ISSN: 2723-9535

Available online at www.HighTechJournal.org

HighTech and Innovation Journal

Vol. 5, No. 3, September, 2024

IoT Attacks Detection Using Supervised Machine Learning Techniques

Malak Aljabri [1], Afrah Shaahid [2*], Fatima Alnasser [2], Asalah Saleh [2], Dorieh Alomari [2], Menna Aboulnour [2], Walla Al-Eidarous [1], Areej Althubaity [3]

[1] *Department of Computer and Network Engineering, College of Computing, Umm Al-Qura University, Makkah 21955, Saudi Arabia.*

[2] *College of Computer Science and Information Technology, Imam Abdulrahman Bin Faisal University, P.O. Box 1982, Dammam 31441, Saudi Arabia.*

[3] *Depatment of Cybersecurity, College of Computing, Umm Al-Qura University, Makkah 21955, Saudi Arabia.*

Received 29 March 2024; Revised 29 July 2024; Accepted 08 August 2024; Published 01 September 2024

Abstract

In recent times, the growing significance of Internet of Things (IoT) devices in people's lives is undeniable, driven by their myriad benefits. However, these devices confront cybersecurity threats akin to traditional network devices, as they depend on networks for connectivity and synchronization. Artificial Intelligence (AI) techniques, specifically Machine Learning (ML) and Deep Learning (DL), have demonstrated notable reliability in the field of cyberattack detection. This study focuses on detecting Flood and Brute Force cyberattacks using Machine Learning (ML) and Deep Learning (DL) models. The primary emphasis lies in identifying traffic features that significantly detect these types of attacks. The experimental study incorporates eight models: Decision Tree (DT), K-Nearest Neighbor (KNN), Random Forest (RF), Support Vector Machines (SVM), Logistic Regression (LR), Gradient Boosting (GB), Naïve Bayes (NB), and Artificial Neural Network (ANN). Two sets of experiments were conducted, with the first set involving six features and the subsequent set, after feature selection, focusing on a reduced set of three features. The evaluation of the proposed model's efficiency and performance relied on metrics such as Accuracy, Precision, Recall, and F1-score. Remarkably, all proposed models exhibited high performance in both sets of experiments. However, the Gradient Boosting (GB) classifier suppressed others, attaining an impressive accuracy level of 95.94% and 95.28% in the sets with six features and three features, respectively.

Keywords: Supervised; IoT Security; Cyberattacks; IoT Attacks.

1. Introduction

In the contemporary era, the internet has become a fundamental aspect of our daily existence. Attempts to breach computer systems and networks have escalated due to the surge in online applications that evolved with the advent of transformative technologies like the Internet of Things (IoT). IoT, seamlessly integrating intelligent objects and devices, has experienced exponential growth, projecting a global connection of 15.1 billion devices in 2023 [1]. The range of IoT applications extends from wearables for health monitoring and smart fridges in home appliances to intelligent boards for education [2]. Nonetheless, IoT confronts a range of cyber threats in the internet's hostile environment, emphasizing the ongoing need for efforts to support network security. Machine Learning (ML) emerges as a highly successful computational model for embedding Artificial Intelligence (AI) in the IoT domain. ML methods in cybersecurity have been instrumental in various network security advancements [3], including network traffic analysis [4-6], intrusion detection [7], and botnet identification [8-10].

* Corresponding author: 2190009057@iau.edu.sa

http://dx.doi.org/10.28991/HIJ-2024-05-03-01

ML plays a crucial role in IoT solutions with its unique ability to automate or adapt knowledge-based behaviors. It can unearth valuable insights from data generated by humans or machines. One of its key applications is providing security services in IoT networks, particularly in the progressive research on attack detection strategies [11]. The threats posed by two prevalent cyberattacks, Flood Denial of Service (DoS) attacks and Real-Time Streaming Protocol (RTSP) brute force attacks, emphasize the significance of this role. These attacks, if not detected and mitigated, can cause severe disruptions, potentially leading to system crashes and rendering the system unreachable for its designated users.

The limited computational power, short battery life, and poor built-in security of IoT devices make them easy targets for attackers to launch DDoS and DoS attacks. Attackers often exploit vulnerabilities in IoT devices using brute-force techniques to compromise credentials and gain access to these devices. Once compromised, these devices can be turned into malicious bots and assembled into large botnets controlled by attackers [12]. Malicious bots could then establish DoS attacks or brute-force attacks.

DoS attacks employ two primary approaches—flooding services or crashing services—where flood attacks occur when a server cannot manage the incoming traffic, resulting in system slowdown or cessation. Common flood attack types encompass buffer overflow attacks, ICMP (Internet Control Message Protocol) floods, and SYN floods. In contrast, brute-force attacks, a hacking technique reliant on trial and error, target encryption keys, passwords, and login credentials to gain unauthorized access. Despite its simplicity, brute-force attacks remain popular among hackers, utilizing computers to test various username-password combinations until they discover the correct login information. This intriguing simplicity and popularity of brute-force attacks highlight the need for advanced security measures.

ML plays a prominent role in attack detection through two cyber-analysis types: signature-based and anomaly-based. Signature-based approaches utilize specific traffic characteristics or "signatures" to accurately identify known attacks without generating excessive false alarms. While effective, these approaches have limitations, such as the inability to detect previously undetected attacks and the need for regular manual updates to attack traffic signatures. On the other hand, Anomaly-based detection identifies anything deviating from usual network behavior as a potential attack, posing the risk of high False Alarm Rates (FARs). FARs introduce the possibility of categorizing formerly unrecognized yet legal behaviors as anomalies. Notably, a hybrid strategy combining signature and anomaly detection methods holds promise.

To commence, Anwer et al. [13] presented a robust framework for detecting malicious network traffic, demonstrating its reliability using the NSL-KDD dataset. The framework, powered by three popular algorithms, namely, Random Forest (RF), Support Vector Machines (SVM), and Gradient Boosted Decision Trees (GBDT), reliably analyzes traffic data to expose data that is maliciously traveling over IoT devices. The RF classifier attained an optimum accuracy of 85.34% and a specificity of 95.09%. Utilizing the same NSL-KDD dataset, Tomer & Sharma [14] introduced an innovative ensemble ML model for real-time attack detection on fog nodes. Their top-performing model, featuring base classifiers such as K-Nearest Neighbors (KNN), Naive Bayes (NB), and Decision Trees (DT), and employing an ensemble approach through voting, achieved exceptional results with the NSL-KDD dataset, yielding 99.4% precision, 99.7% recall, 99.5% F1-score, and a ROC of 99.9%.

Alsamiri & Alsubhi [15] demonstrated the impressive speed and efficiency of their evaluation of seven ML models, including Iterative Dichotomiser 3 (ID3), KNN, RF, Quadratic Discriminant Analysis (QDA), AdaBoost, Multilayer Perceptron (MLP), and NB using Bot-IoT dataset, which covers a wide variety of botnet attacks. Moreover, 84 new network traffic features were extracted using CICFlowMeter [16]. The KNN model, which had accuracy, recall, precision, and an F1-score of 99%, was the best performer. Likewise, Htwe et al. [17] suggested a detection framework employing ML techniques, including the Classification and Regression Trees (CART) algorithm, on the N-BaIoT dataset. The N-BaIoT dataset is a comprehensive IoT intrusion detection dataset containing many network traffic features. Comparative analysis with the NB classifier showed that CART achieved significantly better results, averaging a detection accuracy of 99%.

Gaber et al. [18] proposed an efficient intrusion detection method, employing recursive feature elimination and constant removal for feature selection to counter injection attacks in IoT devices. They evaluated multiple ML algorithms (RF, SVM, and DT) on the Aegean Wi-Fi Intrusion Dataset (AWID), a popular dataset used for research in wireless network security, particularly Intrusion Detection Systems (IDS). The results revealed that DT emerged as the most potent, achieving outstanding results with 99% accuracy, 95% precision, and a 90% F1-score, all accomplished with a concise set of 8 features. To identify common DDOS attacks such as *BASHLITE and Mirai*, Aysa et al. [19] constructed a framework to detect abnormal defense activities, focusing on IoT-specific features. The training and testing of LSVM, Neural Network, J48, and RF ML models determined that the optimal outcome was achieved by combining RF and DT, achieving an outstanding precision, recall, and F1-score of 99.7%.

Notably, Krishnan et al. [20] conducted a significant study where they built three classifiers to predict whether the traffic is malicious or benign. Their research, which involved several supervised feature selection methods on IoT network data, was instrumental in determining the optimal feature selection approach for network intrusion prediction.

