Intelligent IoT Network Malware Classification using Real-time Heterogenous Data

Aqeel Ahmed

Main academic Supervisor: Dr. Christos TJORTJIS, International Hellenic University.

A Master Thesis submitted for the Erasmus Mundus Joint Master Degree on Smart Cities and Communities (SMACCs)

June 2022

University of Mons, Heriot Watt University, International Hellenic University, University of the Basque Country

Acknowledgements

Firstly, I would like to thank God Almighty, WHO blessed me with health and motivation to complete this report. Secondly, I would like to acknowledge my parents. They always supported and motivated me from my home country during this dissertation.

I would like to acknowledge and appreciate my supervisor **Prof. Christos Tjortjis** (Dean School of Science and Technology, & local coordinator SMACCS, IHU) for his continuous guidance and support.

In addition, I would like to thank the SMACCS consortium for sponsoring my participation to the conference for presenting the paper which resulted from dissertation.

Finally, I would like to acknowledge my brothers and friends who kept motivating me through phone calls during this period of thesis

Dedication

I would like to dedicate this work to my dearest elder brother*, (Late) Tariq Hussain Hussaini*, whom we lost during the covid times, and I could not travel back home to his funeral. He had always supported and motivated me throughout my academic life.

I will always miss him.

Acronyms

AI	Artificial Intelligence
ANN	Artificial Neural Network
DL	Deep Learning
DT	Decision Tree
FPR	False Positive Rate
GNB	Gaussian Naïve Bayes
IoT	Internet of Things
ITU	International
IEEE	Institute of Electrical Electronics Engineering
IETF	International Engineering Task Force
KNN	K-Nearest Neighbors
LR	Logistic Regression
ML	Machine Learning
RF	Random Forest
TPR	True Positive Rate
XGB	Extreme Gradient Boosting

Abstract

Due to its wide range of applications, the Internet of Things (IoT) technology is evolving rapidly. One can witness IoT systems in smart cities, smart homes, smart healthcare, smart industry, and smart agriculture. IoT systems usually use low-powered and low-memory devices to sense the data from the environment and transmit it to the destination through wired or wireless communication channels. Although IoT technology is gaining massive attention in every sector of life, the security of these devices is one of the biggest challenges. Due to resource constraints, these devices are often vulnerable to malicious actors. In this work, a machine learning-based intelligent classification of the IoT network attacks using real-time heterogenous data is carried out. Two IoT network malware datasets (Ton-IoT & IoT-23) that include the real-time IoT Botnet attacks are used for the experiments. The data is pre-processed before performing the experimentation. In addition, a information gain based feature selection method is also applied to select the most important features in the dataset. Several classification methods include Logistic Regression (LR), Decision Tree (DT), Random Forest (RF), K-Nearest Neighbors (KNN), Naïve Bayes (NB), and eXtreme Gradient Boosting (XGB) are implemented. These models were evaluated using classification metrics; accuracy, precision, recall, and f1-score. It is concluded that the Naïve Bayes and Logistic Regression are not the best methods to perform classification on these datasets. On the other hand, DT, RF, KNN, and XGB provided an accuracy of 99% for binary labels and 98% for multiclass labels for the Ton-IoT dataset. Using the IoT-23 dataset, these models provided accuracy above 90%. It is found that LR and NB are not the best choices for classification using either dataset. In addition, not all the features in these datasets are essential; hence some can be dropped to reduce the complexity of the model and improve the computational capacity. It is further concluded that heterogeneity in the dataset does not necessarily affect the performance of classification algorithms.

Keywords: IoT malware, Heterogeneity, IDS, Classification, BotNet Attacks

Aqeel Ahmed

Date: 15-6-2022

Table of Contents

List of Figures

List of Tables

1 Introduction

This chapter presents the background of the internet of things and security problem related to IoT devices. In addition, the main aim and the objectives of the study are also discussed in this chapter.

1.1 Background

Internet of Things (IoT) is everywhere; it is one of the building blocks of many technologies, including smart cities. Currently, more than 20 billion of these devices are connected to the Internet, and the number is expected to grow in the future [1]. These devices are vastly deployed for smart city applications, such as smart healthcare, smart homes, smart mobility, smart agriculture, etc. Cyber-attacks are also proliferating with the expansion of these IoT devices in smart environments. On the other hand, securing these devices is one of the biggest challenges in the current digital age. IoT provides huge attacks surface. It is because of weak encryption methodologies deployed by the IoT vendors and vulnerable default password settings. Such vulnerabilities put these devices at stake and allow hackers to exploit these loopholes and gain access to critical data [2].

As we know, pervasive IoT applications enable us to perceive, analyze, control, and optimize traditional physical systems. Currently, IoT devices are deployed at a rapid pace, and the total number of devices has already crossed the number of humans on the planet and is expected to grow manifold by the end of this decade. According to Business Insider's report [3], the IoT market has a huge potential to grow by over 2.4 trillion dollars by 2027. However, severe resource constraints and insufficient security design are two major causes of many security problems in IoT applications. Easy and guessable default passwords by the manufacturers can be easily exploited by the hackers and gain access to these devices. According to the semantics report [4], most of these devices were sent to the customers with top 10 easily guessable passwords, such as admin, root, 12345, user etc. Over time the attacks on IoT devices have increased globally. According to Semantics Internet Security Threat Report (ISTR) [5], monthly average large-scale attacks on IoT devices

have seen a surge in the past few years. The most famous attack that gained security experts' attention was the IoT BotNet attack called Mirai in 2016 [6]. Mirai took over many high-profile web services like Netflix, Twitter, Reddit, and NY times and made them inaccessible. It is vital to meet the IoT security requirements i.e, confidentiality, integrity, availability, access control, and authentication. It has been realized that the traditional intrusion detection systems are not very effective against these BotNet attacks because of the heterogenous nature of IoT devices. The traditional mechanisms are time-consuming and slow in the detection of malware. This challenge requires a complete shift toward advanced methods and effective approaches to secure the IoT ecosystems from malicious actors.

According to a report by Council to Secure the Digital Economy (CSDE) [7], malicious activities against the IoT devices increased significantly during the pandemic period. It is because corporations and companies moved from physical to remote work environments. Specifically, a massive surge has been witnessed in DDoS attacks on IoT devices during the first quarter of 2020. It is found that the DDoS attacks have increased to around 542% during that period [7]. The reason for such a big rise in malicious activities against the IoT devices is the paradigm shift in the form of digitization of many sectors. The pandemic provided an open surface for hackers to exploit the vulnerable devices and gain unauthorized access.

1.2 Motivation

The IoT systems can be connected in different network structures. Mainly, the available IoT network datasets are collected using limited local area network (LAN) systems where one IoT device communicates with the other in a controlled environment. There is not much consideration given to the network heterogeneity in IoT systems. The heterogeneity in IoT network data means the number of network layers used in the testbed, number of attacks, number of features collected, and number of real-time IoT devices used for traffic generation. Therefore, the main motivation of the study is to present the analysis of IoT network malware classification using real-time heterogenous data.

In addition, Artificial Intelligence (AI) and Machine Learning (ML) have been perceived as the main choice of researchers to detect IoT network malware. ML has proven to be a very effective solution in different aspects of our daily life problems.

ML algorithms, given their mathematical complexity, can learn difficult patterns from the data and help in decision making. As far as the IoT networked systems are concerned, ML algorithms can be very effective in monitoring the behaviour in the network traffic data in real-time and detecting any abnormality. Therefore, using ML would further enhance the network security system against the persistent malware threats and detect them early. Hence, realizing the potential of ML, utilizing accurate and efficient ML-based algorithms for IoT attack detection is inevitable.

1.3 Aim of the Study

The aim of the study is the early classification of IoT network malware attacks on IoT devices using ML algorithms. The research will focus on the use of publicly available real-time heterogenous IoT network datasets since the actual efficacy of an algorithm can be assessed only on real-time data.

1.4 Research Objectives

- To select the heterogenous IoT network dataset collected using real-time IoT devices
- To select the dataset that contains real-time IoT BotNet attacks
- To select the important features from the dataset using feature selection method
- To highlight the important common features in datasets
- To analyse the performance of classification algorithms on the selected dataset
- To analyse the impact of removing features on accuracy and training time.

2 IoT Definition, Applications and Security Aspect

This chapter presents a brief introduction to the Internet of things. The chapter covers the definition of the IoT by IEEE and ITU, applications, and security vulnerabilities. In addition, IoT BotNet characteristics and various types of BotNet attacks are also discussed.

2.1 Defining the Internet of Things (IoT)

The Internet of things (IoT) has been around since its inception in the 1960s. However, the most popularly cited event regarding the IoT devices is Coca-Cola vending machine connected to the Internet by programmers at Carnegie Mellon University in 1982 [8]. The term "Internet of things" was first coined by *Kevin Ashton* during his presentation in 1999 at Proctor and Gamble, talking about the role of RFID in the company's supply chain [9]. After 2004, the phrase "Internet of things" was commonly adopted in the literature. According to International Telecommunication Union (ITU) recommendation (ITU-T Y.2060), IoT is defined as; "*A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving, interoperable information and communication technologies*" [10].

In addition, IEEE also defined the term IoT as; "*An IoT is a network that connects uniquely identifiable "things" to the Internet. The "things" have sensing/actuation and potential programmability capabilities. Through the exploitation of the unique identification and sensing, information about the "thing" can be collected, and the state of the "thing" can be changed from anywhere, anytime, by anything"* [11].

Currently, IoT has been reported as the fastest growing technology for it is being integrated to every field of our life**.** It is becoming a billion dollars market as the number of these IoT devices being used is increasing rapidly.

2.2 IoT Applications

Internet of things has now come across the hype; it has gained momentum and it has broad application landscape. Due to the growing market of IoT, different new avenues for the

deployment of IoT systems are explored. One can spot IoT devices almost everywhere now a days in our lives. These devices are being used as smart healthcare wearables, home automation, etc. According to IoT analytics [12], the Industrial IoT (IIoT) has become the top sector for the employment of smart and intelligent manufacturing systems leaving behind the smart cities.It is because of the birth of a new paradigm called Industry 4.0. As described in [13], the top IoT applications shown in Figure 1, are briefly described below.

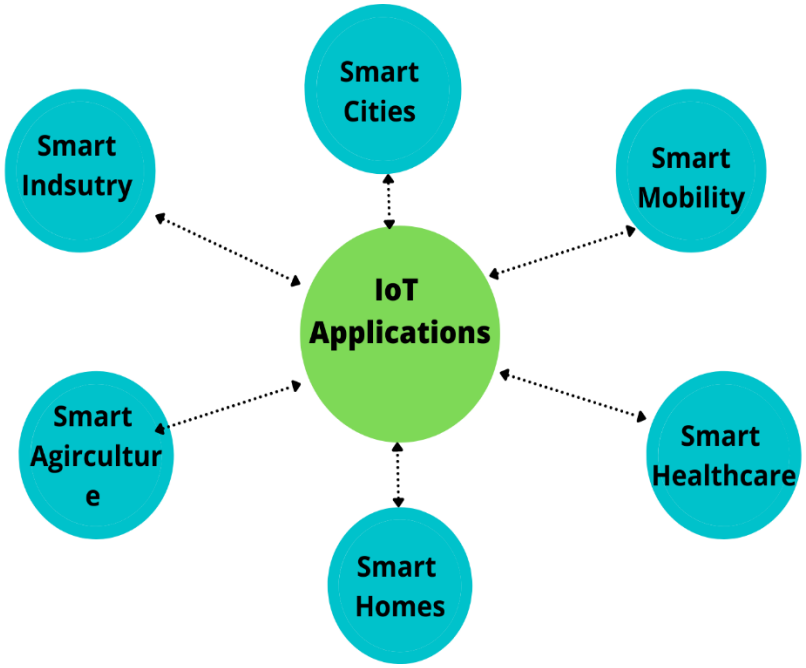


Figure 1: Applications of IoT

- *Industry and Manufacturing (Industry 4.0)*

  Industrial automation is one of the rapidly growing applications of IoT systems. The integration of smart systems makes the manufacturing and production system much faster and more efficient, resulting in the smoothing of the overall process.

- *Smart Healthcare*

  e-Health or smart healthcare is also one of the growing applications of IoT. Various smart gadgets are now easily available; these gadgets are able to record human health conditions and report to the remote server or to a mobile application. These gadgets include wristbands, apple smartwatches, smart glasses, etc.

- *Smart Homes*

  Smart homes are one of the popular use cases of IoT. The home electronic appliances can be controlled through mobile applications or remote web systems. The sensors such as temperature, humidity, and smoke transfer continuous data to the user end, which can be

analysed and controlled if necessary. This results in the home's energy efficiency since one can see the appliance's performance and use them accordingly. In addition, smart locks and smart surveillance systems are also becoming common in homes.

- *Smart Mobility*

  One of the major advancements in the mobility sector is autonomous cars enabled by IoT technology. Big technology companies such as Google and Tesla have already demonstrated self-driving cars using IoT.

- *Smart Agriculture*

  IoT technology has also stepped into the agriculture sector as well. The farmers can now use intelligent IoT systems in their fields to report to the user about the crop and weather conditions. In addition, smart irrigation systems are also developed using IoT technology which has proven to be valuable in reducing the water consumption of the crops.

- *Smart Cities*

  One of the most prominent candidates for the deployment of IoT devices is smart cities. The smart city requires the IoT system to be deployed in various sectors and parts of the city ranging from traffic control, solid waste management, smart water distribution, smart grids, and smart resource management.

The applications of IoT technology are swiftly increasing with the advancement in data processing technology. It is because IoT devices generate a huge amount of data. The storage, processing, and analysis of such a massive amount of data require super-fast, efficient, and competent systems. Thanks to the Big data technologies, which are becoming the solution for this problem.

In the future, one can expect further drift in the development of IoT technology, and its uses will also substantially increase. The integration of other cutting-edge technologies such as Blockchain is also considered a future trend in this domain. The combination of the IoT and Blockchain would completely change the current way of looking at smart systems in the future.

2.3 IoT Security Challenges

Although IoT technology is experiencing exponential growth in the market, everyone is so curious to install IoT systems for the purpose of process automation. However, the biggest challenge for the IoT devices, which was not paid much attention to or completely ignored by IoT manufacturers, is the security of these devices. Securing the users' privacy should be the top priority of IoT companies. Nonetheless, cyber-attacks can be so lethal for the IoT systems deployed in critical infrastructures. Given this technology's distributed nature and pervasive use, it is crucial to understand the security problem and take countermeasures. There are several reasons for this growing challenge of IoT security, mainly the lack of computational capacity, weak password encryption methods, and lack of regular firmware updates. Many pieces of research are being conducted to highlight various security challenges at each layer of the IoT architecture [14]–[16].

2.4 Resource Constraints in IoT Devices

The main characteristics of the IoT systems are the large-scale deployment, distributed nature in terms of network topologies, and heterogenous nature of devices. These devices will be connected to different platforms using different architecture and posing unique integration and security challenges. But resource constraint is the major backdrop of IoT devices. As shown in Figure 2, the IoT devices are not really powerful in terms of processing the data.

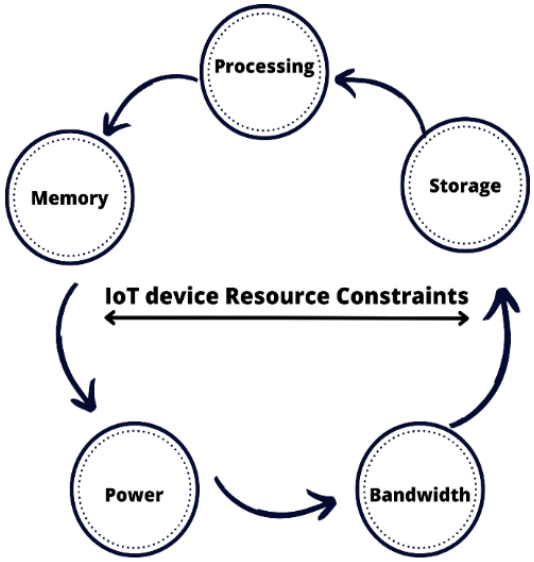


Figure 2: IoT Device Major Resource Constraints

They are not like conventional devices that can take up to 100s of gigabytes and still work. These devices usually run on the microcontrollers with the capability of only 100s of megabytes, not the gigabytes. Additionally, IoT devices cannot store the huge amount of data continuously coming from the sensors. Another major limitation to the IoT devices is the physical environment; not all the devices can sustain the harsh environment without specific protection [17].

2.5 IoT Threats and Vulnerabilities

As defined by the International Engineering Task Force (IETF), a cyber threat is actually "*a potential for violation of security, which exists when there is a circumstance, capability, action, or event that could breach security and cause harm. That is, a threat is a possible danger that might exploit a vulnerability*" [18]. At the same time, a vulnerability is the weakness of a system that can be exploited intentionally by the malicious actors for personal gains. IoT threat is always there, and the consequences can be fatal if IoT-based systems such as autonomous cars, sensor-guided weapons, and large-scale industrial systems are compromised.

