electronics

Article

An Intelligent System to Detect Advanced Persistent Threats in Industrial Internet of Things (I-IoT)

Safdar Hussain Javed [1], Maaz Bin Ahmad [1,*], Muhammad Asif [2], Sultan H. Almotiri [3], Khalid Masood [2] and Mohammad A. Al Ghamdi [3]

[1] College of Computing and Information Sciences, Karachi Institute of Economics and Technology, Karachi 75190, Pakistan; 59356@kiet.edu.pk
[2] Department of Computer Science, Lahore Garrison University, Lahore 54000, Pakistan; drmuhammadasif@lgu.edu.pk (M.A.); khalid.masood@lgu.edu.pk (K.M.)
[3] Computer Science Department, Umm Al-Qura University, Makkah City 24211, Saudi Arabia; shmotiri@uqu.edu.sa (S.H.A.); maeghamdi@uqu.edu.sa (M.A.A.G.)
* Correspondence: maaz@kiet.edu.pk; Tel.: +92-3335264960

Abstract: The Industrial Internet of Things (I-IoT) is a manifestation of an extensive industrial network that interconnects various sensors and wireless devices to integrate cyber and physical systems. While I-IoT provides a considerable advantage to large-scale industrial enterprises, it is prone to significant security challenges in the form of sophisticated attacks such as Advanced Persistent Threat (APT). APT is a serious security challenge to all kinds of networks, including I-IoT. It is a stealthy threat actor, characteristically a nation-state or state-sponsored group that launches a cyber attack intending to gain unauthorized access to a computer network and remain undetected for a longer period. The latest intrusion detection systems face several challenges in detecting such complex cyber attacks in multifarious networks of I-IoT, where unpredictable and unexpected cyber attacks of such sophistication can lead to catastrophic effects. Therefore, these attacks need to be accurately and promptly detected in I-IoT. This paper presents an intelligent APT detection and classification system to secure I-IoT. After pre-processing, several machine learning algorithms are applied to detect and classify complex APT signatures accurately. The algorithms include Decision Tree, Random Forest, Support Vector Machine, Logistic Regression, Gaussian Naive Bayes, Bagging, Extreme Gradient Boosting and Adaboost, which are applied on a publicly available dataset KDDCup99. Moreover, a comparative analysis is conducted among ML algorithms to select the appropriate one for the targeted domain. The experimental results indicate that the Adaboost classifier outperforms the others with 99.9% accuracy with 0.012 s execution time for detecting APT attacks. Furthermore, results are compared with state-of-the-art techniques that depict the superiority of the proposed system. This system can be deployed in mission-critical scenarios in the I-IoT domain.

Keywords: advanced persistent threat; intrusion detection system; industrial internet of things; machine learning; intelligent system

check for updates

Citation: Javed, S.H.; Ahmad, M.B.; Asif, M.; Almotiri, S.H.; Masood, K.; Al Ghamdi, M.A. An Intelligent System to Detect Advanced Persistent Threats in Industrial Internet of Things (I-IoT). *Electronics* **2022**, *11*, 742. https://doi.org/10.3390/electronics11050742

Academic Editor: Nurul I. Sarkar

Received: 20 January 2022
Accepted: 10 February 2022
Published: 28 February 2022

1. Introduction

Industrial Internet of Things (I-IoT) refers to unified sensors, computer systems that are a multitude of networked domains connected through the industrial enterprise applications. It is an evolution of Distributed Control System (DCS) that allows higher automation by using cloud computing to optimize industrial process controls. As industrial intelligent sensor systems process terabytes of data daily, I-IoT positions itself at a crucial stage, where prediction accuracy is of utmost importance. Industrial enterprises are massively adopting I-IoT to induce automation for efficiency enhancement with faultless monitoring of resources and reducing safety risks to enterprise wide networks [1]. As the world's I-IoT market is expected to reach well over 1.1 trillion dollars by 2026, its adoption rate has also been phenomenal, reaching up to 25% of consumer sectors worldwide [2]. Figure 1 shows the worldwide increasing I-IoT adaptation rate projection till 2025.

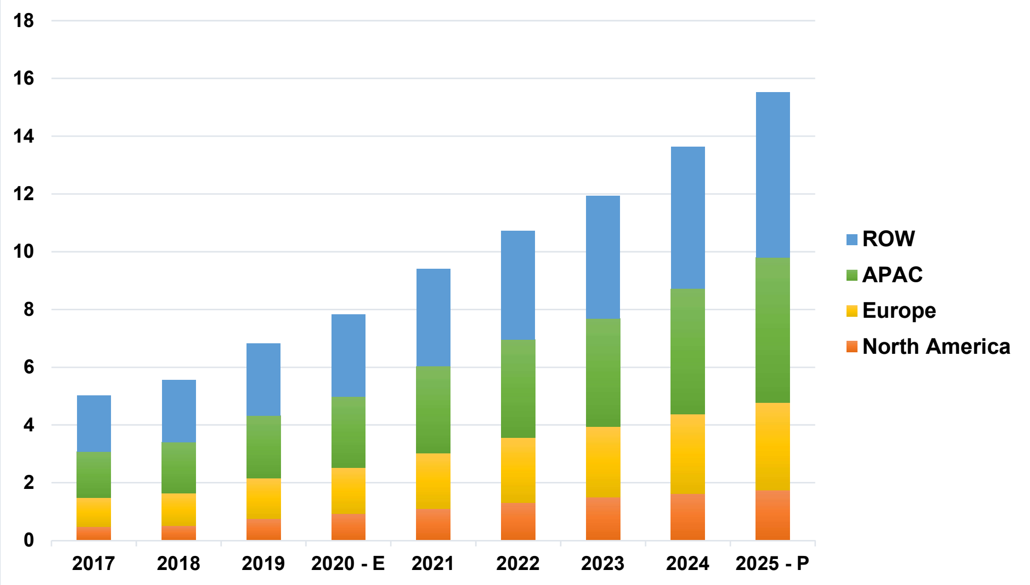


Figure 1. Worldwide I-IoT adaptation rate [3].