The models they used, SVM, RF, and Extreme Gradient Boosting (XGBoost), were rigorously compared, and the analysis concluded that using recursive feature elimination for feature selection was the best choice. The model XGBoost performed exceptionally well, achieving a perfect F1-score of 100%, a recall rate of 99.79, and an accuracy level matching the recall at 99.79. Similarly, Saran & Kesswani [21] identified multi-class intrusion attacks in IoT environments, evaluating the performance of various ML classifiers based on accuracy, precision, recall, and F1-Score. The MQTT-IoT-IDS2020 dataset demonstrated the effectiveness of classifiers such as RF, DT, k-NN, SVM, NB, and Stochastic Gradient Descent (SGD), with RF and DT achieving a high accuracy of 99.98%.

Hammood & Sadiq [5] recently discussed ensemble ML methods for IDS in IoT environments. They utilized three publicly available datasets: UNSW-NB15, IoTID-20, and BotNetIoT, which contain labeled network traffic data categorized as normal or malicious, including specific attack types, such as DoS attacks. The suggested method utilized six ML algorithms: Logistic Regression (LR), NB, DT, Extra Trees, RF, and GBoost. The predictions of all six ML algorithms were combined using an ensemble approach. Performance metrics: accuracy, precision, recall, and F1 score were conducted to evaluate the models. The ensemble method achieved an accuracy of 88.41% on IoTID20, 98.52% on UNSW-NB15, and 91.03% on BoTNeTIoT, outperforming individual ML algorithms. The experiment's results highlight the effectiveness of the ensemble methods for improving intrusion detection accuracy in IoT networks.

Furthermore, Altulaihan et al. [22] utilized the IoTID20 dataset to showcase the effectiveness of ML classifiers like DT, RF, KNN, and SVM in detecting IoT cyberattacks, particularly DoS attacks. The research employed feature selection algorithms such as Correlation-based Feature Selection (CFS) and Genetic Algorithm (GA) to optimize the performance of these classifiers. While the DT and RF classifiers achieved a superior performance of 100% accuracy when trained with GA under specific conditions, the SVM model with GA features showed a lower accuracy of 88.29%. This discrepancy was attributed to the inherent difficulty faced by SVMs in handling large datasets with strong feature correlations, a challenge that has yet to be encountered by DT and RF classifiers.

Several studies have explored the use of DL for attack detection. For instance, Pecori et al. [23] evaluated the effectiveness of DL models of different numbers of hidden layers compared to traditional ML approaches. They curated an extensive dataset of IoT traffic flows for model assessment, employing Hoeffding Tree (HT), DT, and self-constructed DL models with layers ranging from four to seven. The DL architecture outperformed the ML models in both binary and multinomial classification. The DL with seven hidden layers achieved outstanding results, recording an accuracy of 99.75%, a precision of 99.37%, a recall of 99.37%, and an F1-score of 99.37% for binary classification. Simultaneously, the DL with seven hidden layers produced the best results. In the meantime, the DL with six hidden layers yielded optimal results for multiclass classification, securing a 99.73% accuracy, 98.86% precision, 98.67% recall, and a 98.77% F1-score.

Moreover, Alkahtani & Aldhyani [24] focused on botnet attacks on nine commercial IoT devices. They employed a hybrid DL approach denoted as (CNNLSTM), which combins the Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) algorithms. The study utilized the N-BaIoT dataset, which featured benign and malicious patterns obtained from a real system. The CNN-LSTM model achieved accuracies ranging from 87.19% to 90.88% across various IoT devices. Following a more hierarchical approach, Al-zubidi et al. [25] proposed a new intrusion detection system named CNN-LSTM-XGBoost. The study highlighted that the traditional IDSs for DoS and DDoS attacks cannot detect new attacks, resulting in a low accuracy rate. The proposed system combines CNNs, LSTM networks, and XGBoost to achieve high accuracy in attack detection. CNNs and LSTMs extract features from raw network traffic data, identifying temporal and spatial patterns. These features were then fed into XGBoost, a fast and efficient classifier, to categorize the traffic as normal or containing an attack. CNN-LSTM-XGBoost was evaluated on three publicly available datasets, namely CICIDS-001, CIC-IDS2017, and CIC-IDS2018. All three datasets focus on IDS network traffic data and contain normal and malicious traffic with various attack scenarios, including DoS and DDoS attacks. The system achieved over 98% accuracy on all datasets, demonstrating its effectiveness compared to existing methods.

Conversely, Islam et al. [26] aimed to identify IoT threats through IDS. They utilized both ML algorithms (SVM, RF, and DT), as well as DL algorithms (Deep Neural Network (DNN), Deep Belief Network (DBN), LSTM, stacked LSTM, and Bi-LSTM). Five benchmark datasets evaluated the models: IoTDevNet, NSLKDD, DS2OS, IoT Botnet, and IoTID20. The DL approach outperformed ML algorithms, with Bi-LSTM demonstrating the best performance, achieving testing accuracies of 99.27%, 99.97%, 99.39%, 99.99%, and 99.991% on the respective datasets.

Lastly, Karamollaoğlu et al. [27] investigated the use of the CNN for attack classification in IoT networks. The study discussed a hybrid approach combining Principal Component Analysis (PCA) and Bat Optimization Algorithm for dimensionality reduction to improve efficiency on resource constrained IoT devices. The system was evaluated on two datasets, IoTID20 and BoT-IoT, which contain various attack types, including DoS, DDoS, and botnet attacks. The model's performance was evaluated using accuracy, precision, recall, and F1-score. The proposed model has achieved 99.9% accuracy on both datasets.

Table 1 summarizes the reviewed studies, encompassing methods, datasets, attack types detected, and achieved accuracy and F1-scores.

Table 1. Summary of the Literature Review

Reference	Method Used	Dataset	Attack Type	Accuracy	F1-score
Anwer et al. (2021) [13]	RF	NSL-KDD	Probe, DoS, U2R, and R2L	85.34%	-
Tomer & Sharma (2022) [14]	KNN, NB, and DT	NSL-KDD	Probe, DoS, U2R, and R2L	-	99.5%
Alsamiri & Alsubhi (2019) [15]	KNN	Bot-IoT	Probing, DoS, and Information Theft.	99%	99%
Htwe et al. (2020) [17]	CART	N-BaIoT	Ack, scan, syn, udp, udpplain, tcp, junk, and comb attacks.	99%	-
Gaber et al. (2022) [18]	DT	AWID	Injection attacks	99%	90%
Aysa et al. (2020) [19]	RF and DT	A standard dataset containing common attacks	Mirai and BASHLITE	-	99.7%
Krishnan et al. (2021) [20]	XGBoost	Private IoT network data	DoS and spoofing	99.79%	100%
Saran & Kesswani, (2023) [21]	NB, RF, DT, SVM, k-NN, and SGD	MQTT-IoT-IDS2020	intrusion attacks	99.98%	99.98%
Hammood & Sadiq (2023) [5]	Ensemble ML	UNSW-NB15, IoTID-20, BotNetIoT	DoS	88.41% on IoTID20, 98.52% on UNSW-NB15, and 91.03% on BoTNeTIoT	-
Altulaihan et al. (2024) [22]	DT, RF, KNN, SVM	IoTID20	DoS	100%	100%
Pecori et al. (2020) [23]	DL with six hidden layers	-	Scanning, DoS, Mirai, MITM	99.73%	98.77%
Alkahtani & Aldhyani (2021) [24]	CNN-LSTM	N-BaIoT dataset	BASHLITE and Mirai	90.88%	-
Al-zubidi et al. (2024) [25]	CCN-LSTM-XGBoost	CICIDS-001, CIC-IDS2017, CIC-IDS2018	DoS, DDoS	98%	99%
Islam et al. (2021) [26]	Bi-LSTM	NSLKDD, IoTDevNet, DS2OS, IoTID20, IoT Botnet	Scan, MITM, DoS, Prob, U2R, and R2L	99.27% 99.97% 99.39% 99.99% 99.991%	-
Karamollaoğlu et al. (2024) [27]	PCA, BAT, SMOTE, CNN	IoTID-20, BotNetIoT	DoS, DDoS, botnet	99.97%	-

After a thorough literature review, several recurring themes have emerged. Researchers often utilize various datasets to explore IoT traffic, identify malicious network activity, and implement intrusion detection systems (IDS). Each dataset offers unique insights into different attack types, enriching our understanding of IoT security. Despite extensive research, exploration of IoT attacks using newer datasets like the CIC IoT Dataset 2022 remains limited, highlighting other directions to investigate. Moreover, addressing cyber threats such as Flood DoS and RTSP brute-force attacks remains challenging.