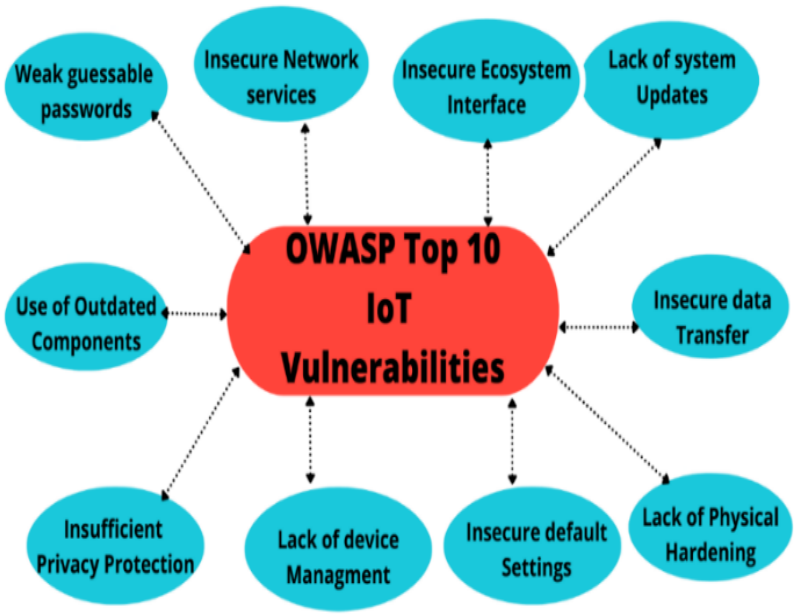


Figure 3:OWASP Top 10 IoT Vulnerabilities

An extensive study of the various IoT vulnerabilities and their exploitation possibilities at different layers with attack types is discussed by the authors in [19].

In addition, various IoT security organizations such as *OWASP* [20] and *IoTSF* [21] are extensively conducting research on IoT vulnerabilities. According to OWASP, the top ten vulnerabilities in IoT devices are shown in Figure 3.

We will briefly describe the OWASP top 10 IoT security vulnerabilities:

1. *Weak and guessable default password*

   According to OWASP, easy default password by manufacturers is among the top vulnerabilities of IoT devices. These passwords can easily be guessed even by naïve hackers. Almost all the manufacturers send their devices encrypted with easy passwords such as admin, 123, or the device name.

2. *Insecure Network Services*

   The IoT devices are not capable of installing traditional intrusion detection systems on them. Hence unauthorized access is not very difficult on these devices. This vulnerability is easily exploited by the hackers who execute DOS attacks that populate the devices' bandwidth capacity, resulting in the failure of the services.

3. *Insecure Ecosystem Interfaces*

   Usually, the web, API, or cloud interface is not encrypted, solid, or secured. Therefore, making the IoT device vulnerable to cyber-attacks. The device can completely lack authorization if the interface is compromised.

4. *Lack of Security Update Mechanisms*

   One of the major backdrops for IoT devices is not having regular security updates. Unfortunately, these do not offer this facility by default providing any guarantee of the security for the end-user. Therefore, OWASP placed it at the 4^{th} position.

5. *Use of Outdated Components*

   It can be understood from this vulnerability that IoT isn't only vulnerable at the interface level. The insecure or outdated software system is also equally dangerous and open to exploitation and can completely block the production system.

6. *Insecure Privacy Protection*

   Device privacy and the user's data privacy is also the main element in making the IoT devices secure. The IoT data traffic can be analyzed or saved without permission.

7. *Insecure Data Transfer Storage*

   It is obvious that data transfer and storage of the data must be secure and safe. Its access must not be allowed to unauthorized actors without permission. Therefore, encryption of data is essential in this case. Plain data transfer and storing it without proper security measures can be harmful.

8. *Lack of Device Management*

   This vulnerability has been placed on number eight by OWASP, but it is also crucial to keep the device management intact. Proper management of the devices with monitoring control should be the primary concern of the users.

9. *Insecure Default Setting*

   As already mentioned, any default setting such as the default username, password, or IP address of IoT devices can become a vulnerability and be exploited by the hackers. Therefore, the default setting must be updated regularly.

10. *Lack of Physical hardening*

    Last but not the least on the list is the lack of hardening of an IoT device. This means the debugging of the ports, secure boots, and removing cards can be a vulnerability.

2.6 IoT BotNet Attack

IoT BotNet attacks are currently considered as deadly and widely spreading attacks on IoT devices. These attacks are initiated by an infected device called a "Bot" or a zombie. A bot can be any IP-based device connected to the Internet through a wired or wireless network. It can be an IP camera, a router, a printer, or IP TV [22] **.** Mirai is one of the first large-scale attacks discovered in 2016, which took down thousands of devices and popular web application services [6]. As shown in Figure 4, in an IoT botnet attack (Mirai BotNet), a malicious actor establishes a command and control (CnC) server to manage the attack. In addition to the CnC server, a ScanListen server and a LoadServer are also established to conduct the attack. Basically, the malicious actor or botmaster uses an infected device to telnet scan the immediate connections to the first bot. When the bot identifies any vulnerabilities in the target, it tries to establish a connection with the target and brute forces to get unauthorized access. If the link is successfully established, the bot informs the ScanListen server. The ScanListen server identifies if the target device is already in the database.

In case the target is new, the LoadServer is initiated to start downloading the malicious codes on the target device to get it infected. This target becomes a bot or zombie and shares the information with CnC and botmaster.

Now that newly infected devices act as zombies, it starts scanning the ports and strives to establish a connection with immediate hosts on the network. This process continues, and millions of devices become zombies and create a colossal BotNet. This is how the famous Mirai BotNet attack worked. However, there are many variants of the Mirai attack shared on hack forums. Hence, every day a new BotNet attack is created. In addition, the worst that could happen is the open availability of these attacks. One can buy the bots and execute the BotNet attacks on the other systems.

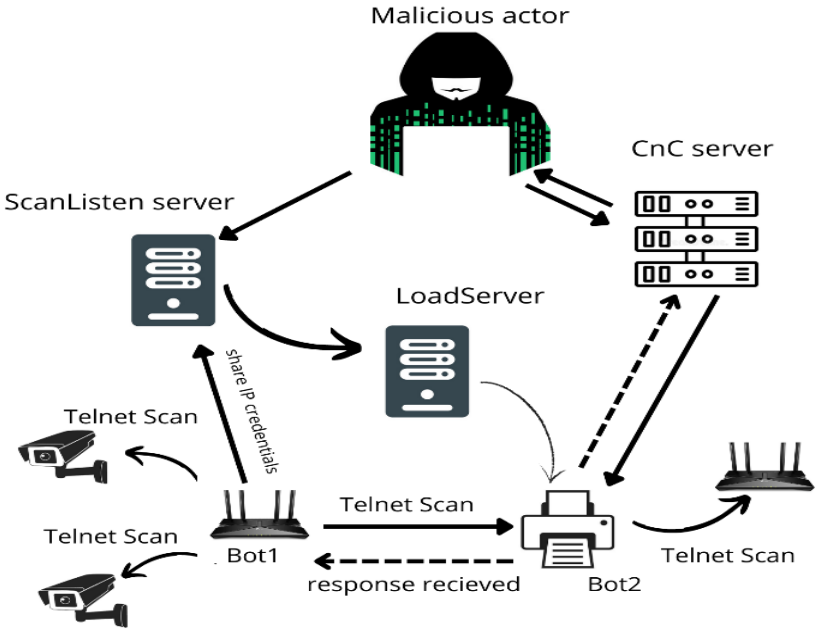


Figure 4: Working Principle of IoT BotNet (Mirai BotNet)

2.7 Popular IoT BotNet Attacks

IoT BotNet attacks are becoming a serious challenge for cybersecurity companies. Many variants of the existing attacks are uploaded regularly on the internet, which can be used by anyone with some knowledge of hacking. We discuss some of the attacks which have been very popular due to well-known attack vectors. These attacks include Mirai, Okiru, Hajime, Hide and Seek, Torri, Bashlite, Hakai etc. These attacks aim to disturb the services by sending DDoS packets and gaining unauthorized access to IoT devices.

2.7.1 Mirai BotNet

The Mirai attack has been discussed previously. It has been one of the infamous and lethal attacks on IoT devices in the recent past. It was used to initiate millions of packets from approximately 380,000 devices placed in different locations.

The DDoS attack by the Mirai botnet was one of the largest attacks ever seen in the history of the Internet. It was very simple yet very dangerous. The Mirai was coded in C programming language, and the code was released on a website called hackforums.com by its author name Anna-Senpai [23]. The release of the code further worsened the situation as even the newbies to the programming could change the code and create a new variant of Mirai.

2.7.2 Okiru BotNet

Okiru is a variant of Mirai discovered in 2018 by an independent researcher [24]. The main target of Okiru botnet was the Linux-based operating system. The IoT devices comprised of (Argnoaut RISC core) ARC processor was found to be vulnerable to this kind of attack. The ARC central processing unit is in common use and widely popular for cameras, mobiles, and smart electricity meters. The Okiru is gaining popularity and its versions are being created because of the code available on the internet.

2.7.3 Bashlite

The Bashlite is another widespread IoT BotNet attack that has been discussed with other names such as Gagfyt and Lizkebab. The primary purpose of this attack is to exploit the Bash vulnerability in shellshock. A hacker can access an IoT device remotely using a Bashlite attack. It is used in exporting shell functions via environment variables. The leading cause of this kind of vulnerability in IoT devices is poor patching. It self-propagates after infecting the devices and gathers information about the immediate hosts using telnet [25].

2.7.4 Torri

Torri is different than Miria and Okiru, as claimed by the Avast researchers [26]. It is a persistent attack that does not initiate the DDoS attacks but applies more advanced methods such as encrypted communication and data exfiltration. The Torri is seen to have a systematic attack strategy. It performs reconnaissance, target initiation, and payload downloads in sequence. It also uses different methods to download the payload on the target and communicate with CnC server through XOR-based cipher. [26].

2.7.5 Hajime

Hajime is another IoT botnet that is in continuous evolution. The word Hajime means 'beginning' in Japanese. The Hajime was first discussed in a public report published by Rapidity Networks [27]. The strange fact about the Hajime is its unknown purpose. Until today, no one has been able to precisely describe the core purpose of this attack. The Hajime is indeed a variant of Mirai, but it builds a peer-to-peer network.

2.7.6 Dark.IoT

A recent variant of Mirai called Dark.IoT BotNet was reported by Palo Alto in 2021 [28]. According to the report published, The new variant could exploit the vulnerabilities in the IoT devices. The Dark.IoT BotNet as named by experts, uses file names Dark architecture. The attackers were able to delete the important files on the target devices and install the binaries. The Dark.IoT has been successful in targeting many devices since last year. Therefore, it is a concern for the researchers.

2.7.7 Hakai

Hakai is based on Mirai and Garget malware. The Hakai is a Japanese word meaning 'destruction'. It was mainly detected in routers because it leverages the vulnerability of routers that allow remote code execution [29]. It seems to have been used to carry as a DDoS attacker, shell command line injector, and telnet scanner.

2.7.8 Hide and Seek

Hide and Seek (HNS) is a variant of Mirai identified by the Bitdefender[30]. The scanner functionality of this attack has been borrowed from the Mirai. The scanner part tries to reach random IPV4 addresses through predefined ports such as HTTP, HTTPS, Orient DB, CouchDB, and Telnet. Once these ports are detected as open, the exploitations start using a simple self-synchronizing cipher. Communication is carried out through peer-to-peer connections like Hajime. These attacks are quite difficult to take down and are seen to get updated frequently. The sample code and updates are available here [31].

3 Related Work

IoT BotNet attack detection is currently the most researched area of IoT security. Various ML-based methods are proposed in the literature to detect these attacks [32]–[34]. Researchers are using classification methods such as Support Vector Machine (SVM) and Decision Trees (DT) [35], Random Forest (RF) [36], and K-Nearest Neighbours (KNN) [37] for classifying the malicious and benign network traffic.

However, these papers lack effective and necessary practices, such as data pre-processing or feature engineering steps to detect the intrusion accurately. In addition, it is essential to use real-time IoT attack data to evaluate the performance of the proposed method in real environment.

The authors in [38] use DT, multiclass decision forests, and multiclass neural networks trained and tested on an imbalanced IoT23 dataset. The performance of these models was evaluated on a smaller dataset created from IoT23 log files, and later the larger version of the same dataset was used by increasing the number of data samples, the accuracy, probability of attack detection, and probability of false alarm detection increase up to 99.9%. The models were trained using a 10-fold cross-validation method.

A KNN, multilayer perceptron (MLP), and Gaussian Naïve Bayes (GNB models were used to classify the malicious and benign IoT traffic using the BoT-IoT dataset [39]. The key takeaway from this research is the combination of feature engineering and Synthetic Minority Oversampling SMOTE technique with ML models. The models were trained both on the class-imbalanced and class-balanced datasets. The features with F1-score greater than the mean value were selected, while those with low scores were rejected. Only eight features provided scores greater than the mean. Among all the algorithms used above, KNN provided the best Area Under Curve (AUC) values and accuracy on the imbalanced dataset.

In [40], the authors presented an IoT attack and anomaly detection using various ML algorithms, including LR, SVM, DT, ANN, and KNN. The proposed models are trained on newly available network anomaly datasets i.e, UNSW-NB15 and CICIDS2017. However, these datasets are not purely from real IoT devices but provide a good overall scenario for understanding BotNet attacks on IoT devices in a smart city environment.

In addition, the results show that for both datasets, the proposed ensemble methods outperformed the other algorithms with performance in terms of True Positive Rate

(TPR),False Positive Rate (FPR), and F1-score. The accuracy of all the methods was better (0.99) on the CICIDS2017 data than the UNSW-NB15.

In addition, many novel methods for attacks are being introduced [41]. In [42], a comparison of ML and deep learning (DL)-based algorithms using the IoT-23 dataset is presented. The CNN used with the rectified linear unit (ReLu) as the activation function is evaluated using precision, recall, f1-scores, and support scores. The authors also show the time-cost comparison of all the models used. It can be observed from the results that DT is the best performing model in terms of both the accuracy (0.73%) and time cost (6 seconds) followed by the CNN model with an accuracy of 0.69% and time cost (242 seconds). Naïve Bayes performed worse than SVM in terms of accuracy but was faster.

In [43], A sequential architecture approach is proposed, which involves different phases such as data collection, data categorization, model training, feature selection, and attack detection. The -sub-engine for the attack detection approach is used in the proposed architecture. Each attack class was assigned a sub-engine, and data samples were assigned to the model is assigned for classification. The architecture is based on ML algorithms, including ANN, J48, and Naïve Bayes. The authors used N-BaIoT datasets containing samples from IoT BotNet attacks such as Mirai and Bashlite.

Interestingly, the performance of the proposed models varies on the type of class used in multiclass classification. The attack detection accuracy for all the models is almost similar, which is 99%. It can be observed that the performance of J48 and ANN does not change by varying the number of features. On the other hand, the performance of the NB degrades by increasing the number of features. In addition to the ML methods, the authors are also proposing the use of the deep learning-based method for intrusion detection.

An Ensemble learning technique based on AdaBoost, RUSBoosted, and bagged method is presented in [44]. The authors proposed a three-phase model architecture to detect the intrusion using N-BaIoT dataset. I addition, the data is pre-processed and scaled using z-score standardisation method before the training process. The feature selection is performed using a correlation score. Moreover, the author used 5-fold cross-validation to avoid overfitting on the training data. The ensemble methods were evaluated using binary, three-class, and multiclass classification.

The proposed ELBA-IoT method provides an accuracy of 99.6% on the Ba-IoT dataset. The experimentation was performed using Matlab-2021 software installed on windows operating system.

A feed-forward neural network (FFNN) has been investigated using flow and flag-based features from multiple datasets [45]. The proposed neural network comprises of one input layer, four hidden layers, and one output layer. The model was tested on binary and multiclass classification problems. In addition, a dropout layer was also added to reduce the probability of overfitting. The authors combined various datasets to increase the number of samples and classes. The authors achieved an average accuracy of 99.10% on the testing set using a learning rate of 0.001, and 100 epochs.

An XGBoost-based method called PSO-XGBoost is proposed for network intrusion detection systems [46]. The proposed technique is a parametric optimization of XG-Boost algorithms that outperform existing Adaboost and Random Forest models. The particle swarm optimization (PSO) technique is derived from the bird's swarm behavior, where the particles are considered as the simulation of birds. The models were experimented with using the benchmark NSL-KDD dataset. The NSL-KDD is an old dataset with four target variables. The dataset was pre-processed before training. The results of the model were evaluated for each class. The average precision, recall, and F1-score achieved with the proposed model were 0.81, 0.75, and 0.71, respectively.

A deep neural network-based attack detection system is proposed in [47]. The authors used a publicly available Bot-IoT dataset which consists of actual IoT BotNet attacks. The dataset includes different types of attacks such as DoS, DDoS, and keylogging. A subset of the dataset was extracted and used for the experiments. Before training the neural network, the data was pre-processed and normalized using min-max scaling. The proposed neural consist of an input layer, two hidden layers, and an output layer. A linear rectified unit (ReLu) activation function was used to introduce the non-linearity in the model. The proposed neural network achieved an accuracy of 95%.

In [48], a comparative analysis of LR and ANN is presented using the N-BaIot dataset. N-BaIoT is also a real-time IoT dataset collected from Doorbell, Thermostat, Baby monitors, and security cameras. It is shown that the proposed ANN provides excellent results in terms of accuracy, precision, and recall values for each device. The proposed neural network achieved an accuracy of 0.92, 0.94, 0.91, and 0.92 for the traffic from Doorbell, Thermostat, Baby monitor, and security camera, respectively.

In [49], several tree-based ensemble learning methods are implemented using the Bot-IoT dataset. Bot-IoT is a famous publicly available and commonly used IoT network dataset. The authors chose RF, Light Gradient Boosting Method (LGBM), Extra Tree, and

XGBoost method for the task of attack classification. The authors selected the ten best features using the Pearson correlation method and entropy. One can observe from the results that LGBM outperformed all other methods in terms of accuracy, precision, and recall scores. LGBM provided an accuracy, precision, and recall score of 99.9%. It is also one of the fastest methods in terms of training and execution time.