According to a survey [4], 8% of organizations around the world experienced cyber attacks targeting I-IoT systems from 2018 to 2021.The UK stood at 86%, Japan with 64%, whereas 84% of healthcare organizations, 79% of manufacturing and production organizations and 78% of transport organizations worldwide experienced cyber attacks on their I-IoT network in the same period. A cyber attack on the I-IoT network comes with various threats having different scales and sizes from different threat actors [5]. Generally, various levels of the attacks including cyber warfare (top-level), cyber espionage (mid-level) and cyber-crime (low-level) may be launched on organizations [6]. The main targets of such attacks are switches, sensors, CCTV cameras that constitute the machine-to-machine (M2M) and industrial communication networks [7,8]. Moreover, industrial installations have geographically dispersed infrastructures; this provides a greater horizon for the cyber adversaries to inflict maximum damage to industrial infrastructure [9–14].

I-IoT provides phenomenal opportunities to enterprise organizations; however, many security challenges also persist in realizing the prospects offered by this domain. These security challenges not only pose a threat to an individual but also cause enterprise wide operational failure with catastrophic effects. The I-IoT system is confronted with the following core security challenges [15–17]:

- Resource Constraint: I-IoT has restricted capability in computing that often makes anomaly detection difficult.
- Limited Communication: Compared to traditional devices, I-IoT devices generate very little traffic with limited user interaction.
- Cyber Security: I-IoT devices are prone to various cyber security challenges. With new sensors, intelligent devices frequently emerging rapidly, they are faced with multiple vulnerabilities ranging from simple to complex types of cyber attacks.
- Integration: In I-IoT devices, the integration of information technology (IT) and operational technology (OT) is difficult. Therefore, it is essential to securely integrate them without compromising data.
- Data Storage: I-IoT devices have less storage capacity, which hinders security updates in the form of security patches.

I-IoT systems are mission-critical systems with security limitations. Therefore, it is essential to develop a real time security monitoring mechanism that focuses on preventing unauthorized and malicious users in industrial critical systems. The conventional security models are generally based on cryptographic techniques that require more processing time to process a large amount of data to identify complex threats such as APT. So, such models are not widely adopted for I-IoT environments due to their large voluminous data size. It is required to identify the security threats promptly that may help to take the necessary actions for minimizing the damage. By identifying the APT data traffic in the I-IoT domain, organizations deploying I-IoT sensors may defend their most valuable assets and data from digital disruption in an industrial processing unit.

In this paper, a machine learning based intelligent detection and classification system is proposed that identifies APT data traffic in the I-IoT domain. Several machine learning classifiers such as Decision Tree (DT), Random Forest (RF), Support Vector Machine (SVM), Logistic Regression (LR), Gaussian Naive Bayes (GNB), Bagging, Extreme Gradient Boost (XGBoost) and Adaboost are applied to successfully detect and classify the APT signature traffic.To develop and evaluate the performance of the proposed system, a publicly available dataset KDDCup99 is used. A comparative analysis is performed among machine learning techniques and results indicate that the Adaboost classifier outperforms the others with 99.9% accuracy with an execution time of 0.012 s for detecting APT attacks, thus making it suitable for use in I-IoT domain.

The rest of the paper is organized as follows: Section 2 describes various phases of APT and its behavior in I-IoT networks. The literature review is illustrated in Section 3. Section 4 presents the proposed APT detection methodology. Section 5 presents the experimental analysis. Section 6 concludes the paper and suggests the future directions of the research.

2. APT in I-IoT Networks

The APT attack in the I-IoT network outlines various methodical attack phases in the core network. The Supervisory Control and Data Acquisition (SCADA) system is the underlying major hardware component of the I-IoT ecosystem, mainly comprised of an organized entity primarily used to monitor the various industrial core processes. A typical I-IoT SCADA system is shown in Figure 2.

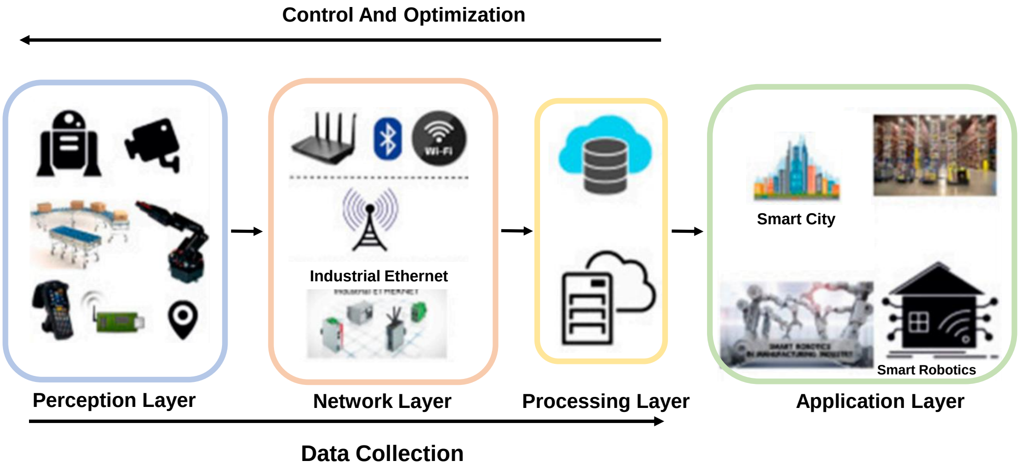


Figure 2. A typical I-IoT SCADA system [18].

The core system of any I-IoT network is composed of the physical layer that includes sensors, actuators and control systems. The network layer consists of wired or wireless communication media, the processing layer houses the core information processing systems and the application layer collects and transmit information from the communications stack [19].