In this study, we contribute to the existing literature by targeting IoT attack scenarios and assessing the effectiveness of ML techniques in detecting IoT network attacks, with a specific focus on flood DoS attacks and RTSP brute-force attacks. Notably, we leverage the CIC IoT Dataset 2022, a recent and comprehensive multi-dimensional profiling dataset that adds an additional perspective to this field [28].

The key contributions of this paper include:

- Improving the detection of network attacks in IoT by thoroughly evaluating the efficacy of ML and DL algorithms on a very recent dataset.
- Extraction of two distinct types of feature sets aimed at enhancing the overall model performance.
- A noteworthy contribution to the IoT and Cybersecurity literature, particularly given the limited number of studies utilizing recent datasets.

This paper is structured as follows: Section 2 presents the Methodology used for the paper, including the dataset description, data pre-processing, feature selection, ML and DL algorithms, evaluation metrics, and experimental setup. Section 3 elaborates on the study's results and discussion. This study's conclusion is summarized in Section 4, providing a comprehensive overview of our findings.

2. Research Methodology

In this study, we analyzed the network traffic of IoT devices using ML and DL techniques. We focus on detecting Flood DoS and RSTP brute-force attacks based on normal traffic patterns. We utilized a variety of classifiers, including ANN, DT, GB, KNN, LR, NB, SVM, and RF, for multi-class classification on our dataset. Each IoT device connection is classified as Flood, Brute-force, or Normal. To evaluate the performance of these models, we used key metrics such as accuracy, precision, recall, and F1-score. The 80:20 ratio signifies how the dataset was partitioned, allocating 80% for training and 20% for testing. Additionally, we conducted two experiments using different feature sets to assess their significance. The methodology adopted for this study is illustrated in Figure 1, providing a clear and structured overview of our research approach.

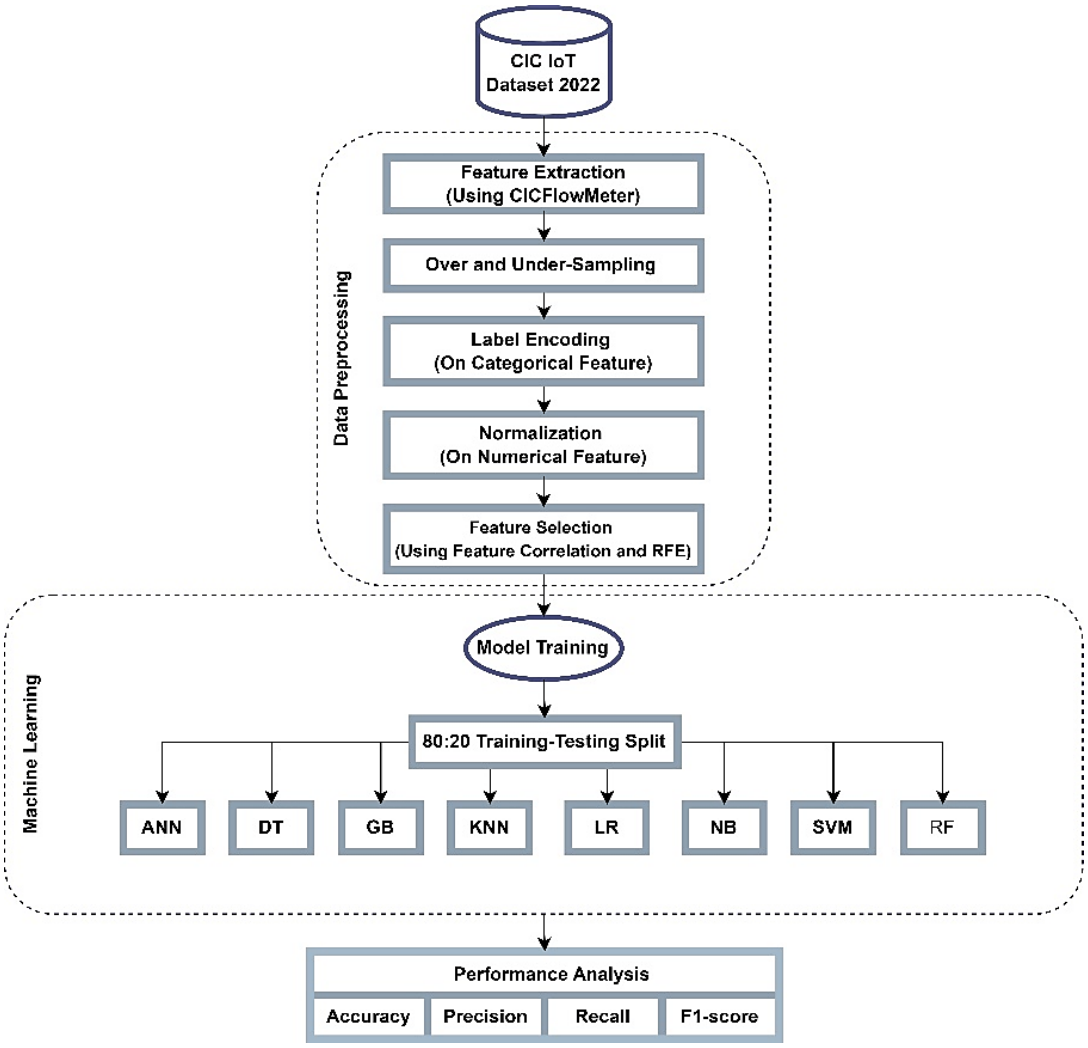


Figure 1. Research Methodology

2.1. Dataset Description

The CIC IoT Dataset 2022 is a publicly available dataset curated by the esteemed Canadian Institution for Cybersecurity (CIC). Dadkhah et al. [28] meticulously constructed this advanced dataset, incorporating 60 distinct IoT devices, for comprehensive vulnerability testing, behavioral analysis, and profiling purposes. The dataset is structured around six distinct experiments, each capturing network packets via Wireshark across various operational states, including Power, Idle, Interactions, Scenarios, Active, and Attacks.

Our investigation contributed by focusing on two specific experiments concerning the SimCam device: the Power and Attack experiments. The Power experiment served as a baseline for normal traffic analysis, while the Attacks experiment enabled the examination and classification of Flood DoS and RSTP Brute-force attacks, providing insights into the device's security vulnerabilities. Initially, the dataset provided packet captures in PCAP (Packet Capture) format, containing essential packet attributes such as protocol name, timestamp, source and destination addresses, and supporting information. To ensure a comprehensive analysis and extract additional network traffic features, we conducted a series of preprocessing steps to ensure the depth and accuracy of our findings. Table 2 outlines the original distribution of packets within the dataset.

Table 2. Original Distribution of the Dataset's Packets

Class	No. of Packets
Flood DoS	885,813
RSTP Brute-force	103,855
Normal	675

2.2. Dataset Pre-processing

Before training and testing the models of interest, pre-processing was performed to convert the dataset's raw data into a usable and effective format. Usually, pre-processing activities include loading, cleaning, manipulating, and converting data to the appropriate form for the desired study. Hence, the PCAP files of the Power and Attack experiments of the SimCam device were converted into CSV files using a network traffic flow generator and analyzer known as CICFlowMeter [16]. After conversion, each row in the CSV file represents a connection flow from start to end, and the tool extracted over 80 features. The CSV files were then labeled manually. Lastly, the dataset's size resulted in 13,315 records. Table 3 displays the dataset distribution obtained after pre-processing.

Table 3. The Dataset Distribution After Pre-processing

Class	No. of Packets
Flood DoS	10,379
RSTP Brute-force	2,784
Normal	152

Figure 2 depicts that the dataset is imbalanced as only 1.1% of the dataset composes normal traffic, and Brute-force records represent only 20.9% of attacks. Therefore, over and under-sampling pre-processing techniques were adopted to resolve the imbalance of the dataset. Under-sampling was performed on the RSTP Brute-Force class, and the records were randomly selected and reduced to 3,000. This prevented the model from being biased towards the majority class. On the other hand, both Flood DoS and Normal classes were oversampled using the Synthetic Minority Over-sampling Technique (SMOTE). SMOTE works by creating synthetic samples from the minority class rather than by over-sampling with replacement, as in the traditional approach. This was done to increase the representation of the minority classes, thereby improving the model's ability to detect them. Moreover, the target class was represented categorically. Thus, label encoding was performed on the target class: Flood DoS (1), RSTP Brute-force (0), and Normal (2). The numerical data was normalized to within the range of 0:1 utilizing the Min-Max method.

