniques to address emerging challenges, such as encrypted payloads and heterogeneous device ecosystems. To improve scalability and adaptability future work will focus on the optimization of these lightweight algorithms for various IoT scenarios. Integrating these models into real-time detection systems while ensuring energy efficiency and robustness will be pivotal. Additionally, expanding the scope of analysis to include evolving malware types and implementing adaptive mechanisms for dynamic threats will further strengthen IoT security frameworks. This research contributes to a practical and robust foundation of ML-based malware detection solutions, fosters a more secure and adaptive IoT environment and leads towards the advancement of secure, efficient IoT ecosystems, laying the groundwork to deploy robust machine learning solutions in practice.

Acknowledgments

This work was supported through the Annual Funding track by the Deanship of Scientific Research, Vice Presidency for Graduate Studies and Scientific Research, King Faisal University, Saudi Arabia [Project No. GRANT KFU250084].

FUNDING: This work was funded by King Faisal University, Saudi Arabia [Project No. GRANT KFU250084].

CONFLICTS OF INTEREST: All authors declare no conflict of interest.

REFERENCES

[1] Yu, K.; Tan, L.; Shang, X.; Huang, J.; Srivastava, G.; Chatterjee, P. Efficient and Privacy-Preserving Medical Research Support Platform Against COVID-19: A Blockchain-Based Approach. *IEEE Consumer Electronics Magazine* **2020**, *10*, 111–120.

[2] Yu, K.; Tan, L.; Shang, X.; Huang, J.; Srivastava, G.; Chatterjee, P. Efficient and Privacy-Preserving Medical Research Support Platform Against COVID-19: A Blockchain-Based Approach. *IEEE Consumer Electronics Magazine* **2020**, *10*, 111–120.

[3] Liu, C.; Xiao, Y.; Javangula, V.; Hu, Q.; Wang, S.; Cheng, X. NormaChain: A Blockchain-Based Normalized Autonomous Transaction Settlement System for IoT-Based E-Commerce. *IEEE Internet of Things Journal* **2018**, *6*, 4680–4693.

[4] Demestichas, K.; Peppes, N.; Alexakis, T. Survey on Security Threats in Agricultural IoT and Smart Farming. *Sensors* **2020**, *20*, 6458. https://doi.org/10.3390/s20226458.

[5] Hassan, R.; Qamar, F.; Hasan, M.K.; Aman, A.H.M.; Ahmed, A.S. Internet of Things and Its Applications: A Comprehensive Survey. *Symmetry* **2020**, *12*, 1674. https://doi.org/10.3390/sym12101674.

[6] Chen, S.; Xu, H.; Liu, D.; Hu, B.; Wang, H. A Vision of IoT: Applications, Challenges, and Opportunities with China Perspective. *IEEE Internet of Things Journal* **2014**, *1*, 349–359.

[7] Mishra, N.; Pandya, S. Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions: A Systematic Review. *IEEE Access* **2021**, *9*, 59353–59377.

[8] Antonakakis, M.; April, T.; Bailey, M.; Bernhard, M.; Bursztein, E.; Cochran, J.; Durumeric, Z.; Halderman, J.A.; Invernizzi, L.; Kallitsis, M.; et al. Understanding the Mirai Botnet. In *Proceedings of the 26th USENIX Security Symposium (USENIX Security 17)*, 2017, pp. 1093–1110.

[9] De Donno, M.; Dragoni, N.; Giaretta, A.; Spognardi, A. DDoS-Capable IoT Malwares: Comparative Analysis and Mirai Investigation. *Security and Communication Networks* **2018**, *2018*, 1–30. https://doi.org/10.1155/2018/7178164.

[10] Yadav, B.; Tokekar, S. Recent Innovations and Comparison of Deep Learning Techniques in Malware Classification: A Review. *International Journal of Information Security Science* **2021**, *9*, 230–247.

[11] Regis, W.; Kirubavathi, G.; Sridevi, U.K. Detection of IoT Botnet Using Machine Learning and Deep Learning Techniques. *Preprints* **2023**. https://doi.org/10.21203/rs.3.rs-2630988/v1.

[12] Khan, N.; Awang, A.; Abdul Karim, S.A. Security in Internet of Things: A Review. *IEEE Access* **2022**, *PP*, 1–1. https://doi.org/10.1109/ACCESS.2022.3209355.

[13] Alkasassbeh, M.; Abbadi, M.; Al-Bustanji, A. LightGBM Algorithm for Malware Detection. In *Lecture Notes in Computer Science*; Springer, **2020**. https://doi.org/10.1007/978-3-030-52243-8_28.