The APT attacks have four significant characteristics , which make them different from regular cyber attacks. Being complex, persistent, targeted and elusive, APT attacks are precisely planned and systematically launched in multiple stages [20]. Stuxnet attack on the Iranian nuclear facility and Shamoon malware attack on the Saudi Aramco facility as well as Sauron, CopyKittens, Volatile Cedar and ShellCrew are a few examples that made headlines in the research community [21]. Typical attack phases of APT in I-IoT are illustrated in Figure 3.

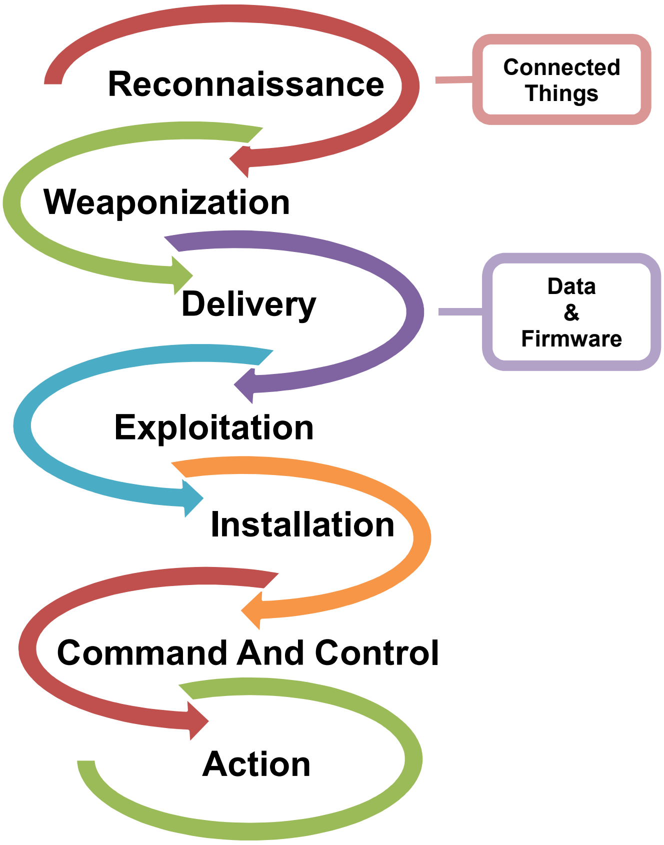


Figure 3. APT Attack phases in I-IoT.

The attack surface of APT starts with the reconnaissance activities such as probing the target industrial network for vulnerable and exploitable components before actually breaching the network. This task is carried out by collecting information about the intended target network by searching the internet for general information about the industrial network or through social engineering tools used for propagation. In the second phase of weaponization, the attacker creates an infected document paired with a customized phishing email or by using a new strain of self-replicating malware to be distributed through wi-fi, USB and other means for point of entry into the core network. The delivery phase involves the transmission of the malware to the intended industrial network where corresponding account and password management measures are weak. This provides an appropriate door to intrusion. In the exploitation phase, the actual manifestation of the attack takes place by exploiting the target network through infiltrated malware. In the installation phase, the attacker installs the APT malware in target machines of the processing layer. The Command and Control (C&C) phase is the post compromised layer through which the infected systems are controlled by the C&C system that mostly resides outside the network for the cyber attacker to gain control of the compromised network. The communication between the host and the C&C system is usually encrypted. The actions on objectives or data ex-filtration phase involves the cyber attacker establishing access to the organization and executing actions to achieve its objectives, which is usually ex-filtration of high valued data. An APT attack scenario is presented in Figure 4.

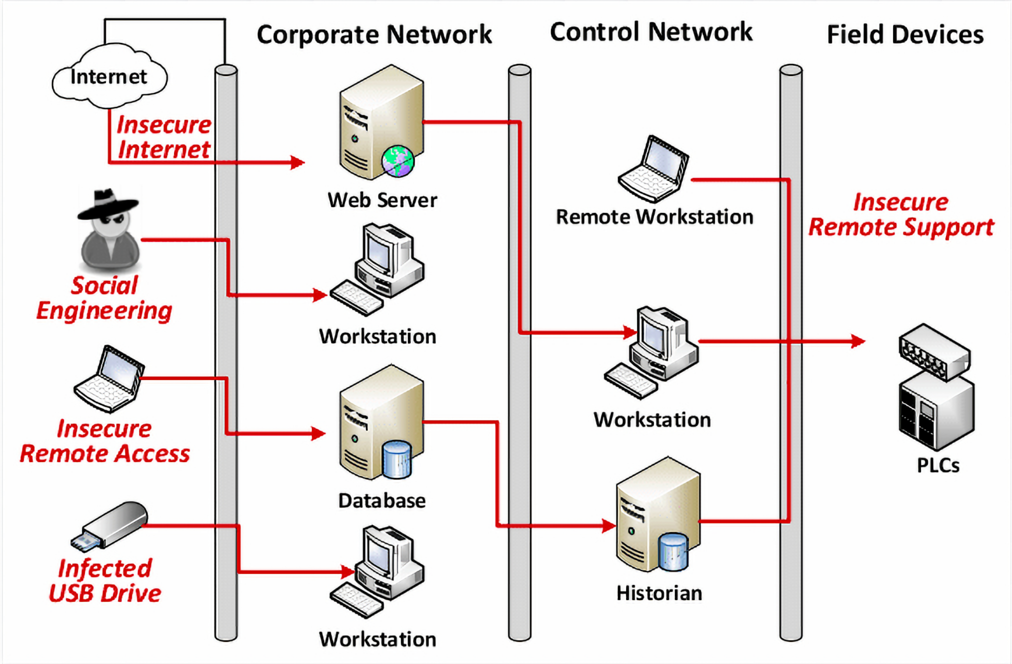


Figure 4. APT attack scenario [22].

3. Related Work

Efficient classification and detection of benign and malicious feature traffic forms the core foundation of any malware detection system. However, accurate classification of APT attacks, especially within the I-IoT domain, is a challenging task. Many pieces of research have been made on security and intrusion detection systems in the domain of I-IoT networks.