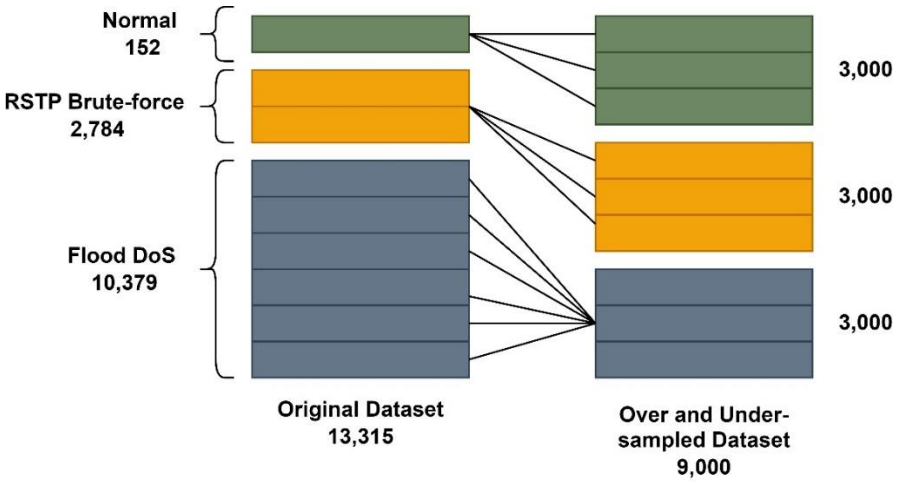


Figure 2. Sampling Techniques Applied

2.3. Feature Selection

After pre-processing and converting the dataset to CSV format using the CICFlowMeter [16] tool, over 80 statistical features were initially extracted. To refine these features, a comprehensive feature selection process was employed, incorporating both Manual Feature Correlation Elimination and Recursive Feature Elimination (RFE). This process involved removing features with all values as zero, features with uniform values, and features with zero correlation to the target class. Such features were excluded because they do not contribute to the model's detection capabilities, as they fail to reveal any distinguishable behavior relevant to the target class. Subsequently, two distinct experiments were conducted to evaluate the impact of different feature sets on the model's performance and to identify the most effective features for the detection process. Table 4 presents the selected features, their descriptions, and their correlations to the target class.

Table 4. Features Selected and Their Description

Features Selected	Description	Correlation
Protocol	It denotes protocol	0.162641
Flow IAT Max	The maximum time interval separating two consecutively transmitted packets within the flow.	0.242171
Fwd Pkts/s	The rate of forward packets per second	0.166998
Bwd Pkts/s	The rate of backward packets per second	-0.309762
Pkt Len Max	The maximum packet length	0.877138
Fwd Act Data Pkts	The count of packets in the forward direction, each containing a minimum of 1 byte of TCP data payload.	0.384332

In the first experiment, six features were selected by eliminating those with high inter-feature correlation, thereby retaining only those features with a strong individual impact on the target class. This approach avoids redundant information and reduces algorithmic complexity.

Meanwhile, a further feature selection technique, RFE, was applied in the second experiment, and only three features from Experiment 1 were used. Table 5 illustrates the features used in each experiment. As the results depict, the feature with the highest correlation to the target class, the Pkt Len Max, stands out as integral for analyzing and identifying the attacks performed against IoT devices. The significance of these findings is substantial, as they demonstrate the efficacy of careful feature selection in enhancing the detection performance of models, particularly in the context of IoT security. The study provides a robust foundation for developing more efficient and accurate detection mechanisms by prioritizing features that offer the most distinctive insights into attack patterns.

Table 5. Set of Features Used in Each Experiment

Experiment 1: Features Selected with Manual Correlation Elimination	Experiment 2: Features in Experiment 1 After Conducting RFE
Protocol	
Flow IAT Max	
Fwd Pkts/s	Fwd Pkts/s
Bwd Pkts/s	Bwd Pkts/s
Pkt Len Max	Pkt Len Max
Fwd Act Data Pkts	

2.4. ML and DL Algorithms

In this section, the features selected are fed into the classifiers and well-known ML and DL algorithms that were applied in this study are analyzed and investigated. The algorithms employed in this study include ANN, DT, GB, KNN, LR, NB, SVM, and RF. Furthermore, the evaluation of each algorithm's performance included analyzing vital metrics such as Accuracy, Precision, Recall, and F1-score.

2.4.1. Artificial Neural Network

A renowned model in the field of perception is the Artificial Neural Network (ANN), modeled after the human brain structure. Many AI scientists believe that understanding how the human brain functions can help in defining solutions for computational problems such as formal algorithms and implementing them. The brain is composed of processing units called neurons that are largely connected through synapses. Correspondingly, an ANN model is a mathematical representation that functions like the human brain. The network neurons are arranged into input, hidden, and output layers, as depicted in Figure 3. This model is renowned as a Feed-Forward Neural Network (FFNN). ANN is non-parametric and finds application in classification and regression problems [29, 30].

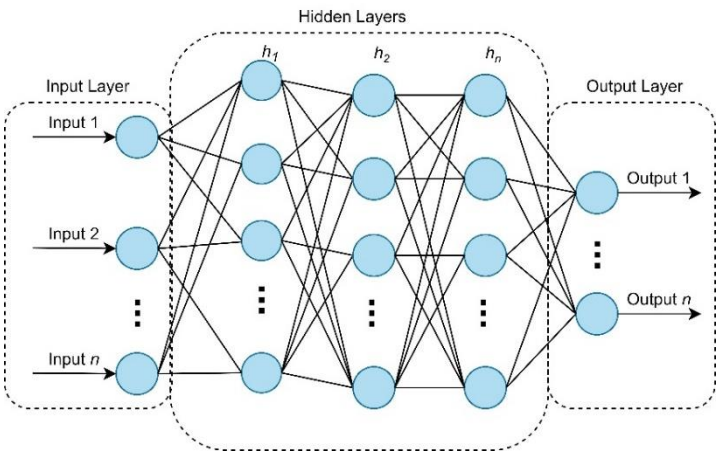


Figure 3. The Layers of Neurons in ANN

2.4.2. Decision Tree

A decision tree (DT), a hierarchical data structure that employs the divide-and-conquer approach, serves as a nonparametric method adeptly applied to both classification and regression problems. A DT is a model for supervised learning where a sequence of recursive splits to identify the desired local region with the fewest steps. DT are composed of internal nodes and terminal leaves. A test function donated as fm(x) is applied at each node m in a DT. The branches are labeled with discrete outcomes, and for a given input, a test is performed at each node. Subsequently, one of the branches is selected based on the results obtained. The process is recursively repeated starting from the root until a leaf node is achieved which outcomes in an output value. Figure 4 shows an abstract view of the Decision Tree process. Moreover, the non-parametric nature of decision trees causes them to develop branches and leaves as they learn, and this growth is contingent upon the complexity of the problem being addressed [29].

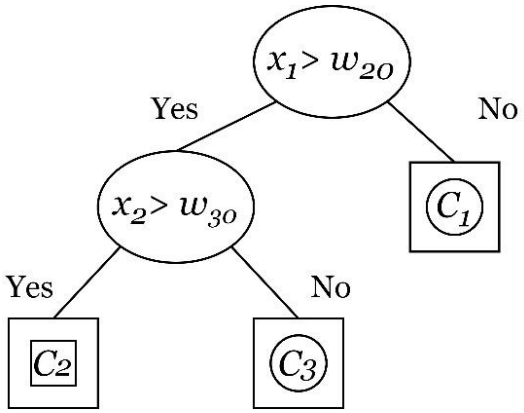


Figure 4. Decision Tree Process

2.4.3. Gradient Boosting

A Gradient Boosting, classified within the ensemble machine learning category, employs a suite of algorithms to merge multiple weak learning models (predictors with limited accuracy), forming a robust predictive model characterized by a high accuracy rate. The Gradient Boosting model depends on a loss function aimed at minimizing errors. In each iteration, the model attempts to enhance the accuracy by reducing the errors fed into it by its predecessor. Hence, during each iteration, a new model is developed by incorporating the residual errors from the previous model, as opposed to fitting an entirely new model [31] as depicted in Figure 5.