In [50], the authors present the analysis of several ML classification algorithms experimented on the Ton-IoT dataset. The feature selection was performed using the chi-square test. Moreover, simple minority oversampling (SMOTE) was used to tackle the imbalance samples in the Ton-IoT dataset. The data was split into a 70:30 train-test ratio, and a 5-fold cross-validation method was used to avoid any overfitting. It can be observed from the results that for binary classification, XGB outperformed other methods with 99.1% accuracy, followed by KNN with 98.8%., decision tree (DT) with 98.0%, and RF (97.9%). In terms of multiclass classification, again, XGB provided an accuracy of 97.9%, while KNN (97.6%), RF (91.1%), and DT(91.3%). All these results are noted after using chi-square and SMOTE methods.

Although, the use of ML for IoT BotNet attack detection is gaining massive attention by the research community. There are still gaps in standardising the important features and attack types. It is essential to underline the important features in the dataset, which contribute more toward the accurate classification of malicious and benign data. It is also critical to highlight the importance of heterogeneity in the IoT network datasets. It is found that not much attention has been paid to that aspect of IoT network malware classification. Hence this study is focused on using real-time heterogenous data for the experimentation purpose.

4 Dataset

This chapter describes the datasets used in this study. The main objective of the research is to investigate the efficacy of ML algorithms on real-time IoT attack datasets. Hence, we chose to work on two real-time IoT network datasets, which are described in detail.

4.1 IoT Network Malware Datasets

The major challenge in IoT network attack detection is the availability of good quality data as the ML algorithms require data to perform prediction. The availability of quality data is the main issue in IoT security scenarios. The main aim of the study is to select the recent IoT network malware dataset. In addition, the dataset must be collected from real-time IoT devices. The reason for choosing real-time IoT datasets is to train the ML models with real IoT network data. The heterogeneity was also set as the main condition for the dataset to be used in this study.

Various network intrusion detection datasets are publicly available; KDD [51], is one of the first among NSL. However, with further advancements in the field of cyber-attack data collection, new datasets, such as ISCX-2012[52], UNSW-NB15[53], and CICIDS2017[54] were published in the last decade. However, all these datasets are not IoT specific and do not cover large-scale attacks of today's time. Moving on, some of the IoT-specific datasets currently published include N-BaIoT [55], Bot-IoT [53], and IoT-23 [56].

These datasets are either collected using real-time IoT devices or emulated ones. However, one of the major concerns is the heterogenous nature of the IoT datasets. Heterogeneity is considered in terms of network structure, the number of malware attacks, and a number of devices used during data collection. It is because, the new attacks are being created and therefore, training ML algorithms using recently available data is inevitable. The mentioned datasets do not reflect the complete heterogenous nature, which is important for addressing the standardization problems.

4.1.1 Ton-IoT Dataset

Ton-IoT is a heterogenous IoT network dataset publicly available. The dataset was collected from realistic IoT sources in a controlled lab environment at Cyber Range and IoT labs, University of South Wales (UNSW) Canberra, Australia. The test-bed includes the scenarios of IoT and Industrial IoT (IIoT) with several sensors to collect telemetry data. In addition, a diverse set of operating systems including; Windows 7, Windows 10, Linux, Kali Linux, and virtual machines, including Ubuntu 14 and 18 TLS were used for the deployment. The critical aspect of the Ton-IoT testbed is its criteria of fulfilling the network heterogeneity. The network created in this test-bed scenario uses all the major layers of communication for an IoT system i.e, the Edge layer, Fog layer, and cloud layer, to ensure the complete sense of network heterogeneity in the data collection [57].

The dataset consists of nine different types of attacks on web applications, network gateways, and operating systems. All these different types of attacks are described below. The network flow dataset is carefully labeled and provided in comma-delimited CSV format for compatibility with various platforms. Figure 5 shows the sample distribution in the dataset. The network flows dataset provided in CSV format is recommended by the authors for experimentation and used in this study. The network dataset consists of 461,043 rows. The normal traffic denoted as '0' comprised of 300000 samples (65.07%), whereas the malicious data denoted as '1' is equal to 161043 samples (34.93%) of the total data. The dataset is further labeled into categories of the attacks carried out and their distribution, as shown in Figure 6.Figure 6

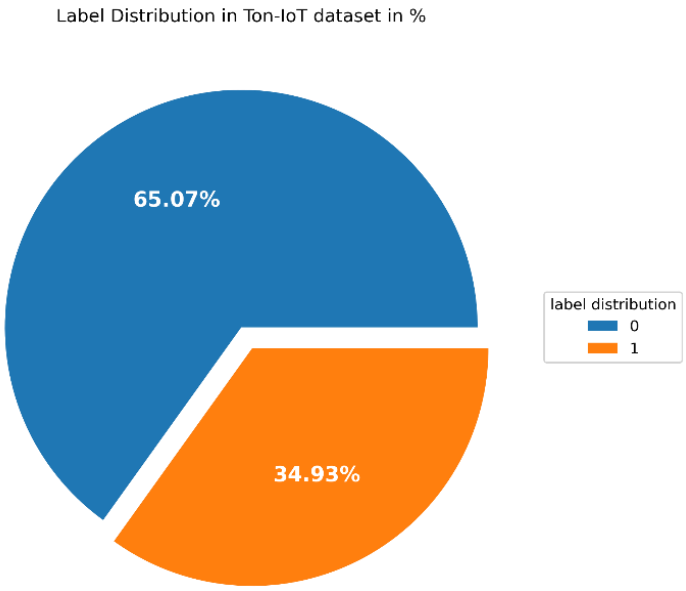


Figure 5: Binary Label Distribution in Ton-IoT Dataset

4.1.2 Attack Categories in Ton-IoT Dataset

The dataset consists of nine different attack categories, as described below.

1. *Scanning Attack:* this is the first step for exploiting any system further. Scanning is used to find the vulnerabilities in the target systems, such as open ports. The scanning attack gathers all this information and carries on with further actions on that victim.
2. *Denial of Service (DoS):* DoS is a famous attack where the target is flooded with packets to interrupt that host's services and ultimately take it down. DoS interrupts the services and makes them unavailable to the customers. It can be very harmful to the businesses until its recovery
3. *Distributed Denial of Service (DDoS):* This attack is even more dangerous than DoS. The target devices are bombarded with a massive number of packets from various attackers such as Bots. IoT devices are vulnerable and weak against these types of attacks because of bandwidth constraints.
4. *Backdoor:* This is a dangerous attack, especially on web applications, as malware is installed without letting know to the host, and unauthenticated access is granted. With this attack, the hacker can access databases on that application, steal the data, or hijack the server. The adversaries can use this attack to gain access to IoT systems and initiate a DDoS attack on the whole network.
5. *Injection attack:* the injection attack has many categories itself. The hacker can exploit the code and change its commands to practice malicious activity. The host cannot identify the code injections and compile them as part of the code hence losing control to the adversary. The IoT systems are quite vulnerable to these types of attacks.
6. *Password Cracking:* The hackers try to break your password using brute force methods in this attack. Due to the default and weak password encryption methods, IoT devices are the easiest target for hackers to gain access. Once the attacker gains access to the devices, he can change the setting and control the devices accordingly.
7. *Man in the middle (MITM):* this is a famous network attack on the target devices where an attacker places himself in the middle of the two devices without letting them know and captures the packets from the source to the destination. This attack is carried to gain access to critical information in the network data, such as login details, credit card information, etc.
8. *Ransomware:* Ransomware is one of the most lethal types of attack on computer systems that completely blocks legitimate user access to the system. The attackers enter the target systems and block the owner or the system's user until he is paid. Every year millions of

dollars are earned by the attackers using this type of attack. This can be very harmful to IoT devices, especially for IoT systems.

9. *Cross-Site Scripting (XSS):* XSS is also a type of injection attack mostly carried out on vulnerable web applications. An IoT server can be infected with malicious code and masquerade as a legitimate user. This is usually done through browser-side scripting.

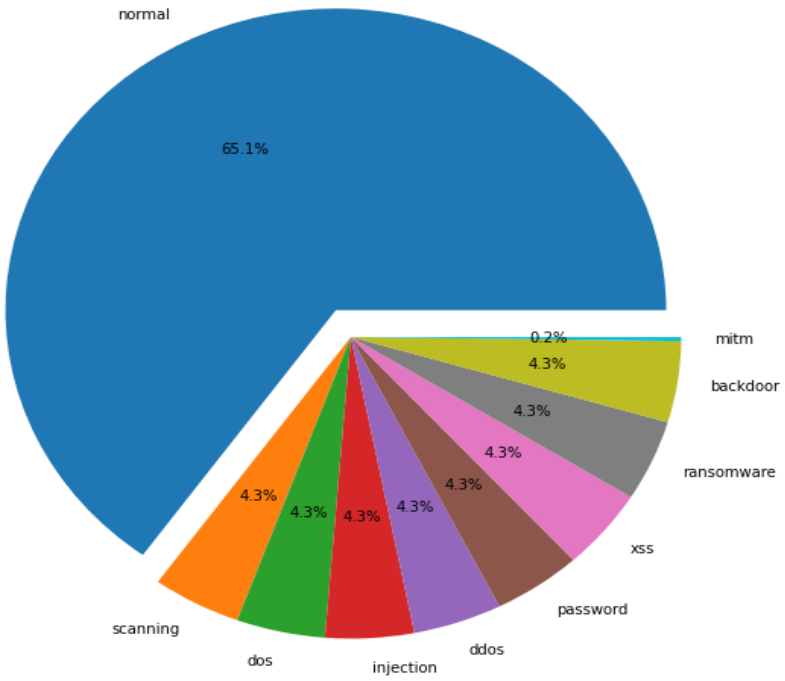


Figure 6: Multiclass Label Distribution in Ton-IoT Dataset

Features in the Ton-IoT dataset are divided into six categories depending upon their nature. The authors of the dataset provided a detailed analysis on the quality Ton-IoT in [58]. The categories of the features include connection activity, DNS activity, SSL activity, statistical activity, HTTP activity, and violation activity. All the features and their description is provided in Table 1. In addition, it is found that most of the traffic in the Ton-IoT dataset consists of TCP packets. It shows the communication protocol used in Ton-IoT is mostly TCP and UDP and some samples of ICMP as shown in Figure 7.

Table 1: Ton-IoT Feature Description

ID	Feature	Description
1	ts	Time of connection flows
2	src_ip	Source IP address
3	src_port	Source device port address
4	dst_ip	Destination device port address
5	dst_port	Designation device port address
6	proto	Transmission Control Protocol
7	service	Dynamically detected protocols (DNS, HTTP, SSL)
8	duration	Time difference between first packet and last packet
9	src_bytes	Payload bytes from sources (TCP sequence numbers)
10	dst_bytes	Destination bytes (TCP sequence numbers)
11	conn_state	Tells about the connection state between src and dest
12	missed_bytes	Number of missing bytes in content gaps
13	src_pkts	Number of original packets from source
14	src_ip_bytes	Number of original IP bytes from source
15	dst_pkts	Number of destination packets from destination
16	dst_ip_bytes	Number of destination IP bytes
17	dns_query	Domain name queries
18	dns_qclass	Specification of DNS query class
19	dns_qtype	Types of DNS query
20	dns_rcode	DNS response code
21	dns_AA	Authoritative answers of DNS server
22	dns_RD	Recursion desired of DNS
23	dns_RA	Recursion available of DNS
24	dns_rejected	DNS query rejected by server
25	ssl_version	SSL version offered by server
26	ssl_cipher	SSL cipher suite which the server chose
27	ssl_resumed	initiate new connection (T denotes resumed)
28	ssl_established	SSL flag indicate establishment of connection
29	ssl_subject	Subject of X.509 cert offered by server
30	ssl_issuer	Trust owner of SLL
31	http_trans_depth	Pipeline depth into HTTP connection
32	http_method	HTTP request method (GET, SET, HEAD)
33	http_uri	URIs used in HTTP request
34	http_version	HTTP version used
35	http_request_body_len	uncompressed size of data transferred from HTTP client
36	http_response_body_len	uncompressed size of data transferred from HTTP server
37	http_status_code	Status code returned by HTTP
38	http_user_agent	Values of user agent in HTTP header
39	http_orig_mime_types	ordered vector of mime types from source
40	http_resp_mime_types	ordered vector of mime types from destination
41	weird_name	Names of anomalies violations
42	weird_addl	Additional information
43	weird_notice	Indicates if violation anomaly was turned into notice

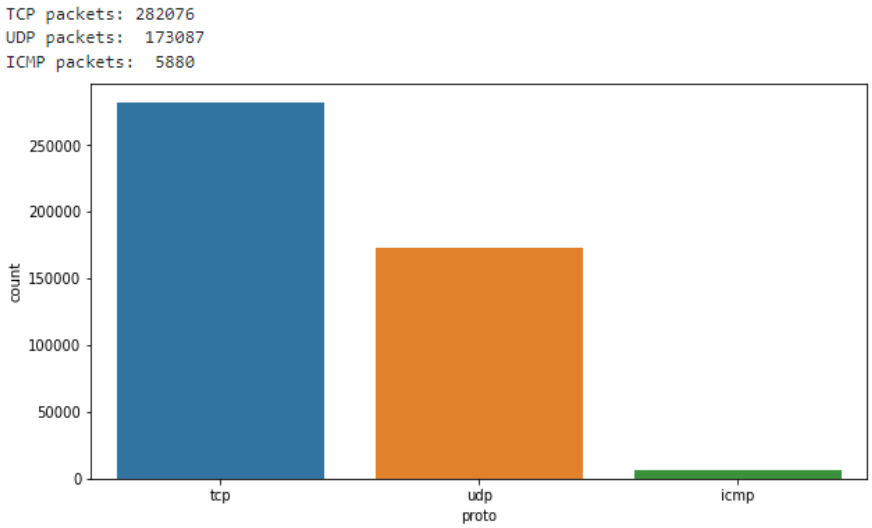


Figure 7: Communication Protocols in Ton-IoT Dataset

4.1.3 IoT-23 Dataset

IoT-23 is a recently released real-time IoT network dataset. It has 20 malware captures executed in IoT devices and three captures for benign IoT device traffic. This IoT network traffic dataset was captured in a controlled environment at Stratosphere Laboratory, AIC group, Czech Technical University, Czech Republic. The packet flows were captured during 2018 and 2019. This dataset is also among the most heterogenous IoT network data because it has various attacks and is collected from real-time IoT devices. The full dataset is enormous as it contains more than 200 million labelled flows. The authors of the dataset created 23 different malware capture scenarios, out of which three captures were completely normal traffic from three IoT devices: Amazon Echo, Philips hue, and Somfy door lock. The packets were captured using Wireshark software and were converted into logs using Zeek software. One of the main advantages of using this dataset is different real-time IoT BotNet attacks such as Mirai, Okiru, Torri, Gagfyt, Kenjiro, Hakai, IRCBot, Muhstik, Hide and Seek, and Hajime was executed on device using Raspberry Pi [56]. These BotNet attacks are already described in the respective section. The features and their description in the IoT-23 dataset are given Table 2**.** The features in italics are used in this study. The attack distribution after pre-processing shown in Figure 9. Most of the scenarios in IoT-23 dataset consist of scanning attacks. It also includes three main communication protocols such as TCP, UDP and ICMP. However, it is observed that most of the packets were the TCP sync floods. Hence one can infer from such a large amount of TCP packets that the BotNets used in IoT-23 dataset targeted TCP ports mostly.

TCP flooding packets are used initiate the DDoS against the target devices. These packets sometimes can be in thousands which results in failure of the devices due to resource constraint.

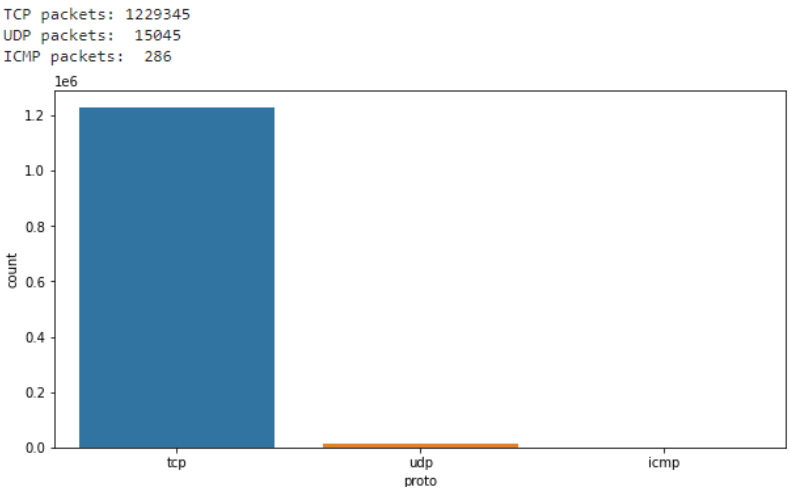


Figure 8: Communication Protocols in IoT-23 Dataset

One can observe from Figure 8, that over one million samples of the data consist of TCP as a communication protocol. This number changes if the number of samples from the original dataset is increased.