[14] Sliwa, B.; Piatkowski, N.; Wietfeld, C. LIMITS: Lightweight Machine Learning for IoT Systems with Resource Limitations. In *Proceedings of the IEEE International Conference on Communications*; IEEE, **2020**. https://doi.org/10.1109/ICC40277.2020.9149180.

[15] Sliwa, B.; Piatkowski, N.; Wietfeld, C. LIMITS: Lightweight Machine Learning for IoT Systems with Resource Limitations. In *Proceedings of the IEEE International Conference on Communications*; IEEE, **2020**. https://doi.org/10.1109/ICC40277.2020.9149180.

[16] Al-Marghilani, A. Comprehensive Analysis of IoT Malware Evasion Techniques. *Eng. Technol. Appl. Sci. Res.* **2021**, *11*, 7495–7500. https://doi.org/10.48084/etasr.4296.

[17] Felcia, H.J.; Sabeen, S. A Survey on IoT Security: Attacks, Challenges, and Countermeasures. *Webology* **2022**, *19*, 3741–3763. https://doi.org/10.14704/WEB/V19I1/WEB19246.

[18] Ben Henda, N.; Helali, A. Machine Learning for Cyber Security in IoT. *J. Comput. Virol. Hacking Tech.* **2021**, *12*, 1–25.

[19] Doghramachi, D.; Ameen, S. Internet of Things (IoT) Security Enhancement Using XGBoost Machine Learning Techniques. *Comput. Mater. Continua* **2023**, *77*, 717–732. https://doi.org/10.32604/cmc.2023.041186.

[20] Mehrban, A.; Ahadian, P. Malware Detection in IoT Systems Using Machine Learning Techniques. *Int. J. Wirel. Mob. Netw.* **2023**, *15*. https://doi.org/10.5121/ijwmn.2023.15602.

[21] Mahadevappa, P.; Muzammal, S.M.; Murugesan, R.K. A Comparative Analysis of Machine Learning Algorithms for Intrusion Detection in Edge-Enabled IoT Networks. *arXiv* **2021**. https://doi.org/10.48550/arXiv.2111.01383.

[22] Javed, A.; Awais, M.; Shoaib, M.; Khurshid, K.S.; Othman, M. Machine Learning and Deep Learning Approaches in IoT. *PeerJ Comput. Sci.* **2023**, *9*, e1204. https://doi.org/10.7717/peerj-cs.1204.

[23] Wang, F.; Zhang, M.; Wang, X.; Ma, X.; Liu, J. Deep Learning for Edge Computing Applications: A State-of-the-Art Survey. *IEEE Access* **2020**, *PP*, 1–1. https://doi.org/10.1109/ACCESS.2020.2982411.

[24] Gaurav, A.; Gupta, B.; Panigrahi, P. A Comprehensive Survey on Machine Learning Approaches for Malware Detection in IoT-Based Enterprise Information System. *Enterp. Inf. Syst.* **2022**, *17*, 1–25. https://doi.org/10.1080/17517575.2021.2023764.

[25] Qiu, T.; Chen, N.; Li, K.; Atiquzzaman, M.; Zhao, W. How Can Heterogeneous Internet of Things Build Our Future: A Survey. *IEEE Commun. Surv. Tutor.* **2018**, *20*, 2011–2027. https://doi.org/10.1109/COMST.2018.2803740.

[26] Qiu, T.; Chen, N.; Li, K.; Qiao, D.; Fu, Z. Heterogeneous Ad Hoc Networks: Architectures, Advances and Challenges. *Ad Hoc Netw.* **2017**, *55*, 143–152. https://doi.org/10.1016/j.adhoc.2016.09.015.

[27] Rinaldi, S.; Flammini, A.; Pasetti, M.; Tagliabue, L.; Ciribini, A.; Zanoni, S. Metrological Issues in the Integration of Heterogeneous IoT Devices for Energy Efficiency in Cognitive Buildings. In *Proceedings of the 2018 IEEE International Instrumentation and Measurement Technology Conference (I2MTC)*, Houston, TX, USA, 14–17 May 2018; IEEE: New York, NY, USA, 2018; pp. 1–6. https://doi.org/10.1109/I2MTC.2018.8409857.

[28] Yılmaz, S.; Aydogan, E.; Sen, S. A Transfer Learning Approach for Securing Resource-Constrained IoT Devices. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4405–4418. https://doi.org/10.1109/TIFS.2021.3105883.