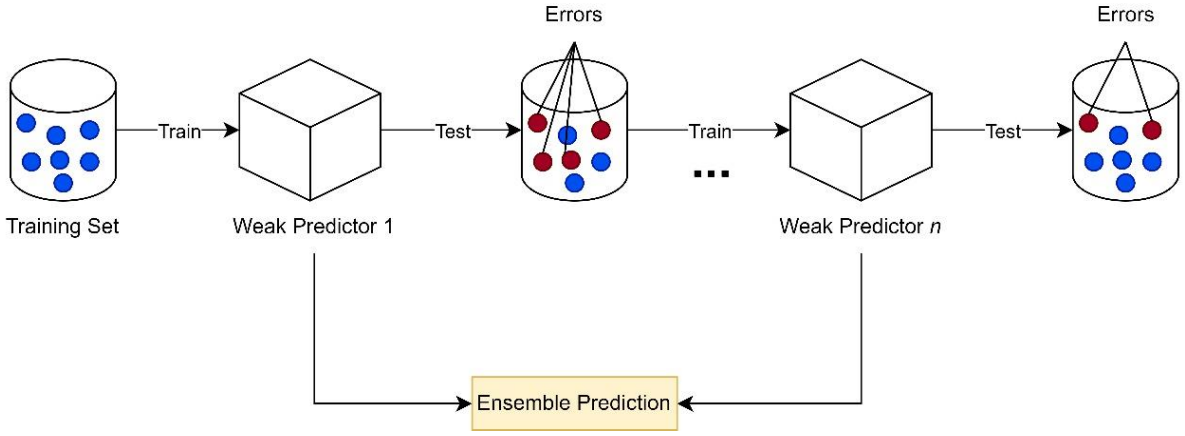


Figure 5. The Gradient Boosting Procedure

2.4.4. K-Nearest Neighbor

KNN algorithm is a straightforward ML technique utilized for classification and regression. The construction of a KNN model involves retaining the training dataset. When predicting a new data point, the algorithm identifies its nearest neighbor or neighbors in the training dataset. Initially, the algorithm typically starts by finding only the single nearest neighbor. However, we can consider a random number of neighbors, as the name KNN indicates. That being done, the algorithm uses voting to make a prediction. Hence, for a test point, the number of neighbors is counted and then assigned to the majority class among the KNN [32] as shown in Figure 6.

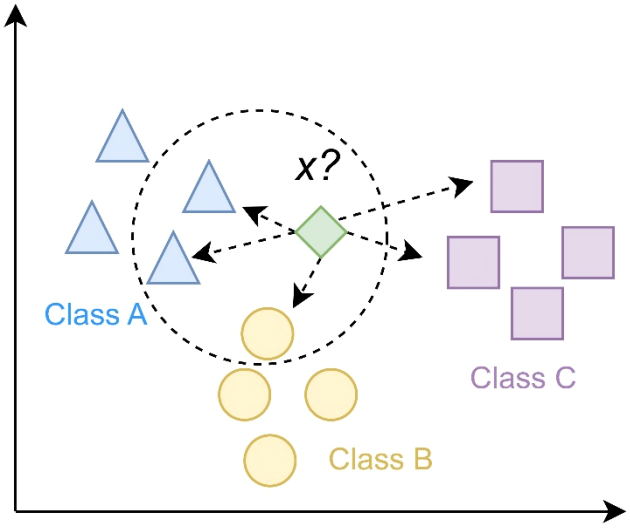


Figure 6. KNN Procedure for (k = 3)

2.4.5. Logistic Regression

Logistic Regression is the foremost statistical analysis algorithm used to predict a binary (0,1) outcome in research based on one or more predictors (independent variables). Typically, LR is used to predict the probability that the outcome or dependent variable equals 1, categorize outcomes, and analyze risks and odds associated with the dependent variables. Its ability to achieve these three goals makes it a unique algorithm [33].

2.4.6. Naïve Bayes

NB classifier is like linear models. However, NB models tend to train faster. The efficiency of the NB model lies in its ability to learn parameters and collect statistics from each feature by exploring each feature individually. Therefore, the NB classifier can be trained to make predictions quickly, and the training process is easy to understand. The model works well on high-dimensional sparse data, is robust to parameters, and can be used as a baseline [32].

2.4.7. Support Vector Machine

SVM is an advanced extension that enables the creation of complex models not simply identified by hyperplanes and input space. The SVM model is used in regression (SVR) and classification (SVC). During training, the model discerns the relevance per training data point in delineating the decision boundary between two classes. Generally, a selective subset of training points, specifically those residing on the border between classes, influences the decision boundary—these are referred to as support vectors, as depicted in Figure 7. In prediction, the algorithm measures the proximity of each support vector, guiding the classification decision accordingly [32].

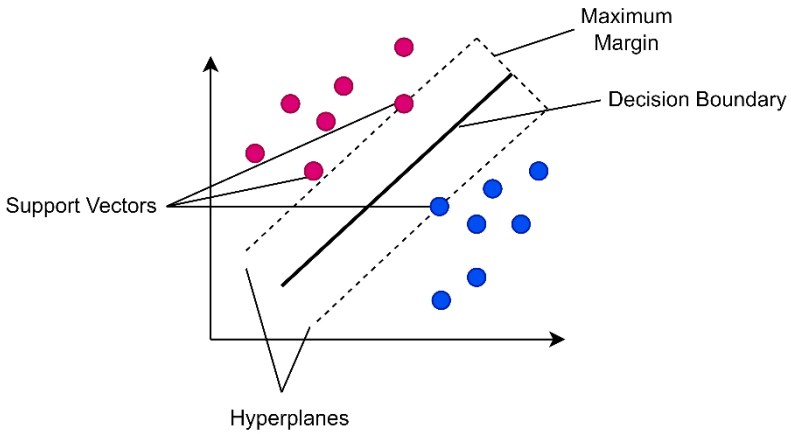


Figure 7. SVM Procedure

2.4.8. Random Forest

Random Forest (RF) is a widely adopted ensemble method primarily designed for classification. In contrast to DT, the RF model grows multiple trees instead of a single tree. This approach enhances the randomness of samples, mitigating the overfitting problem commonly encountered by DT models. As a result, RF provides an excellent predictive model

known for its reliable predictions. The functioning of RF is akin to Decision Trees; however, the concluding prediction is nominated based on the majority vote from all trees. Every tree provides a classification or a "vote," in a Random Forest, the algorithm selects the classification that receives the most votes from all the trees in the forest, as shown in Figure 8. In the case of regression, the method involves averaging the outcomes from all trees [34].

Figure 8. Random Forest Procedure

2.5. Evaluation Metrics

When evaluating the effectiveness of ML and DL models, it's crucial to select performance metrics tailored to the specific problem. The precision of our study's results was ensured by evaluating them using parameters of the confusion matrix - namely Accuracy, Precision, Recall, and F-Measure. Accuracy, a standard statistic, is computed as the ratio of the sum of the True Positive (TP) and True Negative (TN) (Samples correctly classified) to the total number of samples as written in Equation 1. A higher accuracy indicates a better performance of the model utilized. Importantly, it is possible to calculate Precision and Recall on average and per class, allowing for adaptability to different scenarios [28].

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (1)$$

Precision, a pivotal metric, intricately assesses the model's accuracy in correctly identifying samples allocated to a specific attack or normal traffic category within the total samples assigned to that category. The precision computation, as articulated in Equation 2, establishes a ratio of True Positive (TP) samples to the combined count of False Positive (FP) and True Positive (TP) samples. This ratio offers a comprehensive understanding of precision concerning the entirety of detected samples. A greater Precision signifies a reduced false positive rate, underscoring its paramount importance, especially in scenarios where the cost associated with a false positive is significantly elevated [28].

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

Moreover, within the set of all samples that genuinely link to the attack (or normal traffic), Recall is measured as the ratio of accurately identified samples to those specifically belonging to the attack (or normal traffic). The calculation, expressed by Equation 3, involves determining Recall through the division of True Positives (TP) by the sum of True Positives (TP) and False Negatives (FN). It's worth noting that the presence of False Negatives significantly affects the Recall value, adding a layer of complexity and precision to the process. The relevance of Recall becomes particularly evident in scenarios characterized by a notably high False Negative (FN) rate, making it a pivotal factor in the selection of the optimal model [28].

$$PRecall\ (\text{Sensitivity}) = \frac{TP}{TP + FN} \quad (3)$$

Lastly, the F-measure, also known as the F-score, is calculated using the weighted harmonic mean of Recall and Precision, as shown in Equation 4. This metric is particularly useful for evaluating imbalanced data [28].

$$F - Measure = 2 \times \frac{P \times R}{P + R} = \frac{2TP}{2TP + FP + FN} \quad (4)$$

2.6. Experimental Setup

Several models were developed employing both ML and DL algorithms to carry out the mentioned experiments. Python 3.7.13 was used to create the models on the Google Collab notebook platform. 9,000 records were used for the experiments with the target class consisting of three labels: "Normal," " Brute Force," or "Flood." A different set of features were meticulously utilized for each experiment. The initial experiment incorporated six features - Protocol, Flow IAT Max, Fwd Pkts/s, Bwd Pkts/s, Pkt Len Max, and Fwd Act Data Pkts. Subsequently, in the second experiment, only three features were utilized - Fwd Pkts/s, Bwd Pkts/s, and Pkt Len Max. Grid Search with Cross Validation was applied in both experiments to fine-tune the model's parameters. Like K-fold cross validation, CV parameter selection uses different sets for evaluation. Furthermore, Grid Search attempts every possible combination of settings until it obtains the optimal value. The primary goal is to enhance the results' precision while reassuring their validity.