Teyou et al. [23] classified anomaly data traffic in the cyber physical system environment by implementing a Convolutional Neural Network (CNN) based Intrusion Detection System (IDS). The classification model uses the NSL-KDD dataset and its performance is evaluated and compared with the state of the art. The model gave an accuracy of 80.07% and 77.15%, respectively, on the 2 and 5 class classification models. Due to the complexity and resource constraint environment, CNN does not seem suitable for I-IoT and the model performance can be further enhanced to deploy IDS in a real time environment. The approach used five attack classes of the dataset that can be applied to all seven APT phases. Latif et al. [24] proposed a cyber security attack prediction system based on a lightweight random neural network (RaNN) for the I-IoT domain. They targeted malicious operation and control, denial of service (DoS), scan, spying, data type probing and wrong setup attacks. The DS2OS dataset was used for the development of the system comprising seven attack classes that are mapped to seven APT attack phases. The experimental results showed that the system gave 99.20% accuracy with a 0.01 learning rate.

Wang et al. [25] analyzed the features of Command and Control (C&C) in APT and found that HTTP based C&C is quite useful. Moreover, they proposed a new independent access feature and applied it to the DNS records. The public dataset has been applied to validate the proposed approach. The drawback of this approach is that it can easily be evaded when the infected hosts connect to the C&C domains while connected to the malicious server. Brogi et al. [26] proposed a framework known as TerminAPTor, which is an APT detector that highlights links between the traces left by attackers in the monitored system during the diverse stages of an attack campaign. TerminAPTor used the Information Flow Tracking (IFT) to tackle this challenge by correlating generated events in each APT step in a statistical manner. Sigholm et al. [27] proposed a Data Leakage Prevention (DLP) approach that targeted the data exfiltration phase of the APT attack cycle. The DLP scheme processed the network traffic to detect data leaks and generate matching signatures of the leak. This system exploited external cyber counterintelligence (CCI) sensors to track the location and path of the data exfiltration. The approach targeted only the last phase; the data exfiltration of an APT cycle and does not counter the first six phases of the APT life cycle and cannot achieve real time detection.

Cheng et al. [28] presented a cyber situation comprehension framework called APTALCM (APT Alerts and Logs Correlation Method) for edge commuting IoT system devices that are based on APT alerts and correlation of logs on a simulated dataset. Based upon the concept of alert instance, the approach considered only Phases 2, 3, 5 and 6 (Point of Entry, C&C communication, data discovery and data exfiltration). The approach achieved 83.7% TRP and 4.2% FPR. The drawback of this approach is that it did not cover Phases 1 and 4 (Reconnaissance and Exploit) of the APT cycle. Parra et al. [29] proposed a cloud based distributed DL framework for detecting phishing and botnet attacks in the IoT domain. The experimental results provided accuracy of 94.30% and 94.80%, respectively, on the N_BaIoT dataset. The dataset is composed of 10 attack classes related to Botnet targeting IoT devices that covers all seven APT attack phases.

Zheng et al. [30] proposed a linear discriminant analysis based extreme learning technique for IoT intrusion detection. Researchers evaluated the technique on the NSL-KDD dataset. The accuracy of their approach was 92.35% and used four main attack classes, i.e., DoS, Remote to Local (R2L), User to Remote (U2R) and probe attack covering all seven APT phases. Khalvati et al. [31] implemented a system to detect and classify IoT attacks using SVM and Bayesian ML algorithms. The authors achieved an accuracy of 91.50% on the KDDCup99 dataset. The approach targeted four attack classes that cover all seven APT phases. Ghafir et al. [32] presented an autonomous system for the detection of APT signatures. The system is composed of three layer detection, i.e., threat detection, alert correlation and attack prediction that provided an accuracy of 84.4%. The system needs to be tested for real time APT signature covering all seven phases of the APT life cycle.

Shudong et al. [33] proposed an attribution classification method of APT malware in IoT using the ML approach. The method analyzes samples, pre-processes the acquired behavioral data, constructs a behavioral data set of malware samples, then uses the TF-IDF method to perform the feature representation forming a vector matrix and calculates the chi-square value of the high latitude feature vector to perform feature selection. SMOTE-RF model is used in the multi-class model to train and test sets for predicted output with accuracy of 80%. Table 1 summarizes the APT specific related work.

It is observed from the existing literature that many intrusion detection systems are available but they cannot accurately classify modern threat signatures on mission critical systems. The existing intrusion detection techniques can be categorized into three categories based on their working principle, i.e., machine learning (ML) based approaches, deep learning based approaches and autonomous approaches that are deployed in different domains, including IoT and I-IoT domains. The publicly available datasets KDDCup99, NSL-KDD and simulated dataset have been utilized mostly for the development and evaluation of detection schemes. These datasets have widely been used to facilitate specific applications of IoT as well as I-IoT domains. Analysis of the above mentioned approaches suggests that deep learning based approaches demand high processing power and storage capacity to identify complex threats such as Advanced Persistent Threats (APT). These are the prime reasons for not making them first choice for developing intrusion detection systems for I-IoT environment. The main issue with autonomous approaches is that they are not scalable. The machine learning based approaches are highly scalable because of their capability to process huge amounts of data and perform massive computations in a cost- and time-effective manner. This positively affects the productivity (fast deployment).

Moreover, it is observed that existing APT detection approaches are not appropriate for the I-IoT domain because of the large voluminous data size that is generated by the system. Furthermore, the existing research work is not compatible with resource constraint edge devices of I-IoT. Therefore, it is required to develop a computationally efficient APT attack system for I-IoT networks that promptly identifies the security threats.

Table 1. Summarized related work.