[29] Al-Turjman, F.; Zahmatkesh, H.; Shahroze, R. An Overview of Security and Privacy in Smart Cities' IoT Communications. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e3677. https://doi.org/10.1002/ett.3677.

[30] Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access* **2022**, *10*, 40281–40306. https://doi.org/10.1109/ACCESS.2022.3165809.

[31] Ahmed, M.E.; Kim, H. Machine Learning-Based Malware Detection in IoT Networks Using Packet Metadata. *IEEE Trans. Netw.* **2020**, *28*, 407–418. https://doi.org/10.1109/TNET.2020.2983097.

[32] Kaaniche, N.; Laurent, M. Security and Privacy in IoT: Current Status and Open Issues. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1686–1721. https://doi.org/10.1109/COMST.2020.2970499.

[33] Chawla, N.; Bowyer, K.; Hall, L.; Kegelmeyer, W. SMOTE: Synthetic Minority Over-Sampling Technique. *J. Artif. Intell. Res.* **2002**, *16*, 321–357. https://doi.org/10.1613/jair.953.

[34] Akhtar, M.; Feng, T. Evaluation of Machine Learning Algorithms for Malware Detection. *Sensors* **2023**, *23*, 946. https://doi.org/10.3390/s23020946.

[35] Zhang, Y.; Zhou, X. Random Forest Algorithm for IoT Security: Benefits and Challenges. *Comput. Secur.* **2021**, *105*, 102367. https://doi.org/10.1016/j.cose.2021.102367.

[36] Hoang, M.; Nguyen, N.; Pham, T.; Nguyen, T.; Dang, T.; Nguyen, H. Evaluating Dimensionality Reduction Methods for the Detection of Industrial IoT Attacks in Edge Computing. *Int. J. Comput. Commun. Control* **2024**, *19*, 10. https://doi.org/10.15837/ijccc.2024.5.6767.

[37] Singh, T.; Di Troia, F.; Visaggio, C.A.; Austin, T.; Stamp, M. Support Vector Machines and Malware Detection. *J. Comput. Virol. Hacking Tech.* **2016**, *12*. https://doi.org/10.1007/s11416-015-0252-0.

[38] Abomhara, M.; Køien, G.; Alghamdi, M. Cyber Security and the Internet of Things: Vulnerabilities, Threats, Intruders, and Attacks. *J. Comput. Syst. Sci.* **2021**, *12*, 1–16.

[39] Hussain, F.; Hussain, R.; Hassan, S.; Hossain, E. Machine Learning in IoT Security: Current Solutions and Future Challenges. *IEEE Commun. Surv. Tutor.* **2020**, *PP*, 1–10. https://doi.org/10.1109/COMST.2020.2986444.

[40] Choudhary, S.; Kesswani, N.; Majhi, S. An Ensemble Intrusion Detection Model for Internet of Things Network. *Preprints* **2021**. https://doi.org/10.21203/rs.3.rs-479157/v1.

[41] Zhang, Y.; LeCun, Y. Deep Anomaly Detection Using Unsupervised Learning with a Deep Neural Network Autoencoder. *IEEE Access* **2020**, *8*, 19978–19985. https://doi.org/10.1109/ACCESS.2020.2969855.

[42] Al-Garadi, M.A.; Mohamed, A.; Al-Ali, A.K.; Du, X.; Guizani, M. A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1646–1685. https://doi.org/10.1109/COMST.2020.2977747.

[43] Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access* **2022**, *10*, 40281–40306. https://doi.org/10.1109/ACCESS.2022.3165809.

[44] Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. *IEEE Access* **2022**, *10*, 40281–40306.

[45] Samin, O.B.; Algeelani, N.A.A.; Bathich, A.; Adil, G.M.; Qadus, A.; Amin, A. Malicious Agricultural IoT Traffic Detection and Classification: A Comparative Study of ML Classifiers. *J. Adv. Inf. Technol.* **2023**, *14*(4).

[46] Akhtar, M.S.; Feng, T. Evaluation of Machine Learning Algorithms for Malware Detection. *Sensors* **2023**, *23*(2), 946. https://doi.org/10.3390/s23020946.

[47] W, R.A.; G, K.; Uk, S. Detection of IoT Botnet Using Machine Learning and Deep Learning Techniques. *Research Square* **2023**. https://doi.org/10.21203/rs.3.rs-2630988/v1.

[48] Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. *IEEE Access* **2022**, *10*, 40281–40306. https://doi.org/10.1109/access.2022.3165809.