Table 2: The Zeek Log Features in IoT-23 Dataset

Feature	Data Type	Description
ts	Float	Time of the packet sent (unix date command)
uid	string	Unique ID of Connection
id.orig h	string	Source IP address
id.orig p	integer	Source Port
id.resp h	addr	Destination IP address
id.resp p	integer	Destination Port
proto	string	Transmission protocol (tcp, udp, icmp)
service	string	Type of service (dns, arp etc)
duration	integer	Time of the last packet received
orig bytes	integer	Source bytes (payload)
resp bytes	integer	Destination bytes (payload)
conn state	string	Connection state
local orig	bool	If conn originated locally T; if remotely F. If Site::local nets empty, always unset.
missed bytes	integer	Number of missing bytes
history	string	History of previous connection established
orig pkts	integer	Number of packets sent by the source
orig ip bytes	integer	Number of IP level bytes sent by source
resp pkts	integer	Number of packets sent by destination (responder)
resp ip bytes	integer	Number of IP level bytes sent by destination
tunnel parents	set	Used only for tunneled connection

4.1.4 **Attack Categories in IoT-23 Dataset**

The attack types in IoT-23 are described below:

1. *Attack*: This label represents any attacks observed from the packet flow. The authors do not specify the type of attack while labelling this flow.
2. *Benign*: All the expected traffic flows were labelled as benign.
3. *C&C*: This label shows that the target was connected to Command and Control (CnC) server.
4. *DDoS*: Distributed denial of service attacks executed by infected devices. This can be detected by the number of packets directed to the same IP address.
5. *FileDownload*: This label indicates the downloading of the malicious code on the infected device.
6. *HeartBeat*: This label indicates that the packets sent for the infected device were used to keep track of the device.
7. *Mirai:* This label indicated the characteristics of the Mirai botnet.
8. *Okiru:* This label indicates the Okiru botnet attack
9. *PortofAHorizontalPortScan:* This suggests that the connections are used to execute horizontal port scanning for further attacks.
10. *Torri:* This label indicates the Torri botnet attack.

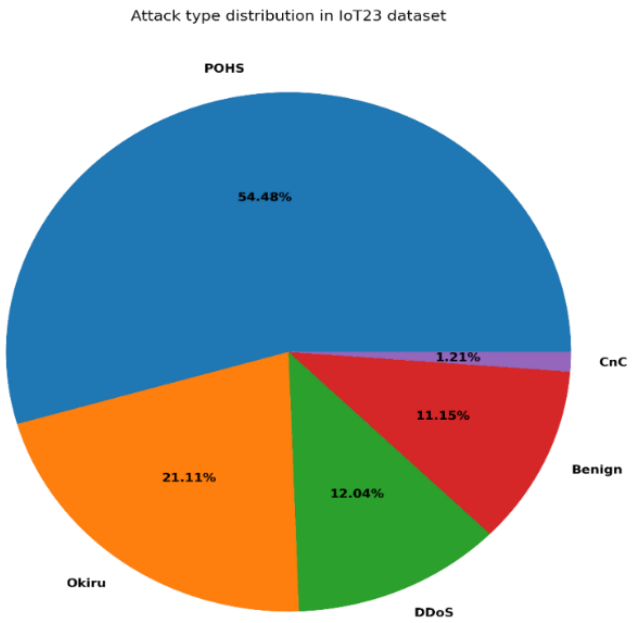


Figure 9: Attack Label Distribution in IoT-23 Dataset

5 Experiments and Methodology

This chapter describes the experimental work done in this study. It mainly discusses the methodology steps taken, such as data preprocessing, feature engineering, and model selection. A brief description of the ML algorithms used in this study is also provided.

5.1 Methodology

The methodological approach adopted in this research is shown in Figure 10. The methodology is divided into four stages. Each stage has different steps and performs various functions. The first stage is related to importing the data and pre-processing it for further use. The second stage is associated with the selection of features. In this stage, the most important features from the dataset are extracted. The model training and testing is performed in the third stage of this system. Finally, the last stage is designed to analyse the results of the models on unseen data and present the findings and analysis.

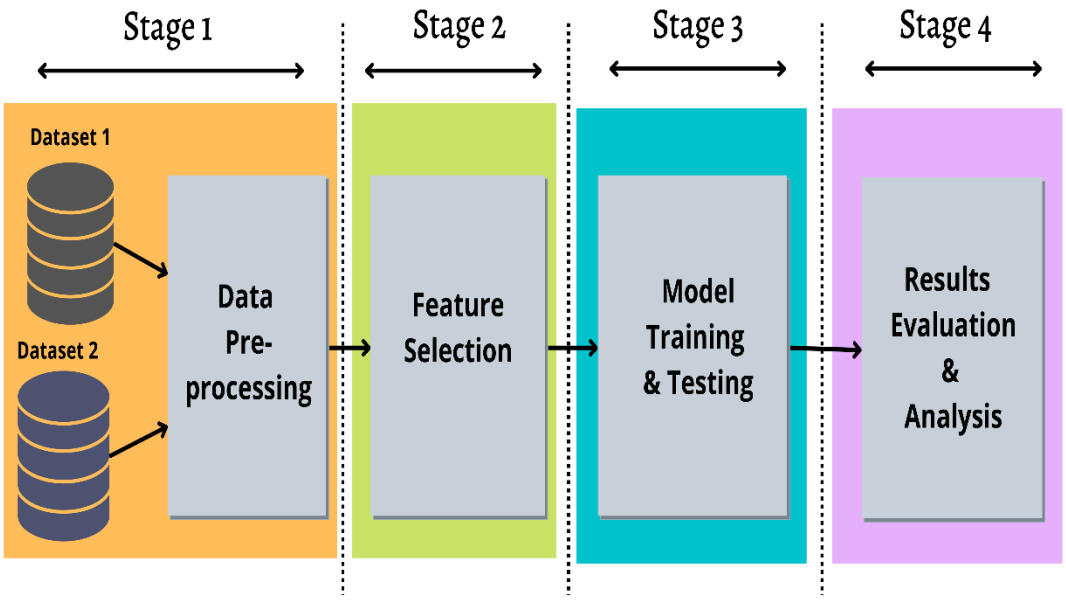


Figure 10: Methodology Stages

After correctly labelling the Ton-IoT dataset is provided by authors in csv format. The data is imported using panda's library and *pd.read_csv* function. It is important to note that for experimentation, the data was split into train-test split to avoid any overfitting. The train-test split ratio was kept as 80:20 as per the Pareto principle [59]. In addition, the hold-out validation method was used to measure the model's validity on test data.

In order to investigate the efficacy of supervised learning algorithms on Ton-IoT dataset, initially, the models were evaluated using all the features included in the dataset. The results achieved are presented the relevant section of the report. And in the second run, same models were evaluated using the remaining features selected using the feature selection technique. Although we could directly remove the features with low information gain value, but the results will all features are presented to analyse the impact on the training time of the models. The evaluation results and training and testing time costs of all the models were recorded.

5.1.1 Data Pre-processing

Data pre-processing is transforming the raw data into clean and well-format so that the ML algorithm can perform its task efficiently [60]**.** Pre-processing is the primary and essential step in ML as not all the time the data come in the standard form. The publicly available datasets can possess duplicate, null, and unnecessary attributes which have no importance for the ML task. Therefore, it is essential to remove the duplicate entries, clean the data, and eliminate the attributes of non-significance. Performing the pre-processing of the data results in less complexity for the ML model, less resource utilization in terms of time, and better model results.

5.1.2 Ton-IoT

The Ton-IoT dataset is imported using the pandas read_csv function and the cleaning process is performed. The dataset has 461,043 rows in total. The normal traffic flows comprise of 300000 approximately (65.07%), whereas the attack data is equal to 161043 (34.93%) of the total data. The dataset has 43 features, excluding the last two columns of label and type. The label columns are 0 and 1 for normal and malicious flows. At the same time, the type columns have nine categories of attacks. In addition, the missing values in the dataset are replaced with 0, as dropping these values might cause a loss of critical information [45]. The features such as 'src_ip', 'src_port', 'dst_ip', 'dst_port', and' ts' are removed in the beginning as these features can cause overfitting in prediction [50].

One of the main challenges while dealing with such datasets is to have many categorical features, and each column has several categories. Therefore, it requires encoding these categorical features into numerical as the ML algorithms do not work with categorical features directly. We convert the categorical features into numerical using the LabelEncoder() function from the sklearn library of python [61] . We use label encoding instead the one-hot encoding as the one-hot encoding increases the number of columns in the dataset by making a column of every single category. This significantly increases the dimensionality of the dataset and can affect the performance of ML. In addition, it also increases the complexity of the overall model [61].

ML algorithms often require data in the same range or scale. To scale the data in the same range, the standard normalization or scaling of the features using StandardScaler() class of the sklearn.preprocessing library is used. We used the unit variance method called the Z-score scaling method to standardize the columns into a normal distribution range [61]. It is preferred because it ensures no bias toward larger values in the dataset. The formula for unit variance is shown as (1):

$$z = \frac{x - u}{s} \quad (1)$$

Where $'x'$ represents the sample, 's' is the standard deviation and $'u'$, represents the mean of the data sample.

5.1.3 Feature Selection

One of the major challenges in building an accurate ML model is the selection of important features. Specifically, a network intrusion dataset can have a huge number of unwanted and redundant features with zero influence on the model's performance. Hence, it is vital to perform feature engineering on a large dataset to select the best features which influence the overall performance of the ML model. In addition, the feature selection will also reduce the computational cost of the model [62]. We selected relevant features using mutual information technique also called information gain. Information gain is a heuristic technique to choose the attribute while building decision trees. We used the mutual_info_classif() library of the sklearn to calculate the mutual information or dependency between the variables with respect to target. The information gain value of all the features is plotted in Figure 11. Higher values mean a higher dependency of the variable. If two variables are independent, the score is zero.

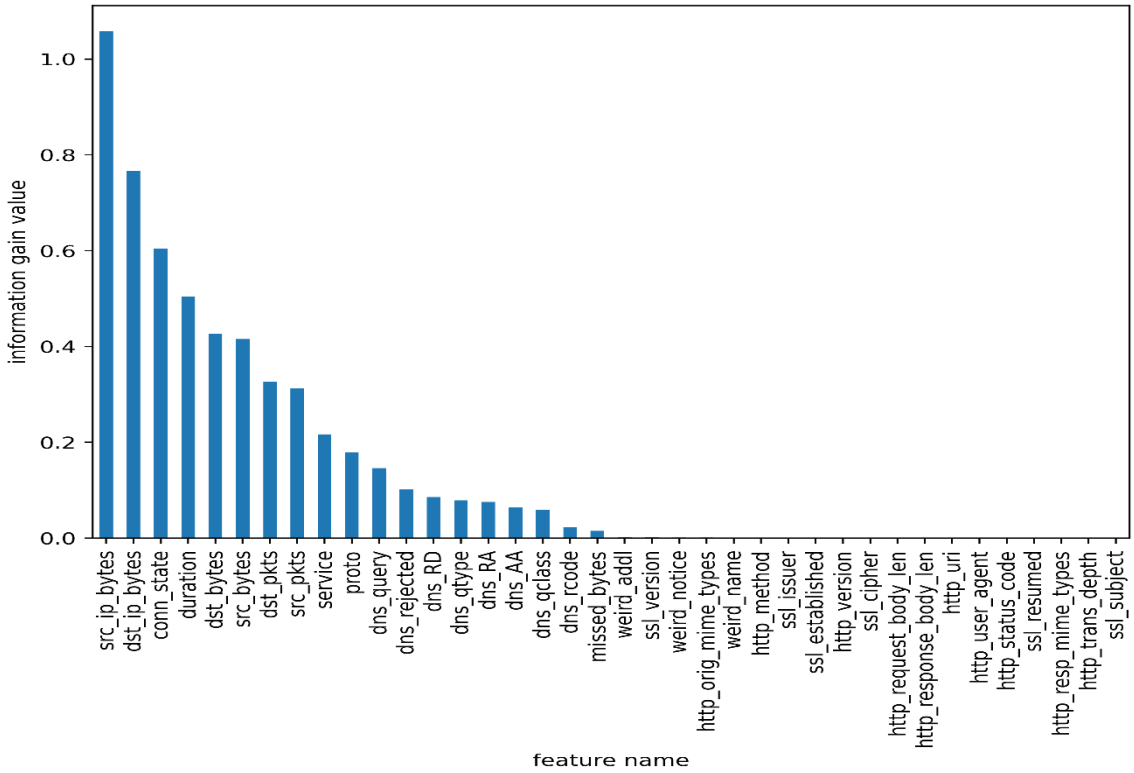


Figure 11: Feature Importance Values

5.1.4 **IoT-23 Dataset**

As discussed earlier, the IoT-23 dataset is one of the largest datasets in terms of data samples. Given our system's technical limitation, we decided to sub-sample the dataset and extract approximately one million samples. In our study, the labeled lighter version of the dataset is used. The dataset is provided in conn.log format. ML models cannot understand and process this file format directly. Hence it is required to be converted into readable form. Some researchers combined the label and detailed-label columns to reduce the complexity [63] . We also combined these columns during the pre-processing phase. In our case, the malware scenarios were combined to generate a new dataset. The Zeek log files are converted into csv format using panda's utilities. Specifically, 1244131 rows were extracted from the log.conn files and converted to a csv file using pandas' utility. This data contains almost all the botnet attacks executed in during the test-bed scenario. However, most of the packets consist of PortOfHorizontalScanattack. Therefore, the dataset is imbalanced in nature.

In addition, the three benign scenario were captured from real time device. Those scenarios are also combined to increase the total number of benign samples.

During prep-processing, detailed_label, tunnel_parents, uid, IP addresses, and port numbers, service and history columns were dropped [64]. Some of the hese features could cause an overfitting problem, and some were either unnecessary for further processing or completely null.

In addition, for simplification, we rename some of the labels. The' PartOfAHorizontalPortScan' was renamed '*POHS*', and Okiru-Attack was simplified as '*Okiru*'. The newly created dataset consists of all the BotNet attacks listed by the authors. However, the new dataset was highly imbalanced for some classes, such as C&C-HeartBeat, C&C-Torri, FileDownload, and few others were hardly in the hundreds.

Hence we dropped these labels as the number of samples of these labels was relatively insignificant. After removing those attack labels, only five labels have remained in the data. The remaining attack distribution is shown Figure 9. Similar to the Ton-IoT, this dataset also consisted of many null values replaced by 0. In addition, the dataset also consisted of categorical variables, which were encoded into numerical using LabelEncoder() class from sklean. Like Ton-IoT, the data was normalized using StandardScaler() class of the sklearn.preprocessing. It should be noted that no feature selection was performed in this dataset since the number of features is already limited. In addition, some of the features which could cause overfitting were already dropped.

5.1.5 Implementation Framework and Tools

The experiments were performed using Intel(R) Core(TM) i5-8365U CPU @ 1.60GHz, 1.90 GHz Dell laptop with 12GB RAM.The algorithms were coded using Python version 3.7 and Sklearn library of the python. The data preparation and pre-processing were performed using pandas and NumPy libraries of the python. To perform all these computations, Jupyter Notebook version 6.1.4 was used. In addition, many ML libraries in python, such as sklearn, matplotlib.pyplot, train_test_split were utilized for performing various tasks.

5.2 Machine Learning Models

Machine Learning can be supervised or unsupervised. The supervised learning uses labeled data to perform the task, whereas unsupervised learning can be applied to unlabeled data.

These methods can be applied depending on the application. The classification methods come under supervised learning, and clustering is called unsupervised learning.

In this study, various classification algorithms are used to predict output from labeled data. These methods include LR, DT, RF, NB, KNN, and XGB. Each method has its advantages and disadvantages. But all these algorithms are widely applied for task of classification. These classification methods are described in detail in the following sections.

5.2.1 Logistic Regression

LR [65] is basically the variation of linear regression. It is commonly used for classification problem, specifically for binary classification. Given an input variable, LR is used to predict the outcome of a non-continuous outcome. It uses the probability distribution to check which sample fits best to which category. The range of LR is bounded between 0 and 1. The mathematical intuition of the logistic regression is given in (2)

$$h_\theta(x) = \sigma\, \theta^T X \qquad (2)$$

Where h_θ is the hypothesis, X represents the input vector, σ is the sigmoid function given in (3) and θ^T shows the transpose of LR parameters.

$$\sigma(x) = \frac{1}{1 + e^{(-x)}} \qquad (3)$$

5.2.2 Decision Tree

DT [66] is a widely used supervised learning method for the purpose of predicting the categorical target variable. The goal of DT is to predict the variable by making a tree of decision rules from the input features. DT can be used for both i.e, binary classification as well as multiclass classification problems. The decision tree uses two methods of splitting criteria already programmed in sklearn library. The default splitting method is Gini; however, one can also choose entropy. In this study, we used Gini as our splitting criteria as given in (3). The max_depth was set to 20 to avoid any overfitting and the split criteria was set as best.

$$G(D) = \sum_{I=1}^{C} (P(i) * \big(1 - P(i)\big) \qquad (4)$$

Where D is the training dataset, C is the collection labels, and p(i) represents the proportion of samples with class label I. If the class is one in C, the Gini impurity is zero.

5.2.3 Random Forest

Random forest is a tree-based ensemble learning method [67]. It is arguably the most popular method among researchers and ML engineers for the purpose of classification. It provides the best results even without turning the hyperparameters a lot.

It creates the forest of the trees by combining various decision rules. It can also be used easily using sklearn library of python. The RandomForestClassifier() method of sklearn library provides the ability to tune the parameters such as max-depth, number of estimators, and splitting criteria [68]. The number of estimators was set to 200, whereas the split criteria, in this case, was also set as Gini impurity.