Papers	Techniques	Dataset	Accuracy	Remarks
Teyou et al. [23]	CNN	NSL-KDD	80.7%	The targeted classes of the dataset are correlated to 7 APT attack phases
Latif et al. [24]	Implementation of RaNN model for classification of anomaly data	DS2OS	99.20%	Covered 7 APT attack phases
Wang et al. [25]	Non-ML autonomous approach	Simulated Dataset	83.3% TPR	Evasion is possible when infected hosts join the C&C domains
Brogi et al. [26]	Non-ML autonomous approach	Simulated Dataset	100% TPR	High FPR rate. Covered 4 APT attack phases
Sigholm et al. [27]	Non-ML autonomous approach	Simulated Dataset	-	Real-time detection is not possible. Targets only 1 APT attack phase of data ex-filtration
Cheng et al. [28]	Non-ML autonomous approach	Simulated Dataset	83.7% TPR	83.7% TRP and 4.2% FPR. Targeted only Phase 1 and 4 (Reconnaissance and Exploit phases) of the APT life cycle
Parra et al. [29]	DL	NBaIoT	94.30%	Focused all 7 APT attack phases
Zheng et al. [30]	Linear discriminant analysis (LDA)	NSL-KDD	92.35%	Targets all 7 APT attack phases
Khalvati et al. [31]	SVM and Bayesian classification	KDDcup99	91.50%	Targets all 7 APT attack phases
Ghafir et al. [32]	Autonomous	Correlation clustered dataset	84.4%	Composed of threat detection, alert correlation and attack prediction layers that covered all 7 APT attack phases
Shudong et al. [33]	ML	Behavioural data of a malware sample	80.0%	APT phases unknown

4. Methodology

In light of the above discussion, the main objective of the present research is to propose a novel lightweight and fast ML based APT signature classification system that can detect and predict malicious nodes using various conventional ML algorithms on publicly available KDDCup99 datasets specifically suitable for the I-IoT domain. The proposed system for APT detection and classification for the I-IoT domain is presented in Figure 5. It consists of data acquisition, pre-processing, prediction and performance evaluation phases. These phases are discussed in the following subsections:

4.1. Data Acquisition Phase

In this phase, the publicly available dataset KDDCup99 is used to develop the proposed system [34]. This dataset is widely used for classification and clustering problem handling for the development of IDS. It was developed by the Cyber Systems and Technology Group of MIT Lincoln Laboratory in 1999 to evaluate computer network IDS and is composed of 4,94,021 single-labeled connection vectors, each containing 42 features. The simulated network traffic can be classified into the following categories:

- Normal
- Probing Attack
- Distributed Denial of Service Attack (DDoS)
- Remote to Local Attack (R2L)
- User to Root Attack (U2R)

The attacks mentioned above occur when an attacker sends packets to an I-IoT node over a network. It aims to exploit vulnerability persistent in the node to gain access and perform network reconnaissance to gather information about the network. The apparent purpose of these types of attacks is to circumvent the security controls of the system. Cyber adversaries use similar attack classes to launch a successful APT attack on their targeted I-IoT network [35]. The attack classes mapped to their corresponding APT attack phase are presented in Table 2.

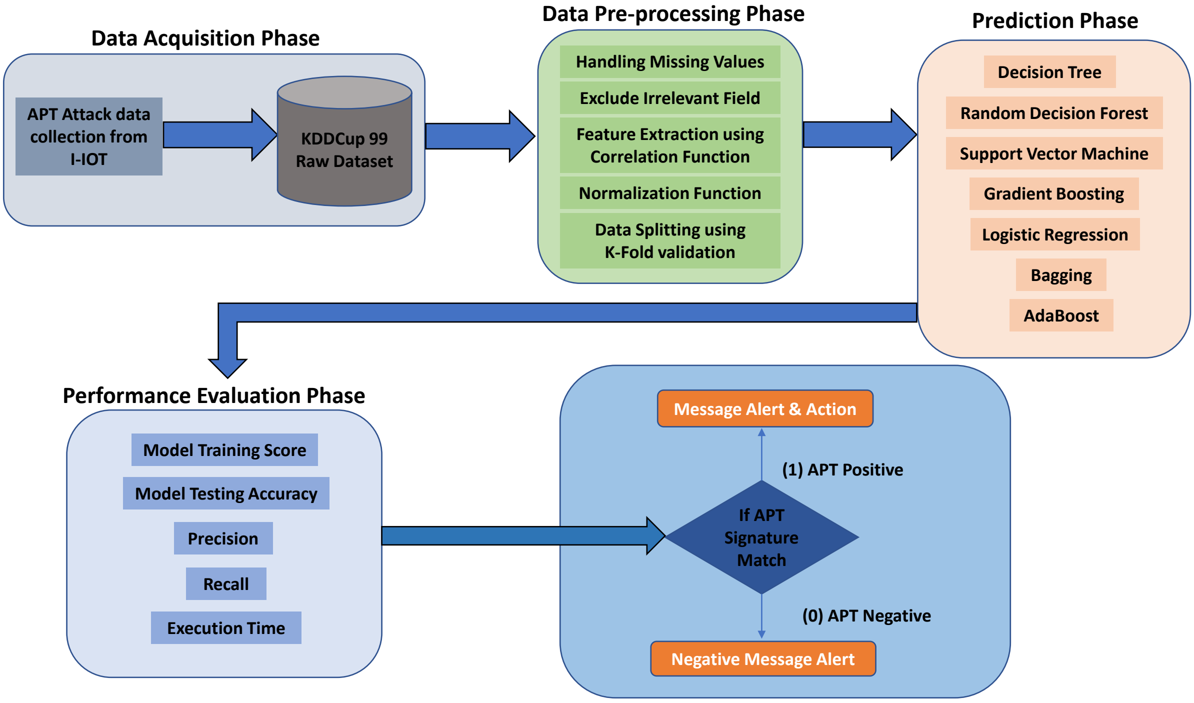


Figure 5. Block diagram of the proposed system.

Table 2. Attack classes and types.