A single input layer represents the number of features, with the first experiment incorporating six features while the second experiment utilizes three features in the construction of the ANN model. Subsequently, three hidden layers were established, culminating in a single output layer with three neurons. The Rectified Linear Unit (ReLU) activation function is applied to the hidden layers, while the SoftMax function is employed for the output layer. The optimal parameter values for each algorithm utilized in this study are shown in Table 6.

Table 6. Models Parameters Optimization

Model	Best Parameters	Optimal Value
ANN	Quantity of Hidden Layers	3
	Quantity of Neurons in Hidden Layers	96,64 and 16
	Activation Function applied in Hidden Layers	ReLU
	Quantity of Neurons in the Output Layer	3
Decision Tree	Criterion	gini
	Max Depth	10
Gradient Boosting	Max Depth	10
	Max Features	log2
	No. of Estimators	5
KNN	No. of Neighbors	2
Logistic Regression	C	0.1
	Penalty	None
Naïve Bayes	-	-
SVM	C	100
	Gamma	1
	Kernel	RBF
Random Forest	Bootstrap	True
	Max Depth	10
	Max Features	Auto
	No. of Estimators	11

3. Results and Discussions

In both experiments, the performance of the multi-class Machine ML and DL models was assessed based on metrics such as Accuracy, Precision, Recall, and F-measure. Each experiment employed a distinct set of features. Table 7 demonstrates the evaluation results of each model.

Table 7. Experiments Results

Model	Evaluation Matrix	Experiment 1	Experiment 2
ANN	Accuracy	95.39%	94.39%
	Precision	0.96	0.95
	Recall	0.95	0.94
	F-measure	0.95	0.94
Decision Tree	Accuracy	95.72%	95.5%
	Precision	0.96	0.96
	Recall	0.96	0.95
	F-measure	0.96	0.95
Gradient Boosting	Accuracy	95.94%	95.28%
	Precision	0.96	0.96
	Recall	0.96	0.95
	F-measure	0.96	0.95
KNN	Accuracy	95.28%	95.06%
	Precision	0.95	0.95
	Recall	0.95	0.95
	F-measure	0.95	0.95
Logistic Regression	Accuracy	92.94%	91.17%
	Precision	0.94	0.91
	Recall	0.93	0.91
	F-measure	0.93	0.91
Naïve Bayes	Accuracy	84.56%	79.67%
	Precision	0.88	0.86
	Recall	0.84	0.80
	F-measure	0.85	0.78
SVM	Accuracy	95.5%	92.28%
	Precision	0.96	0.92
	Recall	0.95	0.92
	F-measure	0.95	0.92
Random Forest	Accuracy	95.61%	95.17%
	Precision	0.96	0.95
	Recall	0.96	0.95
	F-measure	0.96	0.95

Table 7 shows that most models achieved excellent and consistent results in both experiments. There are slight differences in the performance metrics between experiment 1 and experiment 2 across all models. However, in experiment 1, GB stood out with the most favorable outcomes across all evaluation metrics, achieving an accuracy of 95.94%. This indicated its effectiveness in correctly classifying instances. This can be attributed to GB's unique advantages, particularly its flexibility in parameter tuning options and loss functions, which allow it to adapt exceptionally well to the task at hand [31].

Additionally, as depicted in Figure 9, all models demonstrated improved performance in the initial experiment with six features compared to the second one with only three features, where the performance slightly declined by a percentage ranging from 0.22% to 4.89%. Despite this, we still obtained excellent results, further underscoring the importance of the three features: Fwd Pkts/s, Bwd Pkts/s, and Pkt Len Max in IoT device attack detection. Fwd Pkts/s and Bwd Pkts/s denote the rate of forward and backward packets per second, respectively, while Pkt Len Max indicates the maximum length of a packet. After applying RFE in experiment 2, these features showed a high correlation to the target class, with Pkt Len Max being the highest among them, which confirms its significance for the effective detection of IoT attacks.

The following are the key findings regarding the performance of the different models. ANN demonstrated high accuracy, precision, recall, and F-measure in both experiments, indicating its robustness and effectiveness in classification tasks. There's a slight drop in performance from Experiment 1 to Experiment 2, which might be due to variations in the dataset. DT is sensitive to changes in the input data and feature set. Even minor variations in the dataset

or feature selection process can impact the tree's structure and its classification decisions. Therefore, the decrease in performance for DT from Experiment 1 to Experiment 2 could result from sensitivity to changes in the feature set. GB continues to perform well in Experiment 1 and Experiment 2 despite a reduction in the number of features. This indicates that GB can effectively adapt to changes in feature sets while maintaining its predictive power. GB combines multiple weak learners to form a strong ensemble model, allowing it to capture complex relationships between features and target variables. Despite the complexity, GB manages to generalize well and achieve high performance on unseen data. False negatives, where attacks go undetected, pose a significant risk to IoT security, allowing malicious activities to persist undetected. KNN's balanced recall ensures that it effectively captures most of the true positive instances (actual attacks), minimizing the chances of false negatives and improving the overall detection capability of the system. LR balances performance, interpretability, and computational efficiency in IoT attack detection. While it may not achieve the highest accuracy, its transparency, simplicity, and computational efficiency make it a valuable tool for classification tasks in IoT environments, especially when interpretability and resource constraints are important considerations. NB exhibits low accuracy, precision, recall, and F-measure among the models in both experiments. NB assumes that features are conditionally independent given the class label. In practice, this assumption is often violated, especially in complex datasets like those involving IoT attacks. This violation can lead to suboptimal performance, as seen in our results. The performance drop between the two experiments highlights SVM's sensitivity to the feature selection process. SVM relies on a well-chosen set of features to create a robust separating hyperplane. Removing critical features can impact its ability to classify instances accurately. RF shows minimal performance degradation when transitioning from Experiment 1 to Experiment 2 despite the reduction in features. Its strengths in scalability, non-linear relationship modeling, and handling missing data further enhance its suitability for real-world IoT security applications.

IoT attack detection datasets often suffer from class imbalances. Ensuring that the models do not become biased towards the majority class was a crucial challenge that was addressed in our study. Moreover, the experiments also revealed other challenges, particularly related to feature selection sensitivity and computational complexity. While models such as GB and RF showed robustness and high performance, others like SVM and NB highlighted the critical impact of feature selection. Addressing these challenges requires careful tuning. Moreover, our study was conducted in a simulated environment rather than a real-world setting. It is critical to evaluate the robustness of the models to adversarial attacks in the real world. IoT devices have limited resources and computing capabilities. Therefore, the ML models built for IoT attack detection must be efficient and lightweight to avoid performance degradation.