5.2.4 K-nearest Neighbors

KNN [66], is another supervised learning method that works completely different from the probability-based methods such as LR and NB. KNN groups the labels or target variables based on the similarity of using the simple distance-based formula. The number of neighbors is initiated in the beginning, and the distance is calculated on the new data point to decide which neighbor the new sample belongs to, based on the training dataset. A simple method to measure the distance between two points is the Euclidean distance. However, this method is very slow compared to the ensemble learning techniques. The sklearn provides two options of selecting the distance measurement technique i.e the Manhattan distance and Euclidean. In this study, Euclidean distance (5) is used.

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \qquad (5)$$

5.2.5 Gaussian Naïve Bayes

GNB [69] is a variation of Naïve Bayes algorithm, a probability calculation method that relies upon the Bayes Theorem to predict the outcome. These methods assume that the value of a particular variable in a classification problem is independent of the other variables.

Although it is not the most popular method for multiclass problems, still used due to its low computational cost and fast performance. GNB uses Gaussian normal distribution and supports continuous values. The mathematical expression GNB probability calculation is shown in (6)

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma^2}}\ e\hat{}(-\frac{(x_i - u_y)^2}{2\sigma_y^2} \qquad (6)$$

5.2.6 eXtreme Gradient Boosting

XGB [70] is also one of the most widely used gradient boosting methods for the task of classification. XGB is famous because of its ability to control the overfitting problem of a decision tree using a more regularized model formalization. The XGB can be easily implemented using XGBClassifier() in sklearn python [70]. XGB applies robust regularization methods to control the overfitting. Hence, usually, the default method can also produce the best results. It is a fast and efficient method in terms of resource utilization. This method is preferred to optimize the resources such as memory and hardware computation requirements. In our study, default parameters are used.

The XGB algorithm aims at optimizing the cost objective function using an additive approach [71]. The cost function is composed of the loss function (d) and regularization term (beta) as shown in (7)

$$\Omega(\theta) = \sum_{i=1}^{n} d(y_i, \hat{y_i}) + \sum_{k=1}^{K} \beta(f_k) \qquad (7)$$

The first term in equation (10) represents the loss and the second term is the regularization. Where $\hat{y_i}$ represents the predictive value, n is the number of instances, *K* is number of trees and *fk* is the tree from ensemble trees. The regularization term of the XGBoost algorithm is shown in (8)

$$\beta(\mathrm{f_t}) = \gamma \mathrm{T} + \frac{1}{2}\ \alpha \sum_{j=1}^{T} |c_j| + \lambda \sum_{j=1}^{T} c_j^2 \qquad (8)$$

Where γ is the minimum split loss reduction, λ represents the regularization terms and c is the weight associated to each leaf.

6 Results and Discussion

This chapter describes the results of the models implemented in this study. A detailed discussion about the evaluation metrics is presented. The results are divided into binary classification and multiclass classification problems. In the first section of this chapter, the evaluation metrics used in this study are briefly described. We used accuracy, precision, recall, and f1-score as the evaluation criteria for the ML models.

Moreover, the results of ML models using Ton-IoT datasets are discussed in the second part of this chapter. The classification results achieved with the IoT-23 dataset are discussed in the third section.

6.1 Evaluation Metrics

- Accuracy

Accuracy simply represents the total ratio or percentage of correctly classified instances from attack or normal traffic. It is represented as (9)

$$Accuracy = \frac{(TP + TN)}{TP + TN + FP + FN} \quad (9)$$

- Precision

Precision is the measure of attacks which were correctly classified out of total attacks. It is represented as follows (10)

$$Precision = \frac{TP}{TP + FP} \quad (10)$$

- Recall

Recall is the measure of total true positives identified by the model. In simple words, for all attacks that were actually the attacks, recall tells how many were correctly identified by the model. It is calculated as shown in (9)

$$Recall = \frac{TP}{TP+FN} \quad (11)$$

- F1-score

The fourth metric that we use to evaluate the models is the F1-score. It computes the weighted average of the accuracy and recall of the model. It can be represented as follows (12)

$$F1 - score = 2\,x\,\frac{precision * recall}{precision + recall} \quad (12)$$

6.2 Ton-IoT Results

This section presents the results of the ML algorithms on the Ton-IoT dataset.

6.2.1 Binary Classification

The Ton-IoT dataset contains the binary labels as well as the multiclass labels. The binary labels are represented as benign and malicious data. In contrast, multiclass labels are distributed in various types of attacks. The performance of the models in the binary class can be seen in Table 3.

It can be observed that LR and GNB are not the best choices for classifying the malicious and benign samples in this dataset. These two models provide 75% and 44% accuracy scores, respectively. This result was achieved using the default parameters of the models imported using sklearn library of python. However, these algorithms are fast in terms of time computation on the training dataset. In addition, the AUC score of the LR and GNB is achieved to be 0.86 and 0.62, respectively. The ROC curve of LR and GNB is shown in Figure 12.

Table 3:Binary Classification Results of all Models

Model	Accuracy	Precision	Recall	F1-score	Train time (s)	testing time (s)
LR	0.745	0.66	0.57	0.61	33.09	0.02
GNB	0.436	0.38	0.98	0.55	0.38	0.22
RF	0.992	0.986	0.992	0.989	106.24	5.81
DT	0.991	0.984	0.991	0.988	2.71	0.06
KNN Model	0.991	0.984	0.990	0.997	0.06	1925.26
XGB	0.992	0.995	0.992	0.988	18.89	0.07

On the other hand, the tree-based classifiers such as DT, RF, and XGB provided an accuracy of 0.99 with varying time costs. The precision, recall, and f1-score of these models are above 0.98. The training times for DT, RF, and XGB are 2.71, 106.24, and 18.89 seconds respectively. The ROC curve of these classifiers is shown in Figure 13. It can be observed from the results that all tree-based algorithms have provided almost similar results on a binary classification problem. However, the best results were achieved with RF and XGG.

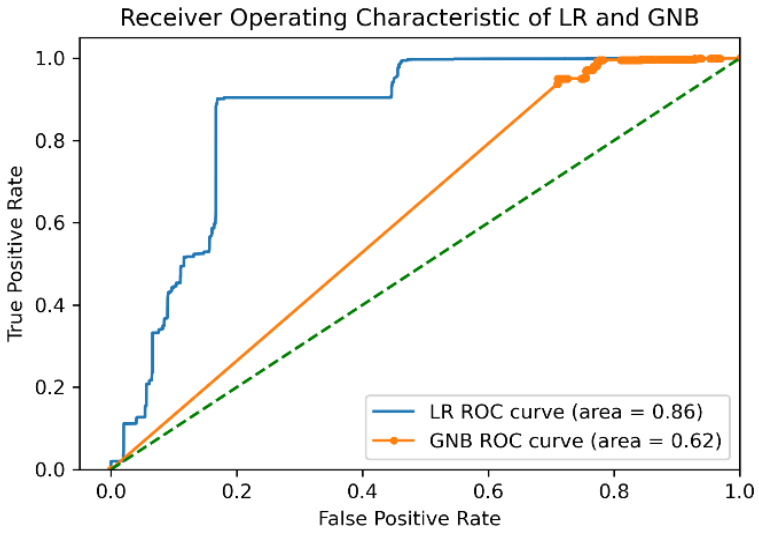


Figure 12: ROC characteristics of LR and GNB

Moreover, the performance of the KNN classifier on Ton-IoT data is also comparable to the tree-based methods. The KNN-classifier provided an accuracy of 0.99, while the precision score was 0.98.

In terms of training and testing time, KNN is the costliest algorithm, with a training time of 0.6 seconds and a testing time of 1925 seconds. The distance measurement method was kept as Euclidean, and the total number of neighbors was set to 5 for KNN.

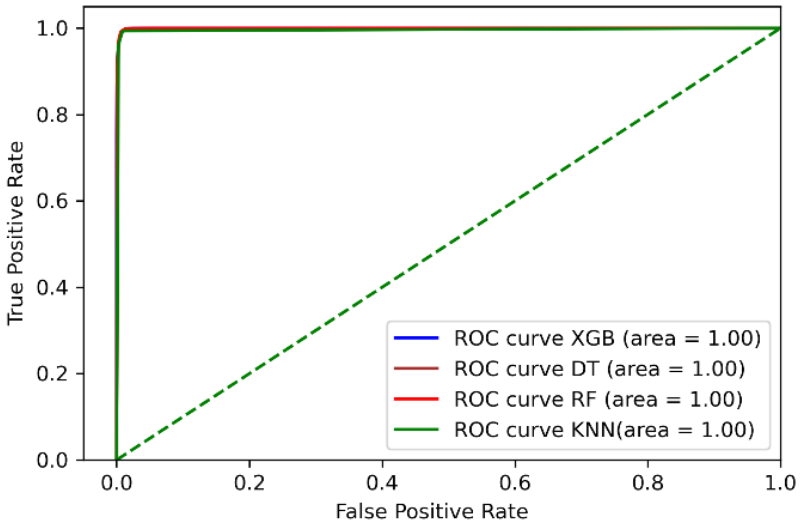


Figure 13:ROC characteristics of DT, RF, KNN, and XGB

In addition, the confusion matrix plot of these models for binary classification is shown in Figure 14. Confusion matrix is another evaluation method commonly used in classification problems. It tells about the actual and the predicted labels. It should be noted that here '0' represents the normal traffic, whereas the '1' represents the malicious traffic. It can be noted from Figure 14(a) that LR has classified true (normal) labels as false. More than 13000 normal samples were misclassified as the attack samples, and more than 9000 samples were predicted as a false negatives. Similarly, the GNB model misclassified almost all normal traffic as an attack. It can be seen in Figure 14 (b) that GNB predicted the majority of samples as false negatives. As given in Figure 14(c) and (d), the confusion matrix of DT and RF looks almost similar. The prediction error rate of these classifiers is shallow. Finally, the KNN and XGB classifiers also predicted the target samples correctly. Figure 14(e) and (f) show the confusion matrix of the KNN and XGB. Only less than 500 samples were misclassified as either false positive or false negative in both cases. Therefore, the TPR rate of these Is higher and FPR is lower.

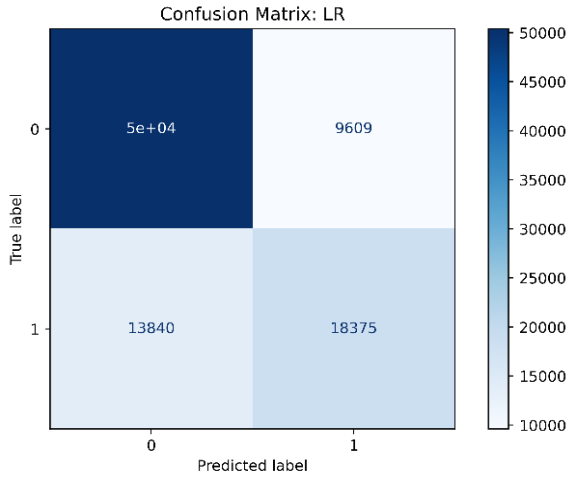


Figure 14 (a)

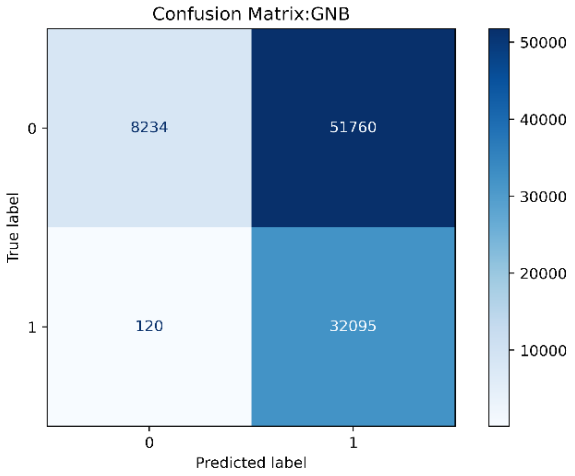


Figure 14 (b)

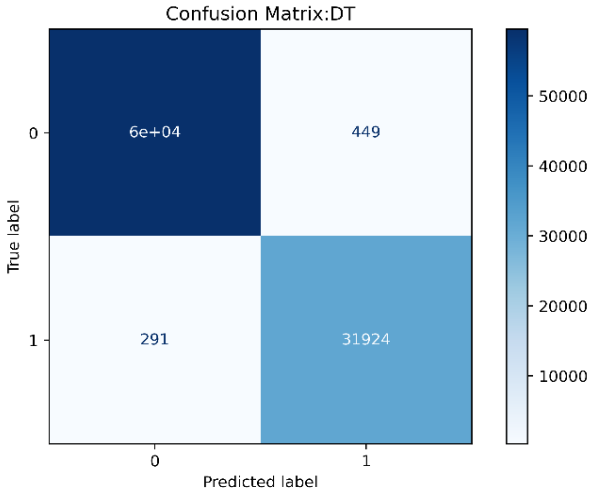


Figure 14 (c)

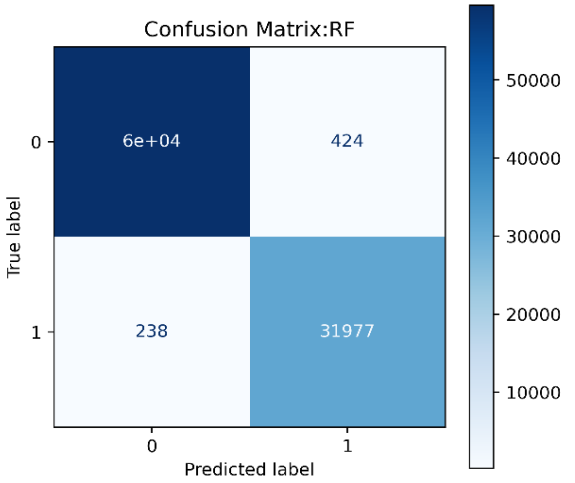


Figure 14 (d)

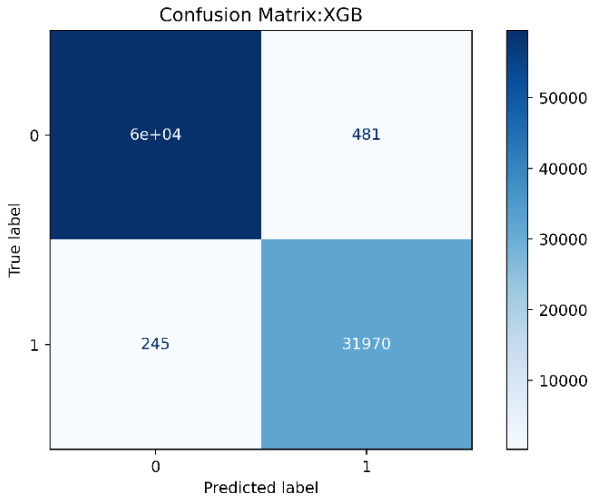


Figure 14 (e)

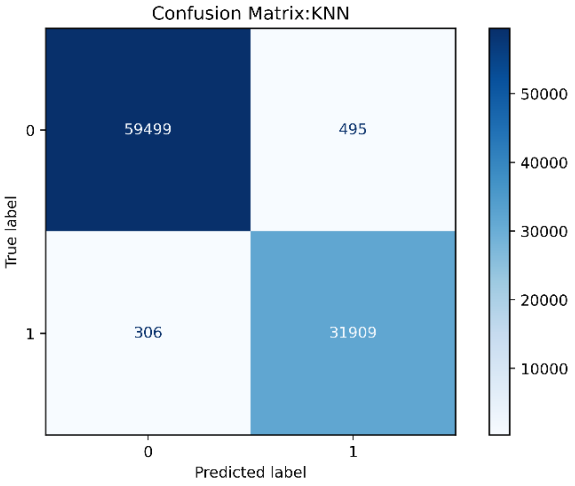


Figure 14 (f)

Figure 14: Confusion Matrix (Binary Classification)

6.2.2 **Multiclass classification**

Similar algorithms are also tested on multiclass target variables. Initially, the results are evaluated on the pre-processed data with all 38 features. It is found that the tree-based algorithms perform excellently with this baseline dataset. The LR can cause a convergence problem if the number of iterations is not optimum. In this study, the parameters of the LR were set as follows; the solver was set as 'sag', because the linear solver doesn't work with multinomial classes. In addition, the default class for the multiclass problem is one-v-rest or 'ovr', we changed this to multinomial, and the number of iterations was set to 300. However, it is concluded that LR is not the best choice, especially for the multiclass problem on the Ton-IoT dataset. It reaches an accuracy of approximately 65%, precision, recall, and F1-score of 0.52, 0.649, and 0.55, respectively.

Table 4: Evaluation Metrics of Models on All Features

Model	Accuracy	Precision	Recall	F1-score
LR	0.649	0.522	0.649	0.548
GNB	0.076	0.684	0.076	0.047
RF	0.983	0.984	0.983	0.983
DT	0.982	0.982	0.982	0.982
KNN	0.981	0.982	0.981	0.982
XGB	0.984	0.985	0.984	0.984

The best performances were achieved with DT, RF, KNN, and XGB algorithms. It can be noted from Table 4, that the accuracy of these models is above 98%. At the same time, the XGB provides the highest accuracy of 98.4%, followed by RF (98.3%) and DT (98.2%). The KNN classifier provided an accuracy of 98.1%. It can further be observed that the other metrics, such as precision, recall, and f1-score, remain almost similar for all the models. The GNB is the worst performing algorithm of all these ML models, with an accuracy of just 7% and a precision of 68%. It is probably because of the imbalance in the dataset. Hence, GNB cannot be chosen as the classification algorithm for multiclass intrusion detection problems. The evaluation metrics can also be seen in Figure 15.

We took a step ahead by investigating the impact of dropping the number of features with the lowest gain ratio information. All the features with shallow mutual importance scores were dropped to see the impact on models' training time and accuracy.