Attack Class	Attack Type	Mapping to APT Phases
Normal	Data without intrusion	Serialized TCP/IP network Packets with no intrusion.
Probing	Satan, Mscan, Portsweep, Nmap	Phase-1: Reconnaissance.
DDOS	Back, Land, Neptune, Pod, Smurf, Teardrop, UDPstrom, Worm	Phase-2: Weaponization
R2L	FTP Write, Guess Password, Imap, Phf, Spy, Xlock, Sendmail, Named, Xsnoop, Waremaster, Multihop	Phase-3: Delivery, Phase-4: Exploitation and Phase-5: Installation.
U2R	Buffer overflow, Loadmodule, Rootkit, Perl,Sqlattack, Xterm	Phase-6: Command and Control and Phase-7: Actions on Intent.

4.2. Data Pre-Processing Phase

Data pre-processing of the dataset is an imperative phase that leads to cleaning the data and making it more meaningful for the classification model to get better accuracy. The following steps were carried out in the data pre-processing phase:

4.2.1. Resolving Missing Value

Analysis of dataset for missing values is a first step towards pre-processing. No missing values were observed in the dataset.

4.2.2. Exclusion of Irrelevant Fields

The only relevant target attributes of APT are selected. Those attributes that do not contribute to the prediction of the attack are excluded by applying the Principle Component Analysis (PCA) technique. The attributes that contribute to the prediction of APT signatures are illustrated in Table 3.

Table 3. Feature Selection and Extraction.

Selected Attributes	Description	Class Level	APT Attack Phase
Normal	Normal data	Data without intrusion	Serialized TCP/IP network packets with no intrusion
Ipsweep	ICMP sweep echo requests to multiple destination addresses	Probe	APT Phase-1: Reconnaissance
Nmap	Network Reconnaissance	Probe	APT Phase-1
Portsweep	ICMP packets to port	Probe	APT Phase-1
Satan	Vulnerability testing of TCP/IP hosts using common TCP/IP protocols	Probe	APT Phase-1
Back	Shut down a machine or network so intended users cannot access	DDos	APT Phase-2: Weaponization and Point of Entry
Smurf	A type of distributed denial of service (DDoS) attack that renders computer networks inoperable	DDos	APT Phase-2
Teardrop	Sends fragmented packets to a target machine intending to crash the target network device	DDos	APT Phase-2.
Neptune	An SYN Flood attack that, exploits the flaws in handshake TCP protocol by sending a large no. of spoofed SYN packets directed the targeted machine	Dos	APT Phase-2
Land	A layer 4 DoS attack which crashes target machine crash due to packet processing in repeatition by TCP stack	Dos	APT Phase-2
Ftp_write	Write on the remote machine or delete any files	Remote to Local (R2L)	APT Phase-3: Delivery, Phase 4:Exploitation and Phase 5: Installation
Guess passwd	Authenticate a particular user	R2L	APT Phase-3,-4,-5
Imap	IMAP for password-spray attacks	R2L	APT Phase-3,-4,-5
Multihop	Identification of malicious traffic source by tracing through several proxies	R2L	APT Phase-3,-4,-5
Phf	A remote to local (R2L) attack against Web Server	R2L	APT Phase-3,-4,-5
Spy	Malware installed on systems without the end user's information	R2L	APT Phase-3,-4,-5
Warezclient (WC)	Exploit the vulnerabilities of FTP. An implementation of FTP-based R2L attacks	R2L	APT Phase-3,-4,-5
Warezmaster (WM)	Exploit the vulnerabilities of FTP. An implementation of FTP-based R2L attacks	R2L	APT Phase-3,-4,-5
Buffer Overflow	System anomaly that overwrites adjoining memory blocks	User to Local (U2L)	Phase-6: Command and Control (C& C), Phase-7: Actions on Intent and Data Exfiltration
Loadmodule	Load dynamically loadable kernel driver	U2L	Phase-6, Phase-7
Perl	A vulnerability that uses a perl script to gain root access	U2L	Phase-6 and Phase-7
Rootkit	Rootkit malicious packets	U2L	Phase-6 and Phase-7

4.2.3. Feature Selection and Extraction

The Pearson correlation coefficient (PCC) technique is used for feature extraction because the study shows [36] that it consistently generates the best results among different sample sizes with an area under the receiver operating characteristics curve (AUC), to be as high as 0.975. The AUC is the best measurement which is independent of the precise threshold. Figure 6 shows that the PCC technique outperforms the PCA technique for the anomaly based dataset. PCA is a data analysis descriptive and adaptive tool used for dimensionality reduction and widely used in describing the variation and covariance arrangement of a set of variables through linear permutation.

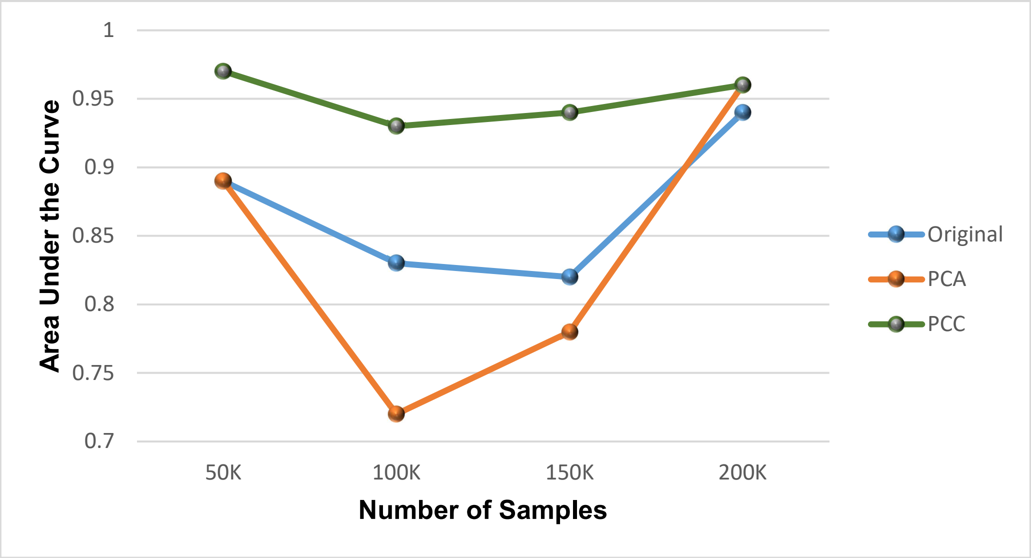


Figure 6. Comparison of PCC with PCA technique.