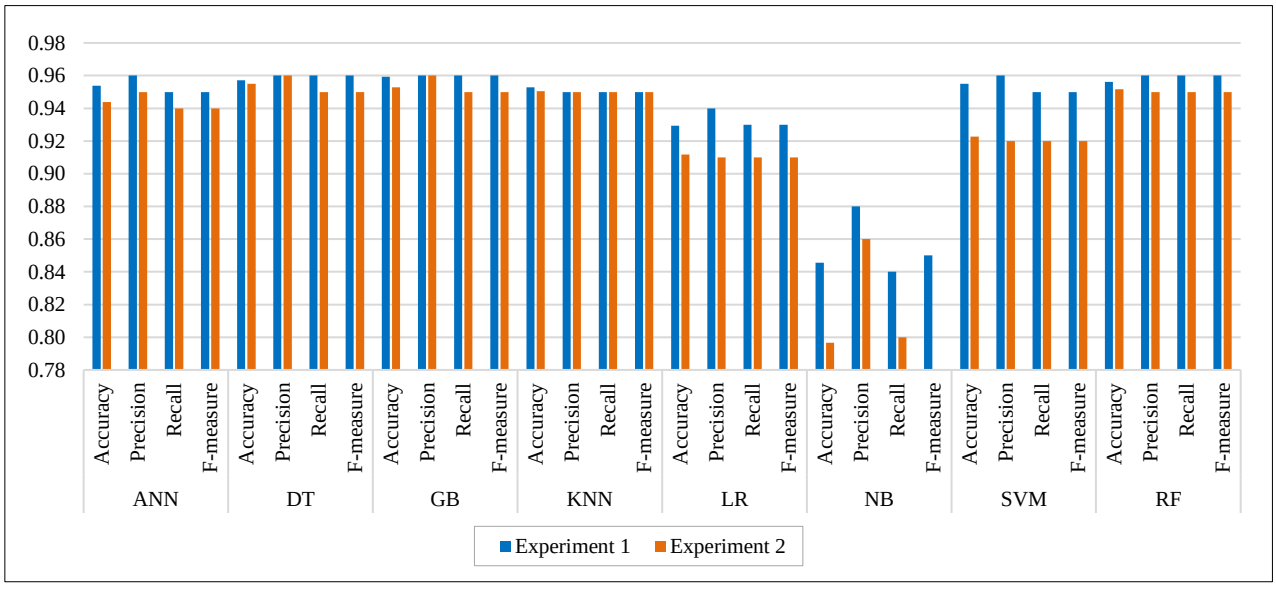
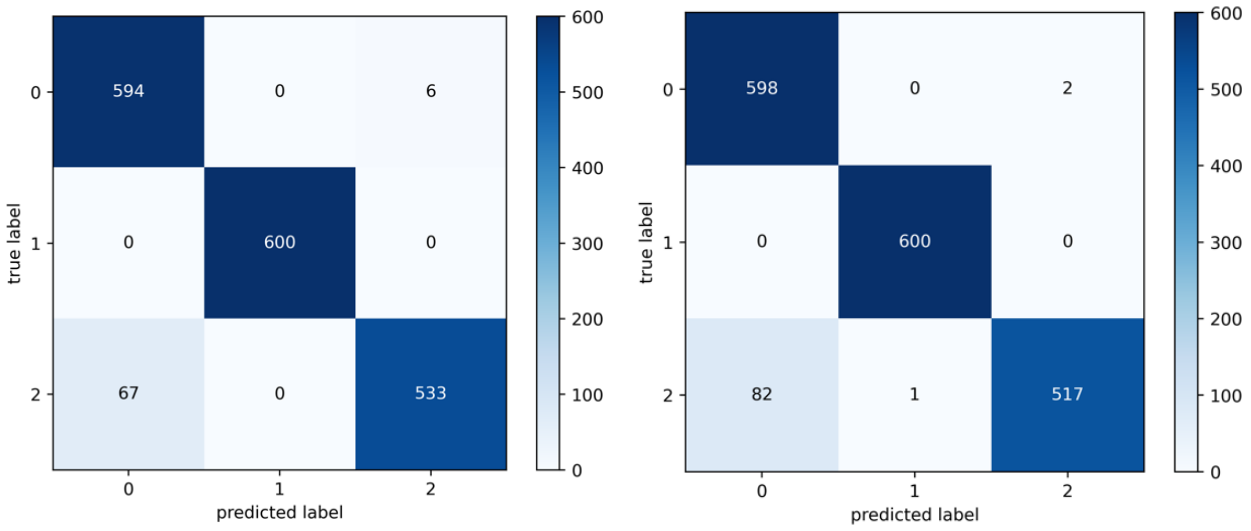


Figure 9. Results Comparison

The confusion matrix was also analyzed for both experiments, as shown in Figure 10(a) and (b), to understand better the types of errors made by the highest accuracy model GB while predicting testing data. The values for correct and incorrect predictions are computed and broken down by each class: class (0) represents RSTP Brute-Force attack, (1) Flood attack, and (2) Normal. The testing data portion contained 1,800 instances divided approximately the same between classes; the diagonal of the confusion matrix depicts the instances that were accurately predicted for each class, while what is left indicates wrong predictions. As illustrated in Figure 10(a) and (b), most instances were classified correctly. However, the Flood attack class and RSTP Brute-Force attack were mostly correctly predicted with few wrong predictions. On the other hand, the normal class showed some wrong predictions, where 67 out of 594 and 78 out of 598 were predicted as the RSTP Brute-Force attack class in the first and second experiments, respectively. This indicates that our model had more ability to classify RSTP Brute-Force and Flood attacks on the IoT devices in the CIC IoT Dataset 2022 [28].



(a) (b)

Figure 10. represents a confusion matrix in which (a) Confusion Matrix for Experiment 1 Using GB Model; (b) Confusion Matrix for Experiment 2 Using GB Model

4. Conclusions

The Internet of Things (IoT) has witnessed a significant proliferation of devices, users, and technological advancements in recent years, revolutionizing our daily activities. However, this convenience is accompanied by heightened security concerns, primarily due to the escalating threat of cyberattacks. In response to these critical challenges, our research is dedicated to detecting IoT network attacks, specifically Flood DoS and RSTP brute-force attacks, using Machine Learning (ML) and Deep Learning (DL) methodologies. This research is vital for safeguarding the security of IoT devices, a crucial aspect of today's digital landscape. Our study begins with an introduction and a review of existing literature, analyzing current research to identify gaps and propose a new perspective on this field. Leveraging the CIC IoT Dataset 2022, a multi-dimensional IoT profiling dataset developed for Cybersecurity, we conducted two meticulous experiments utilizing distinct feature sets: one comprising six features and another with three. The dataset underwent preprocessing with CICFlowMeter to extract over 80 statistical features. To tackle the challenge of dataset imbalance, we utilized under-sampling and over-sampling techniques to ensure that the dataset was not biased toward the majority class while ensuring the reliability and validity of our findings. Feature selection was conducted through Manual Feature Correlation Elimination and Recursive Feature Elimination (RFE). Pkt Len Max, demonstrating the strongest correlation with the target class, was identified as significant for analysis and attack detection due to its association with network attacks. This feature, representing the maximum packet length in a network flow, is a crucial indicator of potential network attacks. Subsequently, the dataset was partitioned into an 80:20 ratio for training and testing purposes. Eight supervised algorithms—ANN, DT, GB, KNN, LR, NB, SVM, and RF—were utilized, and their efficacy was evaluated using key metrics, including Accuracy, Precision, Recall, and F1-score. Notably, Grid Search, with cross-validation, was employed for parameter tuning, enhancing the robustness of our approach.

Our findings from the initial and subsequent experiments were promising. The GB algorithm achieved an impressive accuracy of 95.94%, while the DT algorithm attained an equally commendable accuracy of 95.5% in the subsequent experiment. Analysis of the confusion matrix revealed the superior performance of the GB model in classifying Flood and RSTP brute-force attacks in both instances. These results demonstrate the effectiveness of our suggested approach, ML and DL techniques, in enhancing the detection of such attacks and have implications for the field of IoT network security. By accurately identifying and classifying these attacks, we can strengthen the security of IoT networks, thereby protecting the privacy and integrity of IoT devices and the data they generate. This underscores the pivotal role of ML and DL methodologies in bolstering IoT network security.

While providing valuable insights using eight supervised classifiers, we are still eager to explore additional algorithms in future research, which opens up numerous opportunities to enhance our detection capabilities further. The potential for future advancements, such as integrating ML algorithms to construct a multi-layered model, is promising and could significantly improve detection performance. Classifying and detecting new attacks and contributing to the ever-evolving domains of IoT and Cybersecurity is another direction to explore.

5. Declarations

5.1. Author Contributions

Conceptualization, M.A., A.Sh., F.A., A.S., D.A., M.Ab., W.A., and A.A.; methodology, M.A., A.Sh., F.A., A.S., D.A., M.Ab., W.A., and A.A.; software, F.A., A.S., and D.A.; validation, M.A., A.Sh., F.A., A.S., D.A., M.Ab., W.A., and A.A.; formal analysis, M.A., W.A., and A.A.; investigation, M.A., A.Sh., F.A., A.S., D.A., and M.Ab.; resources, M.A.; data curation, A.Sh. and F.A.; writing—original draft preparation, M.A., A.Sh., F.A., A.S., D.A., M.Ab., W.A., and A.A.; writing—review and editing, M.A., A.Sh., F.A., W.A., and A.A.; visualization, F.A.; supervision, M.A.; project administration, M.A.; funding acquisition, M.A. All authors have read and agreed to the published version of the manuscript.

5.2. Data Availability Statement

The data presented in this study are available on request from the corresponding author.

5.3. Funding

The authors received no financial support for the research, authorship, and/or publication of this article.

5.4. Acknowledgements

We would like to thank Dr. Rachid Zagrouba for reviewing the paper and providing feedback.

5.5. Institutional Review Board Statement

Not applicable.

5.6. Informed Consent Statement

Not applicable.

5.7. Declaration of Competing Interest

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

6. References

[1] Statista. (2024). IoT connected devices worldwide 2019-2030. Statistics, Hamburg, Germany. Available online: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide (accessed on May 2024).

[2] Bonaccorsi, M., Betti, S., Rateni, G., Esposito, D., Brischetto, A., Marseglia, M., Dario, P., & Cavallo, F. (2017). 'HighChest': An augmented freezer designed for smart food management and promotion of eco-efficient behaviour. Sensors (Switzerland), 17(6), 1357. doi:10.3390/s17061357.