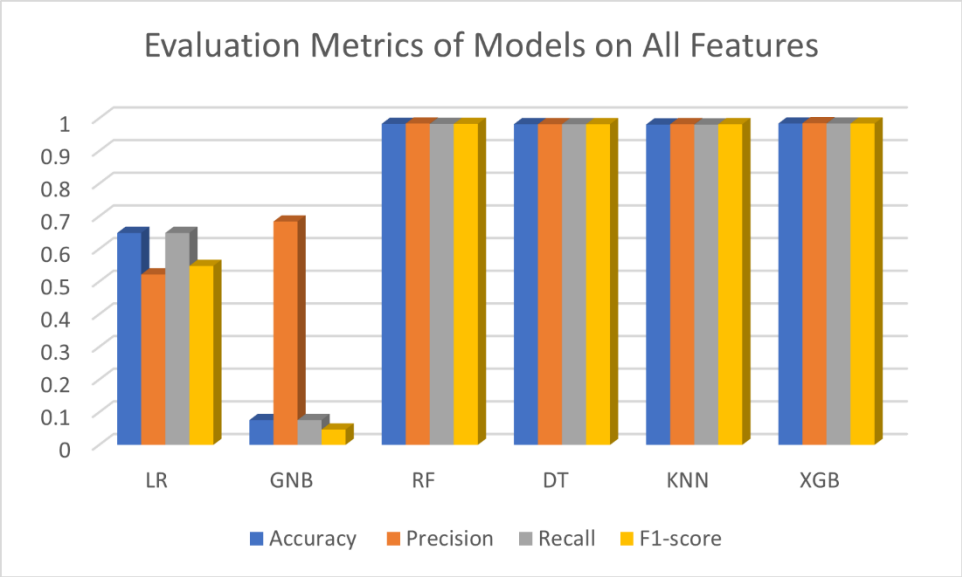


Figure 15: Evaluation Metric on All Features (Ton-IoT)

The performance of the models using the 18-best features can be seen in Table 5. It can be observed that the performance of these models doesn't improve or degrade a lot by dropping the number of features. However, the training time and model complexity decrease essentially. It is found that with a further reduction of the features from the dataset, the accuracy performance of the model may start reducing. The evaluation metrics can also be seen from Figure 16.

Table 5: Performance Metric with 18 Features

Model	Accuracy	Precision	Recall	F1-score
LR	0.649	0.659	0.65	0.65
GNB	0.12	0.71	0.12	0.10
RF	0.983	0.984	0.983	0.983
DT	0.982	0.982	0.982	0.983
KNN	0.981	0.982	0.981	0.982
XGB	0.984	0.985	0.984	0.984

On the other hand, one can observe the impact on the training time of all the models from Table 6. It can be derived that the training time of all the models keeps reducing with the reduction in the total number of features. As far as the time cost of the models is concerned, GNB, and DT consume the least training time, whereas the LR, RF, and XGB consume comparatively more time than others. It should be kept in mind that the KNN uses less time during training, but it is the most expensive model for prediction.

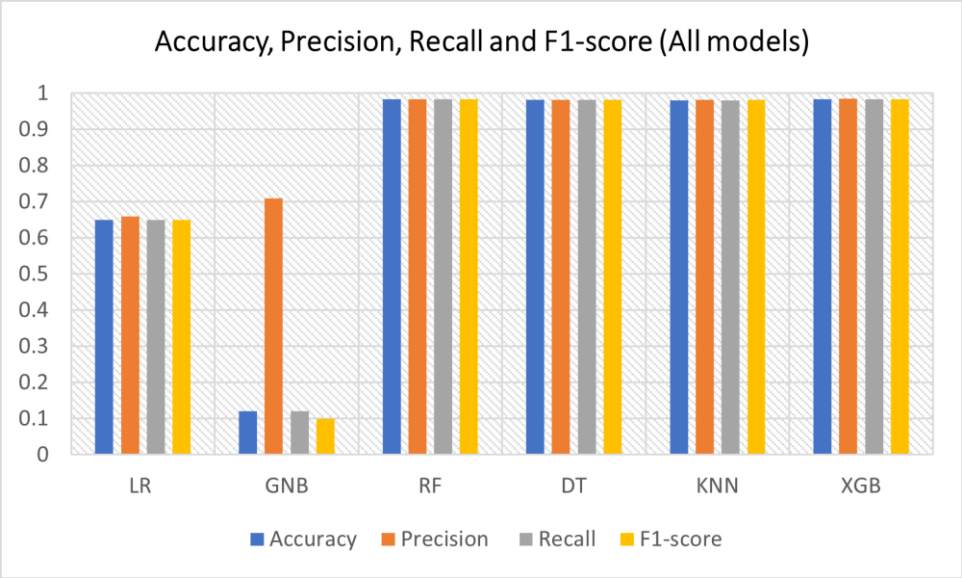


Figure 16: Evaluation Metrics of All Models (18-features)

However, reducing the number of features results in less time and memory computations. The training time of LR with 18 features was reduced from 128.81 seconds to 41 seconds. Similarly, DT and RF also show a reduction in training time at 18 features. A significant change from 2866 seconds to 1990 seconds can be observed in the prediction time of KNN.

Table 6: Model Training Time (second) Vs Number of Features

Model	features38	features18
LR	128.81	41
GNB	1.38	0.15
DT	1.98	1.3
RF	83.71	68
KNN	2866	1990.56
(XGB)	238.8	148.48

It is important to note that not all the features in Ton-IoT contribute to the accurate detection of network attacks. Hence many of the features can be dropped. However, certain features cannot be dropped in any case while detecting the IoT BotNet attacks.

These features include: proto, service, src_bytes, dst_bytes, conn_state, duration, src_pkts, src_ip_bytes, dst_pkts, dst_ip_bytes. These features carry important information, such as the type of protocol that can help understand the packet and the attacker's intention. In addition, the number of source packet bytes can tell the size of the packets, which can be compared with the normal packet size and the malicious packet size. Malicious packet bytes would be different or sometimes lesser than the original legitimate source.

6.2.3 Accuracy of Models in Each Class

The model's performance was further evaluated in terms of accuracy in each class. There are ten different class labels in the Ton-IoT dataset. The authors correctly labelled these attacks during the collection of the dataset. A further evaluation of the models for these types of attacks are carried out. It can be observed from Table 7, that some of the attacks were entirely not detected or classified by the models. Hence their accuracy is 0. For example, in the case of GNB, the overall accuracy of this model is the lowest. But it completely misclassified some of the attacks, such as backdoor, injection, and XSS. However, its accuracy with a ransomware attack is 100%.

In addition, LR has also performed. These models could not detect some attacks because of the imbalance of the data. Since we split our data into an 80:20 ratio, some attacks might likely be in the training dataset but not the test set.

Table 7: Accuracy of Models in Each Class Label

Class Labels	GNB	LR	DT	RF	KNN	XGB
Backdoor	0	0	*0.998*	*0.999*	0.999	0.998
DdoS	0.004	0.25	0.92	0.974	0.972	0.977
DoS	0.0002	0	0.982	0.982	0.98	0.984
Injection	0	0.001	0.969	0.971	0.964	0.969
MITM	0.08	0	0.705	0.785	0.705	0.7723
Normal	0.003	0.94	0.992	0.993	0.991	0.992
Password	0.0017	0.0037	0.979	0.979	0.974	0.979
Ransomware	1	0	0.933	0.933	0.932	0.936
Scanning	0.001	0	0.99	0.992	0.992	0.994
XSS	0	0.569	0.899	0.907	0.901	0.918

Similarly, it was also unable to detect some labels such as backdoor, DdoS, MITM, and scanning. But LR classified the 'normal' class perfectly with an accuracy of 94%, followed by XSS with 54%. On the other hand, DT predicted almost all the categories of attacks

with outstanding performance. The best classification accuracy of over 99% was achieved with the backdoor, 'normal', and scanning class. At the same time, the lowest accuracy (75%) was achieved with MITM. Similarly, RF performed better with all the classes except MITM. It provided better accuracy (78.5%) than DT for the MITM label. A similar pattern can be observed in the case of the KNN and XGB classifiers. The KNN and XGB classified all the labels with a high accuracy rate.

6.2.4 **Results Comparison**

The investigation of the efficacy of ML algorithms on Ton-IoT is still yet to be fully exploited. It is because the Ton-IoT dataset is released recently. Hence, not much has been explored about this dataset as far as ML algorithms are concerned. We compare our results with recently published research in [50]. The authors in this work proposed various models based on the algorithms we implemented in this work.

The authors used the Chi-square test for feature selection and also implemented SMOTE method for data balancing. In our case, the feature selection is performed using mutual information. The comparison of the results is shown in Table 8**.** It can be observed that our proposed models and existing work perform more or less equally. In our case, DT and RF models provide better accuracy of 98.2% and 98.3%, respectively.

In comparison, these models in the current work provide an accuracy of 93.4% and 93.7%, respectively. The proposed LR method in the existing one offers better accuracy than our model, whereas the XGB algorithm provides similar accuracy in both works.

Table 8: Results Comparison with Existing Work

	[50]	Our work	[50]	Our work	[50]	Our work	[50]	Our work
Model	Accuracy	Accuracy	Precision	Precision	Recall	Recall	F1-score	F1-score
LR	0.77	0.65	0.509	0.512	0.509	0.65	0.509	0.548
DT	0.934	0.982	0.934	0.983	0.934	0.982	0.875	0.982
RF	0.937	0.983	0.937	0.984	0.937	0.983	0.937	0.983
KNN	0.979	0.981	0.979	0.981	0.979	0.981	0.979	0.981
XGB	0.983	0.984	0.983	0.985	0.983	0.984	0.983	0.984

6.3 IoT-23 Results

As mentioned earlier, IoT-23 is one of the largest IoT network malware datasets available. Due to the technical limitations of our system, a portion of the dataset is used for experimentation in this study. Although samples from all the scenarios were combined, some of the attack labels were relatively scarce. Hence, those labels needed to be dropped for proper experimentation and evaluation of the models.

After pre-processing and cleaning the dataset, approximately 1.2 million data samples were saved for training and testing. In addition, the two label columns were combined to reduce the complexity of the data. Only five class labels were used for classification, and the results are presented in this section. Like Ton-IoT, the models tested on the IoT-23 are evaluated using accuracy, precision, recall, and f1-score.

The results of the model models can be seen In Table 9. It can be observed that GNB and LR models are again behind other models in terms of accuracy. The accuracy achieved with these models was noted to be 0.384 and 0.648 for GNB and LR, respectively.

Simultaneously the precision, recall, and f1-score for GNB is reported to be 0.622, 0.382, and 0.325, respectively. However, this model is quite fast in the training process. A similar pattern of the results can be seen with LR. The LR models provided a 0.679, 0.684, and 0.609 for precision, recall, and f1-score.

Table 9: IoT-23 Performance Evaluation Metrics

Model	Accuracy	Precision	Recall	F1-score	Training Time(s)
GNB	0.384	0.622	0.382	0.325	1.01
LR	0.648	0.679	0.648	0.609	70.79
DT	0.902	0.917	0.902	0.9	5.63
RF	0.902	0.917	0.902	0.9	25.806
KNN	0.862	0.875	0.862	0.864	921
XGB	0.902	0.916	0.902	0.9	184.57

On the other hand, the DT and other ensemble learning methods i.e RF and XGB, accurately classified the target variable. The DT provided an accuracy of 0.902; a similar accuracy score was achieved with the RF and XGB. However, KNN did not reach the best results on IoT-23 compared to Ton-IoT. The KNN provided an accuracy score of 0.862.

Finally, the XGB method was also able to achieve an accuracy score similar to DT and RF. Hence, it can be concluded that ensemble learning methods successfully classify the target variables in the IoT-23 dataset.

The precision, recall, and f1-score values and training time of these models can be seen in Table 9 . Figure 17 also shows the performance evaluation metrics of all models on the IoT-23 dataset.

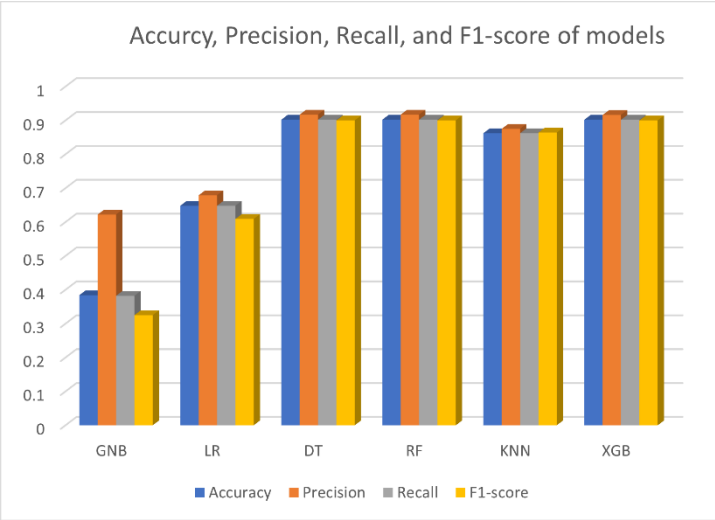


Figure 17: IoT-23 Evaluation Metrics

Further evaluation of these models is carried out for each class label. The performance of these models for the accuracy of each target variable is presented in Table 10 . It can be observed that LR was able to classify DDoS and POHS labels correctly. However, the accuracy of the model for other class labels is not good. A similar trend can be seen with GNB; however, GNB models were able to classify the Okiru attack with an overall accuracy of 0.99. The ensemble learning method DT, RF, and XGB provided similar patterns while recognizing each class label. However, KNN was able to classify only CnC with high accuracy, but it found it difficult to detect other attacks.

Table 10: IoT-23 Accuracy on Each Label Class

	Accuracy for each class label					
Class label	LR	GNB	DT	RF	KNN	XGB
Benign	0.1	0.1	0.78	0.78	0.78	0.78
CnC	0	0.11	0.99	0.99	0.98	0.99
dDoS	0.81	0.81	0.81	0.81	0.82	0.81
Okiru	0.37	0.99	0.75	0.75	0.88	0.75
POHS	0.85	0.12	0.99	0.99	0.87	0.99

6.4 Discussion

The IoT networks can be structured in a distributed hierarchy. The network can be divided into different layers, including edge, fog, and cloud networks. Hence, heterogeneity is the main element of IoT network systems. The datasets used in this study were selected for fulfilling the criterion of heterogeneity. These datasets include traffic from real-time IoT devices and several types of malicious attacks. In addition, the Ton-IoT dataset contains real-time attacks, which are more frequent on IoT systems. On the other hand, IoT-23 contains attacks from real IoT BotNets, such as Mirai, Okiru, and Bashlite. The use of these datasets makes this study more robust. However, the network heterogeneity does not seem to have affected the overall performance of classification algorithms. The main reason for ensemble learning providing excellent results is the inclusion of important feature sets in the dataset.

Many features in both datasets were not useful and could cause overfitting of the model on training data. Hence it was necessary to drop those columns to avoid bias. The models were able to generalize well on the unseen data and provide excellent results. The datasets used in the study are collected from packet captures. In both datasets, the statistical features which represent the number of bytes from source and destination appear to have a higher impact on the overall results. It is also because the main difference between the malicious and benign is spotted using IP header payload, duration between two consecutive packets, and transmission protocol. The difference in the size of the payload could alarm a network analyser about the threat. Therefore, these features are of high importance and contribute more to the correct classification of attacks. On the other hand, it is quite interesting to note that many feature categories, especially in the Ton-IoT dataset, do not really help ML algorithms to detect attacks. These categories include http and ssl features. Since this dataset also consists of DNS attacks, the DNS activity features are found to have a higher dependency.

6.5 Threats to Validity

All necessary steps were taken to avoid any validity threats as much as possible. During the pre-processing phase, all those features were removed that could cause overfitting. In addition, the data is split into training and testing sets to validate the model on unseen data. Further, the data is shuffled to ensure that the model does not receive the same samples during the testing. Besides, to make the results more robust and avoid bias in the prediction,

the hyperparameters of the models such as maximum depth, number of estimators, and number of neighbors were controlled.

However, one can observe the imbalance in the datasets, and this imbalance is logical because it is impossible to have a balance network dataset in a real-world scenario. The IoT-23 dataset used in this study was also highly imbalanced. We removed some of the classes which were significantly less in quantity. Still, POHS samples seemed to have a high percentage. But that doesn't seem to have affected the overall results of the model since all the features which were causing the overfitting were removed in this dataset as well. However, an imbalance in the dataset might affect the performance of some algorithms. And this is quite visible in case of LR and GNB. However, the ensemble learning methods perform well on even imbalanced datasets.

The IoT-23 is a massive dataset, and we only use 1.2 million samples approximately. More samples and classes extracted from the log files might alter the results. The time of the day (ts) feature in the IoT-23 dataset seems to have high importance and contributes to higher accuracy, precision, and recall scores. Although our models do not produce biased results overall, 'ts' might result in overfitting unintentionally for some models since it has many unique values.

Finally, the performance of models such as RF, DT, KNN, and XGB is well established in the classification domain. Also, evaluation metrics such as accuracy, precision, recall, and f1-score are also widely accepted. Hence, we are confident that the models used in this study are able to generalize well on unseen data.

7 Conclusion and Future Work

This chapter summarizes the research contribution of this study. Moreover, final concluding remarks and proposed future work are also provided in the later sections.

7.1 Research Contribution

The IoT network malware detection is an emerging field of research. It is a fast and dynamic field; hence many research works are being published. However, to the best of our knowledge, this is the first time where these two heterogeneous datasets are used for the analysis.