4.2.4. Data Normalization

As a pre-processing procedure, the dataset is statistically analyzed to check the normality of the dataset being used by the classifiers. It ensures high prediction results for deploying the ML algorithms. It also helps in noise reduction and outlier removal. In this step, the dataset is checked through a series of features, including data having an asymmetric bell shape. The mean and median are both equivalent and centrally distributed. About 68% of data lies within 1 Standard Deviation (SD). As the dataset fulfills all features, the dataset is normally distributed. Figure 7 illustrates the covariance matrix plot of chosen attributes. The covariance matrix is an efficient method for selecting features and reducing dimensions [37].

4.2.5. Bi-Variate Data Visualization

The t-Distributed Stochastic Neighbor Embedding (tSNE) plot is used to understand high dimensional data and project it into low dimensional space like 2D. It is a non-linear technique for reducing dimensionality suited for visualization of elevated dimension datasets [38]. Figures 8 and 9 describe the bivariate analysis of the KDDCup99 dataset with two different perplexities and levels of iteration. For the dataset, both graphs illustrate that the dataset is not linearly separable.

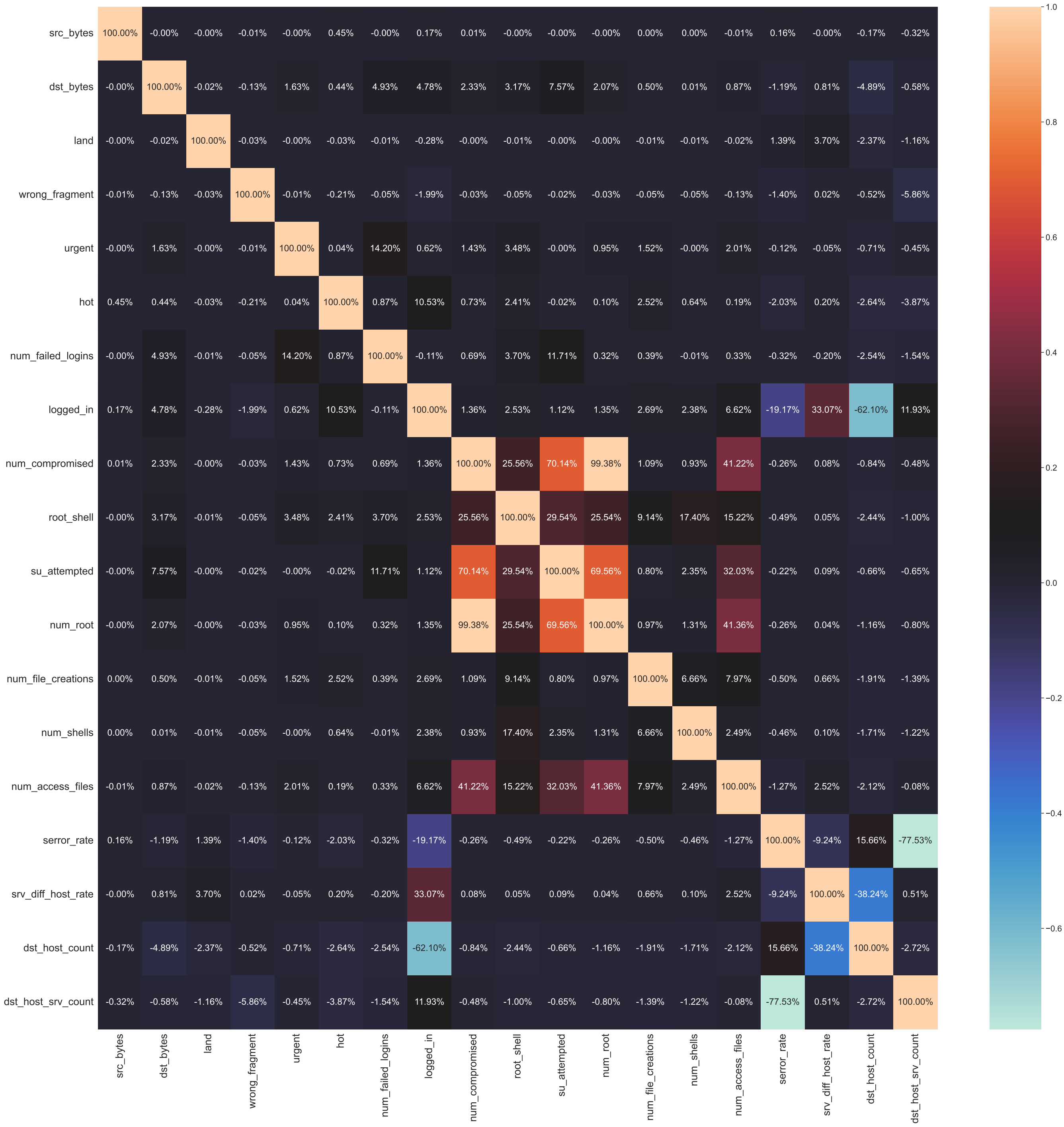
4.2.6. Dataset Splitting

This phase of the dataset involves the splitting of the dataset into training and testing data. The train–test split approximates the performance of ML algorithms used to predict the data attributes not used to train the model. The K-fold validation method is applied to split the data into test and train parts. At first, the dataset is shuffled to make the inputs and outputs completely random. This is done to ensure that the inputs are not biased.

4.3. Prediction Phase

After the pre-processing phase, the dataset is ready to train and test the classification models. In this work, eight classification models are used for APT prediction, including

Decision Tree, Random Forest, Support Vector Machine, Logistic Regression, Gaussian Naive Bayes, Bagging, XGBoost and Adaboost.