[3] Aljabri, M., Aljameel, S. S., Mohammad, R. M. A., Almotiri, S. H., Mirza, S., Anis, F. M., Aboulnour, M., Alomari, D. M., Alhamed, D. H., & Altamimi, H. S. (2021). Intelligent techniques for detecting network attacks: Review and research directions. Sensors, 21(21). doi:10.3390/s21217070.

[4] Arnaldo, I., Cuesta-Infante, A., Arun, A., Lam, M., Bassias, C., & Veeramachaneni, K. (2017). Learning representations for log data in cybersecurity. In S. D. Shlomi & Lodha (Eds.), Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics): Springer International Publishing, Vol. 10332 LNCS, 250–268. doi:10.1007/978-3-319-60080-2_19.

[5] Hammood, B. A. K., & Sadiq, A. T. (2023). Ensemble machine learning approach for IoT intrusion detection systems. Iraqi Journal for Computers and Informatics, 49(2), 93-99.

[6] Loganathan, G., Samarabandu, J., & Wang, X. (2018). Sequence to Sequence Pattern Learning Algorithm for Real-Time Anomaly Detection in Network Traffic. Canadian Conference on Electrical and Computer Engineering, 2018-May, 1–4. doi:10.1109/CCECE.2018.8447597.

[7] Lambert II, G. M. (2017). Security analytics: Using deep learning to detect cyber-attacks. UNF Graduate Theses and Dissertations University of North Florida, Florida, United States. Available online: https://digitalcommons.unf.edu/etd/728/ (accessed on May 2024).

[8] Stevanovic, M., & Pedersen, J. M. (2016). Detecting bots using multi-level traffic analysis. International Journal on Cyber Situational Awareness, 1(1), 182–209. doi:10.22619/ijcsa.2016.100109.

[9] Alzahrani, R. A., & Aljabri, M. (2023). AI-Based Techniques for Ad Click Fraud Detection and Prevention: Review and Research Directions. Journal of Sensor and Actuator Networks, 12(1). doi:10.3390/jsan12010004.

[10] Aljabri, M., Zagrouba, R., Shaahid, A., Alnasser, F., Saleh, A., & Alomari, D. M. (2023). Machine learning-based social media bot detection: a comprehensive literature review. Social Network Analysis and Mining, 13(1), 20. doi:10.1007/s13278-022-01020-5.

[11] Aljabri, M., Alahmadi, A. A., Mohammad, R. M. A., Alhaidari, F., Aboulnour, M., Alomari, D. M., & Mirza, S. (2023). Machine Learning-Based Detection for Unauthorized Access to IoT Devices. Journal of Sensor and Actuator Networks, 12(2), 27. doi:10.3390/jsan12020027.

[12] Kumari, P., & Jain, A. K. (2023). A comprehensive study of DDoS attacks over IoT network and their countermeasures. Computers and Security, 127, 103096. doi:10.1016/j.cose.2023.103096.

[13] Anwer, M., Umer, M., Khan, S. M., & Waseemullah. (2021). Attack Detection in IoT using Machine Learning. Engineering, Technology and Applied Science Research, 11(3), 7273–7278. doi:10.48084/etasr.4202.

[14] Tomer, V., & Sharma, S. (2022). Detecting IoT attacks Using an Ensemble Machine Learning Model. Future Internet, 14(4), 102. doi:10.3390/fi14040102.

[15] Alsamiri, J., & Alsubhi, K. (2019). Internet of things cyber-attacks detection using machine learning. International Journal of Advanced Computer Science and Applications, 10(12), 627–634. doi:10.14569/ijacsa.2019.0101280.

[16] Lashkari, A. H. (2024). CICFlowmeter-V4.0: GitHub. Available online: https://github.com/ahlashkari/CICFlowMeter (accessed on May 2024).

[17] Htwe, C. S., Thant, Y. M., & Thwin, M. M. S. (2020). Botnets attack detection using machine learning approach for IoT environment. Journal of Physics: Conference Series, 1646(1), 012101. doi:10.1088/1742-6596/1646/1/012101.

[18] Gaber, T., El-Ghamry, A., & Hassanien, A. E. (2022). Injection attack detection using machine learning for smart IoT applications. Physical Communication, 52. doi:10.1016/j.phycom.2022.101685.

[19] Aysa, M. H., Ibrahim, A. A., & Mohammed, A. H. (2020). IoT ddos Attack Detection Using Machine Learning. 4th International Symposium on Multidisciplinary Studies and Innovative Technologies, ISMSIT 2020 - Proceedings, 1–7, Istanbul, Turkey. doi:10.1109/ISMSIT50672.2020.9254703.

[20] Krishnan, S., Neyaz, A., & Liu, Q. (2021). IoT Network Attack Detection using Supervised Machine Learning. International Journal of Artificial Intelligence and Expert Systems, 10(2), 18–32.

[21] Saran, N., & Kesswani, N. (2022). A comparative study of supervised Machine Learning classifiers for Intrusion Detection in Internet of Things. Procedia Computer Science, 218, 2049–2057. doi:10.1016/j.procs.2023.01.181.

[22] Altulaihan, E., Almaiah, M. A., & Aljughaiman, A. (2024). Anomaly Detection IDS for Detecting DoS Attacks in IoT Networks Based on Machine Learning Algorithms. Sensors, 24(2), 713. doi:10.3390/s24020713.

[23] Pecori, R., Tayebi, A., Vannucci, A., & Veltri, L. (2020). IoT Attack Detection with Deep Learning Analysis. Proceedings of the International Joint Conference on Neural Networks, 1–8. doi:10.1109/IJCNN48605.2020.9207171.

[24] Alkahtani, H., & Aldhyani, T. H. H. (2021). Botnet Attack Detection by Using CNN-LSTM Model for Internet of Things Applications. Security and Communication Networks, 2021. doi:10.1155/2021/3806459.

[25] Al-Zubidi, A. F., Farhan, A. K., & Towfek, S. M. (2024). Predicting DoS and DDoS attacks in network security scenarios using a hybrid deep learning model. Journal of Intelligent Systems, 33(1), 20230195. doi:10.1515/jisys-2023-0195.

[26] Islam, N., Farhin, F., Sultana, I., Kaiser, S., Rahman, S., Mahmud, M., Hosen, S., & Cho, G. H. (2021). Towards Machine Learning Based Intrusion Detection in IoT Networks. Computers, Materials and Continua, 69(2), 1801–1821. doi:10.32604/cmc.2021.018466.

[27] Karamollaoğlu, H., Doğru, İ. A., & Yücedağ, İ. (2024). An Efficient Deep Learning-based Intrusion Detection System for Internet of Things Networks with Hybrid Feature Reduction and Data Balancing Techniques. Information Technology and Control, 53(1), 243–261. doi:10.5755/j01.itc.53.1.34933.

[28] Dadkhah, S., Mahdikhani, H., Danso, P. K., Zohourian, A., Truong, K. A., & Ghorbani, A. A. (2022). Towards the Development of a Realistic Multidimensional IoT Profiling Dataset. 2022 19th Annual International Conference on Privacy, Security and Trust, PST 2022, 1–11. doi:10.1109/PST55820.2022.9851966.

[29] Alpaydin, E. (2020). Introduction to Machine Learning (Adaptive Computation and Machine Learning Series). Natural Language Engineering: The MIT Press, 14(01), 133–137. doi:10.1017/s1351324906004438.

[30] Chiroma, H., Noor, A. S. M., Abdulkareem, S., Abubakar, A. I., Hermawan, A., Qin, H., Hamza, M. F., & Herawan, T. (2017). Neural networks optimization through genetic algorithm searches: A review. Applied Mathematics and Information Sciences, 11(6), 1543–1564. doi:10.18576/amis/110602.

[31] Bahad, P., & Saxena, P. (2020). Study of AdaBoost and Gradient Boosting Algorithms for Predictive Analytics. International Conference on Intelligent Computing and Smart Communication 2019, 235–244. doi:10.1007/978-981-15-0633-8_22.

[32] Müller, A. C., & Guido, S. (2016). Introduction to machine learning with Python: a guide for data scientists. O'Reilly Media, California, United States.

[33] Hilbe, J. M. (2016). Practical guide to logistic regression. Practical Guide to Logistic Regression. CRC Press, Florida, United States. doi:10.18637/jss.v071.b03.

[34] Talekar, B. (2020). A Detailed Review on Decision Tree and Random Forest. Bioscience Biotechnology Research Communications, 13(14), 245–248. doi:10.21786/bbrc/13.14/57.