We highlight our contribution in the following points

- A detailed overview of the current state of art in IoT malware detection and classification are presented
- Specific consideration is given to the current IoT BotNet attacks
- To the best of our knowledge, this is the first time where these two real-time datasets with high heterogeneity levels in terms of network structure, number of devices, and number of attacks are used for the analysis
- The feature selection process is performed to select the important features in the dataset.
- Important features in the dataset are highlighted
- Various classification algorithms are evaluated on real-time IoT network datasets.
- Detailed analysis and discussion on the performance of classification algorithms are provided

7.2 Concluding Remarks

IoT networked systems are facing one of the biggest challenges in the form of malware. One of the main reasons for being an easy attack for hackers is to have various vulnerabilities in the IoT devices. In this work, we studied several IoT security challenges and opportunities for AI and ML to protect these devices. ML algorithms require good quality data. After reviewing many publicly available IoT network datasets, two heterogeneous and real-time IoT network datasets were selected for this study. These datasets are named Ton-IoT and IoT-23. In addition, supervised learning methods including DT, RF, GNB, KNN, LR, and XGB were selected for experimentation.

Furthermore, both datasets were pre-processed before training and testing. A feature selection method was also implemented to separate the best features from the Ton-IoT dataset.

It is found that not all the features contribute to the classification of attacks. Hence those features can be dropped to reduce the training time and the computational complexity of the models. A hold-out validation method was used to validate the models learning test data.

It is concluded that not all the models implemented in this study provide perfect classification results. Among the selected models, LR and GNB are suboptimal choices for malware classification using the above-mentioned datasets. At the same time, the best results were achieved with the tree-based ensemble learning methods and K-Nearest neighbors.

It is recommended that classification methods such as DT, RF, KNN, and XGB can provide excellent result when used for classification IoT network attacks in real-time. These models can detect attacks with high accuracy given the proper data.

7.3 Future Work

For future work, there are many IoT network datasets publicly available. Transfer learning is an ML phenomenon that focuses on learning from one problem and applying the knowledge to similar context problems.

It is highly suggested to train models using these heterogenous datasets and test on other similar publicly available datasets. This would enhance the efficacy of the models and solve the security problems of IoT systems.

In addition, more such data from the industry could be helpful in training the models and ensuring the quality of prediction. The IoT requires more research on different aspects of security so that industry standards can be formalized with regard to important features and attacks.

Bibliography

[1] P. Michalski, Z. Wisniewski, and J. Gralewski, *To grow or not to grow - The strategic plan for acquisition and integration*, vol. 783. 2019.

[2] R. Ahmad and I. Alsmadi, "Machine learning approaches to IoT security: A systematic literature review," *Internet of Things*, vol. 14, p. 100365, 2021, doi: 10.1016/j.iot.2021.100365.

[3] B. Insider, "The internet of things 2020:here's what over 400 IoT decision-makers say about the future of enterprise connectivity and how IoT companies can use it to grow revenue," 2020. [Online]. Available: https://www.businessinsider.com/internet-of-things-report.

[4] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The social internet of things (SIoT) - When social networks meet the internet of things: Concept, architecture and network characterization," *Comput. Networks*, vol. 56, no. 16, pp. 3594–3608, 2012, doi: 10.1016/j.comnet.2012.07.010.

[5] Symantec, "Internet Security Threat Report VOLUME 24, February 2019," *Netw. Secur.*, vol. 21, no. February, p. 61, 2019, [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1353485805001947.

[6] A. Marzano *et al.*, "The Evolution of Bashlite and Mirai IoT Botnets," *Proc. - IEEE Symp. Comput. Commun.*, vol. 2018-June, pp. 813–818, 2018, doi: 10.1109/ISCC.2018.8538636.

[7] Council to Secure the Digital Economy (CSDE), "International Botnet and Iot Security Guide," 2021, [Online]. Available: https://securingdigitaleconomy.org/wp-content/uploads/2021/03/CSDE-2021-Botnet-Report-March-24-2021.pdf.

[8] J. Teicher, "The little-known story of the first IoT device," *IBM*, 2018. https://www.ibm.com/blogs/industries/little-known-story-first-iot-device/ (accessed Feb. 20, 2022).

[9] K. Ashton, “That ‘Internet of Things’ Thing,” 2009. https://www.itrco.jp/libraries/RFIDjournal-That Internet of Things Thing.pdf (accessed Feb. 20, 2022).

[10] ITU, “Overview of the Internet of things (Recommendation ITU-T Y.2060),” 2012. [Online]. Available: https://www.itu.int/rec/T-REC-Y.2060-201206-I.

[11] H. Minn, M. Zeng, and V. Bhargava, “Towards a definition of the {Internet of Things (IoT)},” *IEEE Internet Initiat.*, pp. 1–86, 2015.

[12] K. L. Lueth, “Top 10 IoT applications in 2020,” *IoT Analytics*, 2020. https://iot-analytics.com/top-10-iot-applications-in-2020/ (accessed Feb. 25, 2022).

[13] M. Lombardi, F. Pascale, and D. Santaniello, “Internet of things: A general overview between architectures, protocols and applications,” *Inf.*, vol. 12, no. 2, pp. 1–21, 2021, doi: 10.3390/info12020087.

[14] P. Malhotra, Y. Singh, P. Anand, D. K. Bangotra, P. K. Singh, and W. C. Hong, “Internet of things: Evolution, concerns and security challenges,” *Sensors*, vol. 21, no. 5, pp. 1–35, 2021, doi: 10.3390/s21051809.

[15] J. Mohanty, S. Mishra, S. Patra, B. Pati, and C. R. Panigrahi, *IoT Security, Challenges, and Solutions: A Review*, vol. 1199. Springer Singapore, 2021.

[16] K. M. Sadique, R. Rahmani, and P. Johannesson, “Towards security on internet of things: Applications and challenges in technology,” *Procedia Comput. Sci.*, vol. 141, pp. 199–206, 2018, doi: 10.1016/j.procs.2018.10.168.

[17] van S. Mark and Y. J. Joshi, *The IoT Security Landscape*. 2019.

[18] IETF, “Internet Security Glossary,” 2007. https://datatracker.ietf.org/doc/html/rfc4949 (accessed Feb. 10, 2022).

[19] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, “Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations,” *IEEE Commun. Surv. Tutorials*, vol. 21, no. 3, pp. 2702–2733, 2019, doi: 10.1109/COMST.2019.2910750.

[20] “Internet of Things (IoT) Top 10,” 2018. https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project#tab=IoT_Top_10 (accessed Feb. 20, 2022).

[21] “Router and IoT Device Vulnerabilities: Solutions to a Design Flaw,” 2022.

https://www.iotsecurityfoundation.org/router-and-iot-device-vulnerabilities-solutions-to-a-design-flaw/ (accessed Feb. 20, 2022).

[22] I. Ali *et al.*, "Systematic Literature Review on IoT-Based Botnet Attack," *IEEE Access*, vol. 8, pp. 212220–212232, 2020, doi: 10.1109/ACCESS.2020.3039985.

[23] B. Krebs, "Source Code for IoT Botnet 'Mirai'' Released — Krebs on Security,'" *Brian Krebs's cyber-security blog*, 2016. https://krebsonsecurity.com/2016/10/source-code-for-iot-botnet-mirai-released/ (accessed Apr. 21, 2022).

[24] Unixfreaxjp, "Malware must die," 2020. https://github.com/unixfreaxjp/malwaremustdie/blob/master/etc/IoT_Botnet_Monitoring_Report.md (accessed Apr. 20, 2022).

[25] TrendMicro, "Bashlite Updated with Mining and Backdoor Commands. Trend Micro," *Section: Research*. https://www.trendmicro.com/en_us/research/19/d/bashlite-iot-malware-updated-with-mining-and-backdoor-commands-targets-wemo-devices.html (accessed Apr. 20, 2022).

[26] Avast, "New Torii Botnet uncovered, more sophisticated than Mirai." https://blog.avast.com/new-torii-botnet-threat-research (accessed Apr. 22, 2022).

[27] S. Edwards and I. Profetis, "Hajime: Analysis of a decentralized internet worm for IoT devices," 2016.

[28] A. D. Vaibhav Singhal, Ruchna Nigam, Zhibin Zhang, "New Mirai Variant Targeting Network Security Devices." https://unit42.paloaltonetworks.com/mirai-variant-iot-vulnerabilities/ (accessed May 15, 2022).

[29] R. Nigam, "Unit 42 Finds New Mirai and Gafgyt IoT/Linux Botnet Campaigns." https://unit42.paloaltonetworks.com/unit42-finds-new-mirai-gafgyt-iotlinux-botnet-campaigns/ (accessed Apr. 21, 2022).

[30] L. ARSENE, "Hide and Seek IoT Botnet Learns New Tricks: Uses ADB over Internet to Exploit Thousands of Android Devices," 2018. https://www.bitdefender.com/blog/labs/hide-and-seek-iot-botnet-learns-new-tricks-uses-adb-over-internet-to-exploit-thousands-of-android-devices/ (accessed Apr. 21, 2022).

[31] Y. Rootkiter, "HNS Botnet Recent Activities." https://blog.netlab.360.com/hns-

botnet-recent-activities-en/ (accessed Apr. 21, 2022).

[32] M. Abdullahi, Y. Baashar, H. Alhussian, A. Alwadain, and N. Aziz, “Detecting Cybersecurity Attacks in Internet of Things Using Artificial Intelligence Methods : A Systematic Literature Review,” pp. 1–27, 2022, doi: doi.org/10.3390/electronics11020198.

[33] R. Ahmad and I. Alsmadi, “Machine learning approaches to IoT security: A systematic literature review,” *Internet of Things (Netherlands)*, vol. 14, 2021, doi: 10.1016/j.iot.2021.100365.

[34] K. Shinan, K. Alsubhi, A. Alzahrani, and M. U. Ashraf, “Machine learning-based botnet detection in software-defined network: A systematic review,” *Symmetry (Basel).*, vol. 13, no. 5, pp. 1–28, 2021, doi: 10.3390/sym13050866.

[35] S. A. Mulay, P. R. Devale, and G. V. Garje, “Intrusion Detection System Using Support Vector Machine and Decision Tree,” *Int. J. Comput. Appl.*, vol. 3, no. 3, pp. 40–43, 2010, doi: 10.5120/758-993.

[36] N. Farnaaz and M. A. Jabbar, “Random Forest Modeling for Network Intrusion Detection System,” *Procedia Comput. Sci.*, vol. 89, pp. 213–217, 2016, doi: 10.1016/j.procs.2016.06.047.

[37] W. Li, P. Yi, Y. Wu, L. Pan, and J. Li, “A new intrusion detection system based on KNN classification algorithm in wireless sensor network,” *J. Electr. Comput. Eng.*, vol. 2014, no. 1, 2014, doi: 10.1155/2014/240217.

[38] M. Hegde, G. Kepnang, M. Al Mazroei, J. S. Chavis, and L. Watkins, “Identification of Botnet Activity in IoT Network Traffic Using Machine Learning,” *2020 Int. Conf. Intell. Data Sci. Technol. Appl. IDSTA 2020*, no. 1, pp. 21–27, 2020, doi: 10.1109/IDSTA50958.2020.9264143.

[39] S. Pokhrel, R. Abbas, and B. Aryal, “IoT Security: Botnet detection in IoT using Machine learning,” pp. 1–11, 2021, [Online]. Available: http://arxiv.org/abs/2104.02231.

[40] M. M. Rashid, J. Kamruzzaman, M. M. Hassan, T. Imam, and S. Gordon, “Cyberattacks detection in iot-based smart city applications using machine learning techniques,” *Int. J. Environ. Res. Public Health*, vol. 17, no. 24, pp. 1–21, 2020, doi: 10.3390/ijerph17249347.

[41] H. Alkahtani and T. H. H. Aldhyani, “Botnet Attack Detection by Using CNN-

LSTM Model for Internet of Things Applications," *Secur. Commun. Networks*, vol. 2021, 2021, doi: 10.1155/2021/3806459.

[42] Y. Liang and N. Vankayalapati, "Machine Learning and Deep Learning Methods for Better Anomaly Detection in IoT-23 Dataset Cybersecurity," pp. 1–6.

[43] Y. N. Soe, Y. Feng, P. I. Santosa, R. Hartanto, and K. Sakurai, "Machine learning-based IoT-botnet attack detection with sequential architecture," *Sensors (Switzerland)*, vol. 20, no. 16, pp. 1–15, 2020, doi: 10.3390/s20164372.

[44] Q. A. Al-Haija and M. Al-Dala'ien, "ELBA-IoT: An Ensemble Learning Model for Botnet Attack Detection in IoT Networks," *J. Sens. Actuator Networks*, vol. 11, no. 1, 2022, doi: 10.3390/jsan11010018.

[45] I. Ullah and Q. H. Mahmoud, "An Anomaly Detection Model for IoT Networks based on Flow and Flag Features using a Feed-Forward Neural Network," pp. 363–368, 2022, doi: 10.1109/ccnc49033.2022.9700597.

[46] H. Jiang, Z. He, G. Ye, and H. Zhang, "Network Intrusion Detection Based on PSO-Xgboost Model," *IEEE Access*, vol. 8, pp. 58392–58401, 2020, doi: 10.1109/ACCESS.2020.2982418.

[47] J. Alsamiri and K. Alsubhi, "Internet of things cyber attacks detection using machine learning," *Int. J. Adv. Comput. Sci. Appl.*, vol. 10, no. 12, pp. 627–634, 2019, doi: 10.14569/ijacsa.2019.0101280.

[48] F. Abbasi, M. Naderan, and S. E. Alavi, "Intrusion Detection in IoT With Logistic Regression and Artificial Neural Network: Further Investigations on N-BaIoT Dataset Devices," *J. Comput. Secur. Res. Artic.*, vol. 8, no. 2, pp. 27–42, 2021, [Online]. Available: http://www.jcomsec.org.

[49] P. Chauhan and M. Atulkar, "Selection of tree based ensemble classifier for detecting network attacks in IoT," *2021 Int. Conf. Emerg. Smart Comput. Informatics, ESCI 2021*, pp. 770–775, 2021, doi: 10.1109/ESCI50559.2021.9397033.

[50] A. R. Gad, A. A. Nashat, and T. M. Barkat, "Intrusion Detection System Using Machine Learning for Vehicular Ad Hoc Networks Based on ToN-IoT Dataset," *IEEE Access*, vol. 9, pp. 142206–142217, 2021, doi: 10.1109/ACCESS.2021.3120626.

[51] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network

Intrusion Detection for IoT Security Based on Learning Techniques," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 3, pp. 2671–2701, 2019, doi: 10.1109/COMST.2019.2896380.

[52] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, 2012, doi: 10.1016/j.cose.2011.12.012.

[53] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Futur. Gener. Comput. Syst.*, vol. 100, pp. 779–796, 2019, doi: 10.1016/j.future.2019.05.041.

[54] Kurniabudi, D. Stiawan, Darmawijoyo, M. Y. Bin Bin Idris, A. M. Bamhdi, and R. Budiarto, "CICIDS-2017 Dataset Feature Analysis with Information Gain for Anomaly Detection," *IEEE Access*, vol. 8, pp. 132911–132921, 2020, doi: 10.1109/ACCESS.2020.3009843.

[55] Y. Meidan *et al.*, "N-BaIoT-Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, 2018, doi: 10.1109/MPRV.2018.03367731.

[56] S. Garcia and and M. J.-E. Agustin Parmisano, Sebastian Garcia, "IoT-23: A labeled dataset with malicious and benign IoT network traffic," 2020. .

[57] N. Moustafa, "A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets," *Sustain. Cities Soc.*, vol. 72, no. May, p. 102994, 2021, doi: 10.1016/j.scs.2021.102994.

[58] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. H. D. Hartog, "ToN_IoT: The Role of Heterogeneity and the Need for Standardization of Features and Attack Types in IoT Network Intrusion Data Sets," *IEEE Internet Things J.*, vol. 9, no. 1, pp. 485–496, 2022, doi: 10.1109/JIOT.2021.3085194.

[59] H. B. Harvey and S. T. Sotardi, "The Pareto Principle," *J. Am. Coll. Radiol.*, vol. 15, no. 6, p. 931, 2018, doi: 10.1016/j.jacr.2018.02.026.

[60] M. Sarhan, S. Layeghy, N. Moustafa, M. Gallagher, and M. Portmann, "Feature Extraction for Machine Learning-based Intrusion Detection in IoT Networks," 2021, [Online]. Available: http://arxiv.org/abs/2108.12722.

[61] M. Austin, “IoT Malicious Traffic Classification Using Machine Learning,” 2021, doi: 10.33915/etd.8024.

[62] E. Jaw and X. Wang, “Feature Selection and Ensemble-Based Intrusion Detection System: An Efficient and Comprehensive Approach,” *Symmetry (Basel).*, vol. 13, no. 10, p. 1764, 2021, doi: 10.3390/sym13101764.

[63] N. A. Stoian, “Machine Learning for Anomaly Detection in IoT networks: Malware analysis on the IoT-23 Data set,” *Univ. Twente*, 2020.

[64] Y. Liang and N. Vankayalapati, “Machine Learning and Deep Learning Methods for Better Anomaly Detection in IoT-23 Dataset Cybersecurity,” pp. 1–6.

[65] C. Ioannou and V. Vassiliou, “An Intrusion Detection System for Constrained WSN and IoT Nodes Based on Binary Logistic Regression,” no. October, pp. 259–263, 2018, doi: 10.1145/3242102.3242145.

[66] M. S. Alsahli, M. M. Almasri, M. Al-Akhras, A. I. Al-Issa, and M. Alawairdhi, “Evaluation of Machine Learning Algorithms for Intrusion Detection System in WSN,” *Int. J. Adv. Comput. Sci. Appl.*, vol. 12, no. 5, pp. 617–626, 2021, doi: 10.14569/IJACSA.2021.0120574.

[67] N. Prashil, Y. Trivedi, and R. Mangrulkar, “Intrusion Detection System Using Random Forest on the NSL-KDD Dataset,” in *Emerging Research in Computing, Information, Communication and Applications*, Singapore: Springer, 2019, pp. 519–531.

[68] C. Okur and M. Dener, “Detecting IoT Botnet Attacks Using Machine Learning Methods,” *2020 Int. Conf. Inf. Secur. Cryptology, ISCTURKEY 2020 - Proc.*, pp. 31–37, 2020, doi: 10.1109/ISCTURKEY51113.2020.9307994.

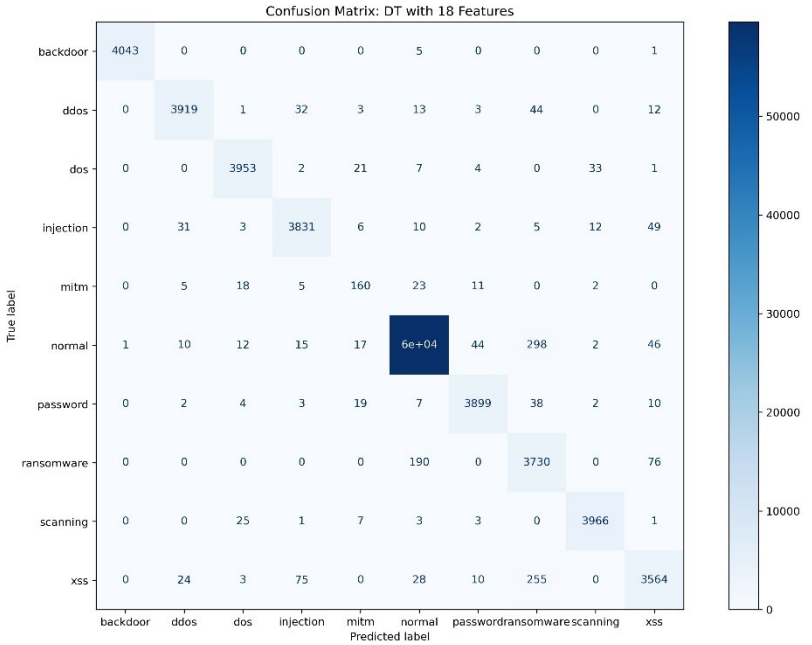
[69] G. L. Nguyen, B. Dumba, Q. D. Ngo, H. V. Le, and T. N. Nguyen, “A collaborative approach to early detection of IoT Botnet,” *Comput. Electr. Eng.*, vol. 97, no. November 2021, p. 107525, 2022, doi: 10.1016/j.compeleceng.2021.107525.

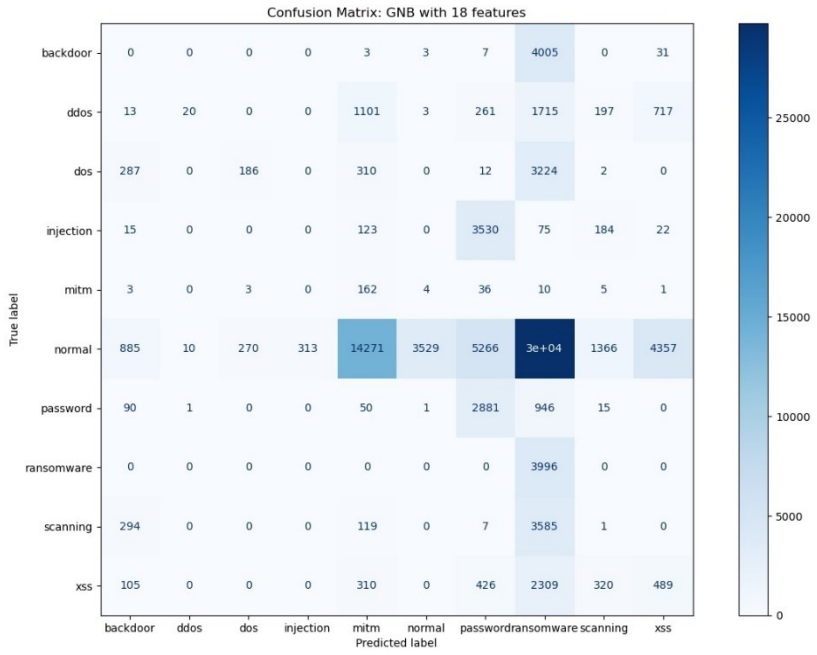
[70] Dmlc XGBoost stable, “XGBoost Python Package.” https://xgboost.readthedocs.io/en/stable/python/python_intro.html (accessed Apr. 10, 2022).

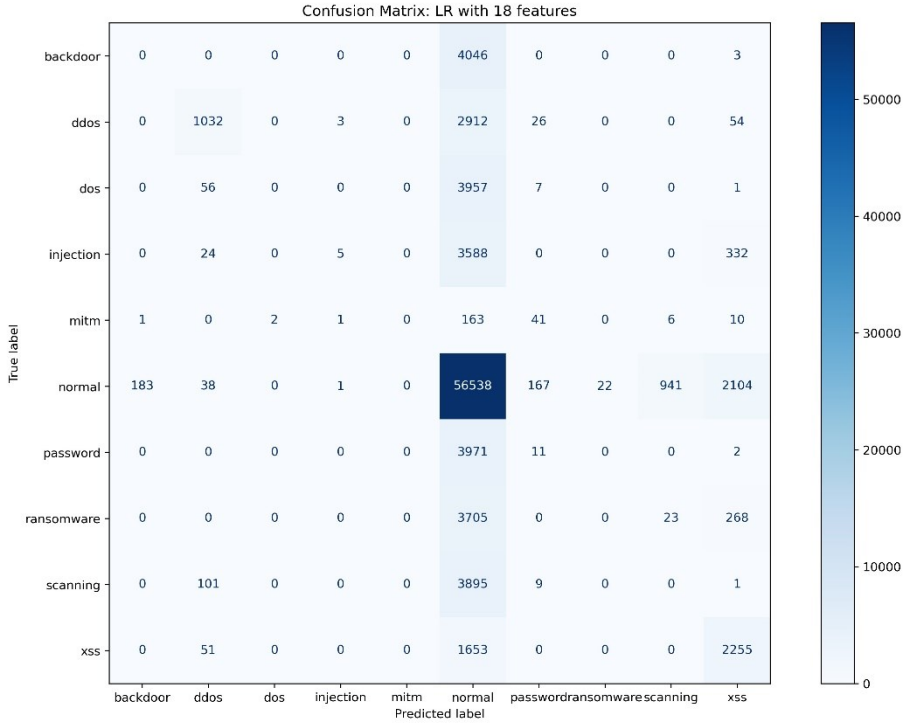
[71] I. L. Cherif and A. Kortebi, “On using eXtreme Gradient Boosting (XGBoost) Machine Learning algorithm for Home Network Traffic Classification,” *IFIP Wirel. Days*, vol. 2019-April, 2019, doi: 10.1109/WD.2019.8734193.

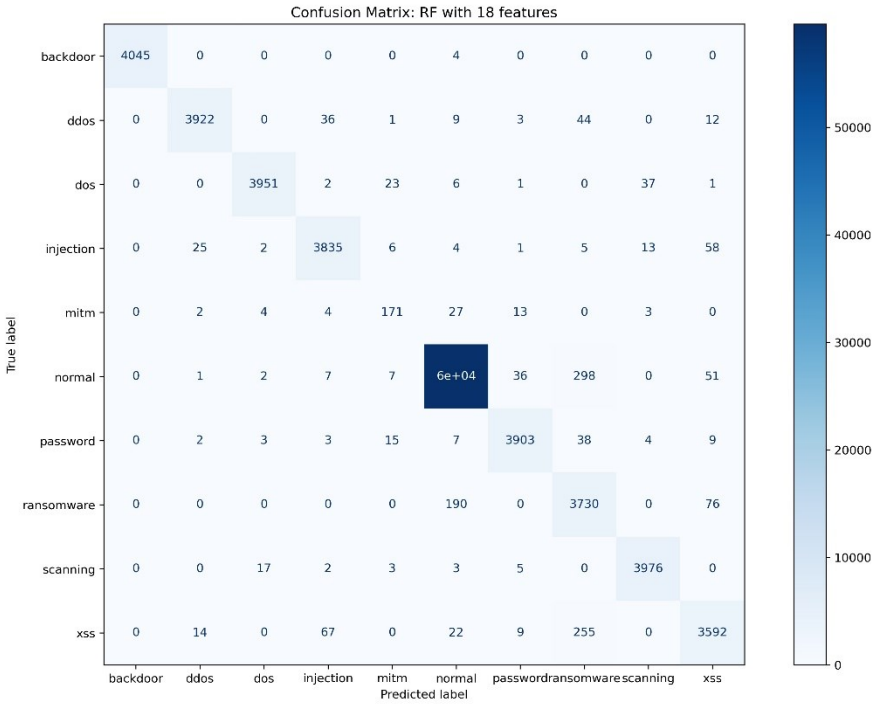
Appendix

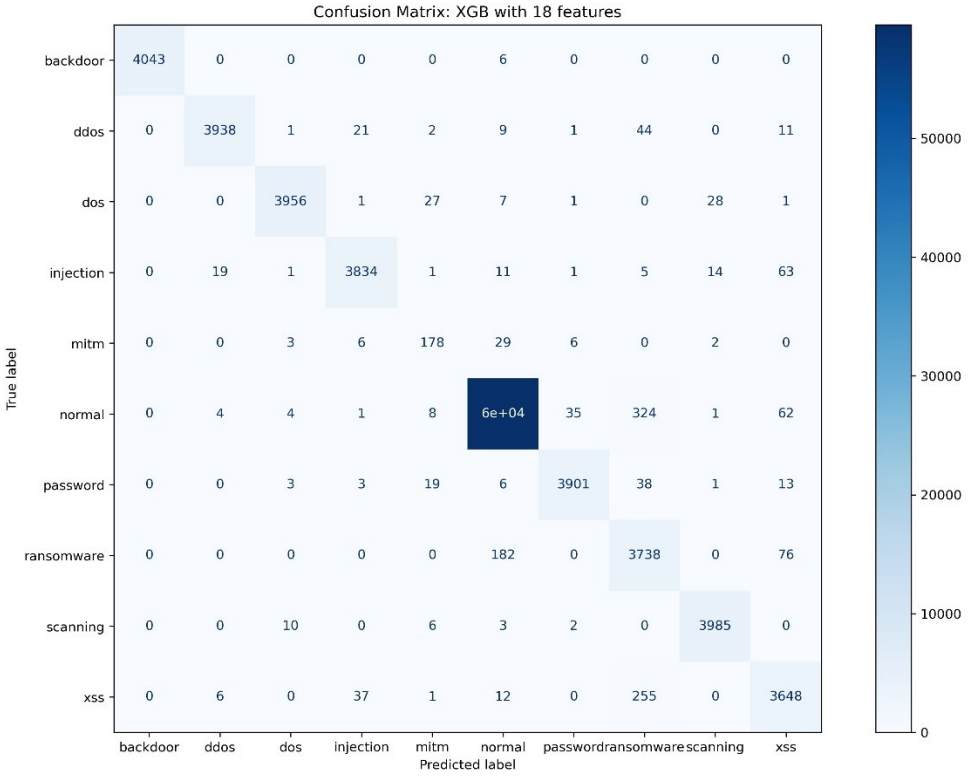
Extra Figures

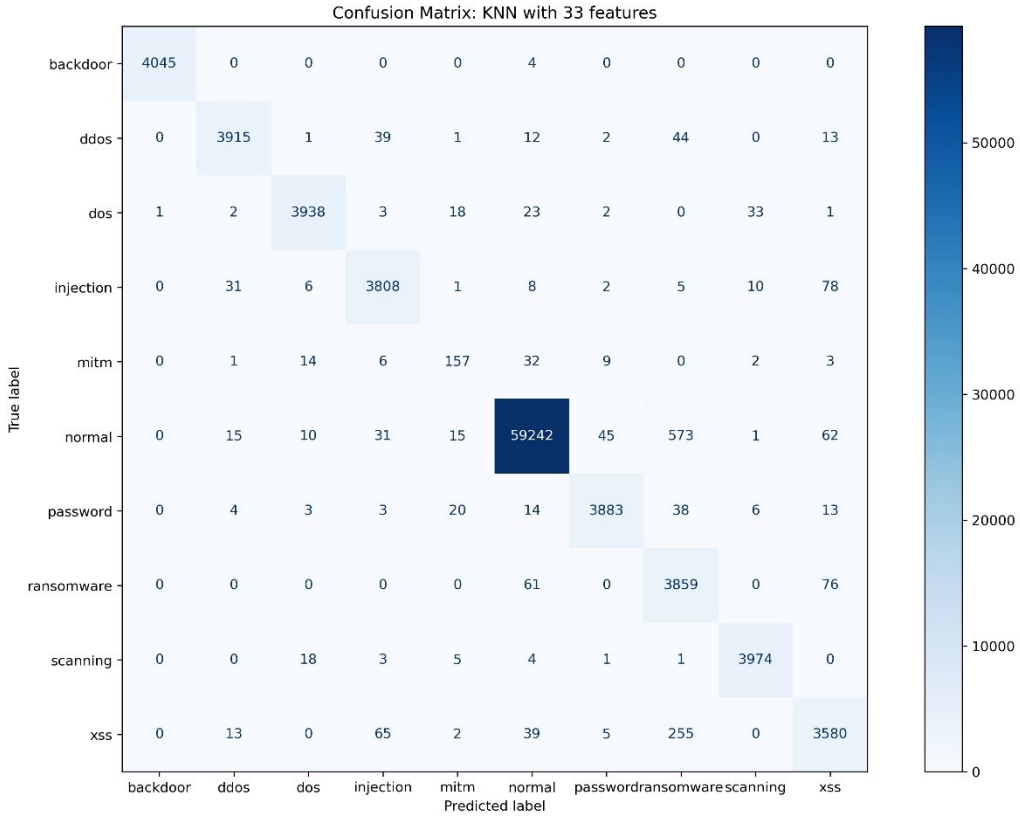





Confusion Matrix: LR with 18 features
backdoor 0 0 0 0 0 4046 0 0 0 3
ddos 0 1032 0 3 0 2912 26 0 0 54
dos 0 56 0 0 0 3957 7 0 0 1
injection 0 24 0 5 0 3588 0 0 0 332
mitm 1 0 2 1 0 163 41 0 6 10
normal 183 38 0 1 0 56538 167 22 941 2104
password 0 0 0 0 0 3971 11 0 0 2
ransomware 0 0 0 0 0 3705 0 0 23 268
scanning 0 101 0 0 0 3895 9 0 0 1
xss 0 51 0 0 0 1653 0 0 0 2255
True label
backdoor ddos dos injection mitm normal password ransomware scanning xss
Predicted label
50000
40000
30000
20000
10000
0


Confusion Matrix: RF with 18 features
backdoor 4045 0 0 0 0 4 0 0 0 0
ddos 0 3922 0 36 1 9 3 44 0 12
dos 0 0 3951 2 23 6 1 0 37 1
injection 0 25 2 3835 6 4 1 5 13 58
mitm 0 2 4 4 171 27 13 0 3 0
normal 0 1 2 7 7 6e+04 36 298 0 51
password 0 2 3 3 15 7 3903 38 4 9
ransomware 0 0 0 0 0 190 0 3730 0 76
scanning 0 0 17 2 3 3 5 0 3976 0
xss 0 14 0 67 0 22 9 255 0 3592
True label
backdoor ddos dos injection mitm normal password ransomware scanning xss
Predicted label
50000
40000
30000
20000
10000
0


Confusion Matrix: XGB with 18 features
backdoor 4043 0 0 0 0 6 0 0 0 0
ddos 0 3938 1 21 2 9 1 44 0 11
dos 0 0 3956 1 27 7 1 0 28 1
injection 0 19 1 3834 1 11 1 5 14 63
mitm 0 0 3 6 178 29 6 0 2 0
normal 0 4 4 1 8 6e+04 35 324 1 62
password 0 0 3 3 19 6 3901 38 1 13
ransomware 0 0 0 0 0 182 0 3738 0 76
scanning 0 0 10 0 6 3 2 0 3985 0
xss 0 6 0 37 1 12 0 255 0 3648
backdoor ddos dos injection mitm normal password ransomware scanning xss
True label
Predicted label
50000
40000
30000
20000
10000
0


Confusion Matrix: KNN with 33 features
backdoor 4045 0 0 0 0 4 0 0 0 0
ddos 0 3915 1 39 1 12 2 44 0 13
dos 1 2 3938 3 18 23 2 0 33 1
injection 0 31 6 3808 1 8 2 5 10 78
mitm 0 1 14 6 157 32 9 0 2 3
normal 0 15 10 31 15 59242 45 573 1 62
password 0 4 3 3 20 14 3883 38 6 13
ransomware 0 0 0 0 0 61 0 3859 0 76
scanning 0 0 18 3 5 4 1 1 3974 0
xss 0 13 0 65 2 39 5 255 0 3580
backdoor ddos dos injection mitm normal password ransomware scanning xss
True label
Predicted label
50000
40000
30000
20000
10000
0

Source Code:

The source code can be accessed here: https://github.com/aqeel2022/IoT-Network-Malware-classification-2022