	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	num_failed_logins	logged_in	num_compromised	root_shell	su_attempted	num_root	num_file_creations	num_shells	num_access_files	serror_rate	srv_diff_host_rate	dst_host_count	dst_host_srv_count
src_bytes	100.00%	-0.00%	-0.00%	-0.01%	-0.00%	0.45%	-0.00%	0.17%	0.01%	-0.00%	-0.00%	-0.00%	0.00%	0.00%	-0.01%	0.16%	-0.00%	-0.17%	-0.32%
dst_bytes	-0.00%	100.00%	-0.02%	-0.13%	1.63%	0.44%	4.93%	4.78%	2.33%	3.17%	7.57%	2.07%	0.50%	0.01%	0.87%	-1.19%	0.81%	-4.89%	-0.58%
land	-0.00%	-0.02%	100.00%	-0.03%	-0.00%	-0.03%	-0.01%	-0.28%	-0.00%	-0.01%	-0.00%	-0.00%	-0.01%	-0.01%	-0.02%	1.39%	3.70%	-2.37%	-1.16%
wrong_fragment	-0.01%	-0.13%	-0.03%	100.00%	-0.01%	-0.21%	-0.05%	-1.99%	-0.03%	-0.05%	-0.02%	-0.03%	-0.05%	-0.05%	-0.13%	-1.40%	0.02%	-0.52%	-5.86%
urgent	-0.00%	1.63%	-0.00%	-0.01%	100.00%	0.04%	14.20%	0.62%	1.43%	3.48%	-0.00%	0.95%	1.52%	-0.00%	2.01%	-0.12%	-0.05%	-0.71%	-0.45%
hot	0.45%	0.44%	-0.03%	-0.21%	0.04%	100.00%	0.87%	10.53%	0.73%	2.41%	-0.02%	0.10%	2.52%	0.64%	0.19%	-2.03%	0.20%	-2.64%	-3.87%
num_failed_logins	-0.00%	4.93%	-0.01%	-0.05%	14.20%	0.87%	100.00%	-0.11%	0.69%	3.70%	11.71%	0.32%	0.39%	-0.01%	0.33%	-0.32%	-0.20%	-2.54%	-1.54%
logged_in	0.17%	4.78%	-0.28%	-1.99%	0.62%	10.53%	-0.11%	100.00%	1.36%	2.53%	1.12%	1.35%	2.69%	2.38%	6.62%	-19.17%	33.07%	-62.10%	11.93%
num_compromised	0.01%	2.33%	-0.00%	-0.03%	1.43%	0.73%	0.69%	1.36%	100.00%	25.56%	70.14%	99.38%	1.09%	0.93%	41.22%	-0.26%	0.08%	-0.84%	-0.48%
root_shell	-0.00%	3.17%	-0.01%	-0.05%	3.48%	2.41%	3.70%	2.53%	25.56%	100.00%	29.54%	25.54%	9.14%	17.40%	15.22%	-0.49%	0.05%	-2.44%	-1.00%
su_attempted	-0.00%	7.57%	-0.00%	-0.02%	-0.00%	-0.02%	11.71%	1.12%	70.14%	29.54%	100.00%	69.56%	0.80%	2.35%	32.03%	-0.22%	0.09%	-0.66%	-0.65%
num_root	-0.00%	2.07%	-0.00%	-0.03%	0.95%	0.10%	0.32%	1.35%	99.38%	25.54%	69.56%	100.00%	0.97%	1.31%	41.36%	-0.26%	0.04%	-1.16%	-0.80%
num_file_creations	0.00%	0.50%	-0.01%	-0.05%	1.52%	2.52%	0.39%	2.69%	1.09%	9.14%	0.80%	0.97%	100.00%	6.66%	7.97%	-0.50%	0.66%	-1.91%	-1.39%
num_shells	0.00%	0.01%	-0.01%	-0.05%	-0.00%	0.64%	-0.01%	2.38%	0.93%	17.40%	2.35%	1.31%	6.66%	100.00%	2.49%	-0.46%	0.10%	-1.71%	-1.22%
num_access_files	-0.01%	0.87%	-0.02%	-0.13%	2.01%	0.19%	0.33%	6.62%	41.22%	15.22%	32.03%	41.36%	7.97%	2.49%	100.00%	-1.27%	2.52%	-2.12%	-0.08%
serror_rate	0.16%	-1.19%	1.39%	-1.40%	-0.12%	-2.03%	-0.32%	-19.17%	-0.26%	-0.49%	-0.22%	-0.26%	-0.50%	-0.46%	-1.27%	100.00%	-9.24%	15.66%	-77.53%
srv_diff_host_rate	-0.00%	0.81%	3.70%	0.02%	-0.05%	0.20%	-0.20%	33.07%	0.08%	0.05%	0.09%	0.04%	0.66%	0.10%	2.52%	-9.24%	100.00%	-38.24%	0.51%
dst_host_count	-0.17%	-4.89%	-2.37%	-0.52%	-0.71%	-2.64%	-2.54%	-62.10%	-0.84%	-2.44%	-0.66%	-1.16%	-1.91%	-1.71%	-2.12%	15.66%	-38.24%	100.00%	-2.72%
dst_host_srv_count	-0.32%	-0.58%	-1.16%	-5.86%	-0.45%	-3.87%	-1.54%	11.93%	-0.48%	-1.00%	-0.65%	-0.80%	-1.39%	-1.22%	-0.08%	-77.53%	0.51%	-2.72%	100.00%

Figure 7. Selected features covariance matrix.

4.4. Performance Evaluation Phase

This phase involves the performance evaluation of the classifiers based on different performance measures including model training score, accuracy, precision, recall, F-measure and model execution time.

4.4.1. Model Training Score

Model training score demonstrates how perfectly and accurately training data fits the model. This phase is carried out on training data before the execution of the model on testing data. The training score closer to 1 yields the best-fit model.

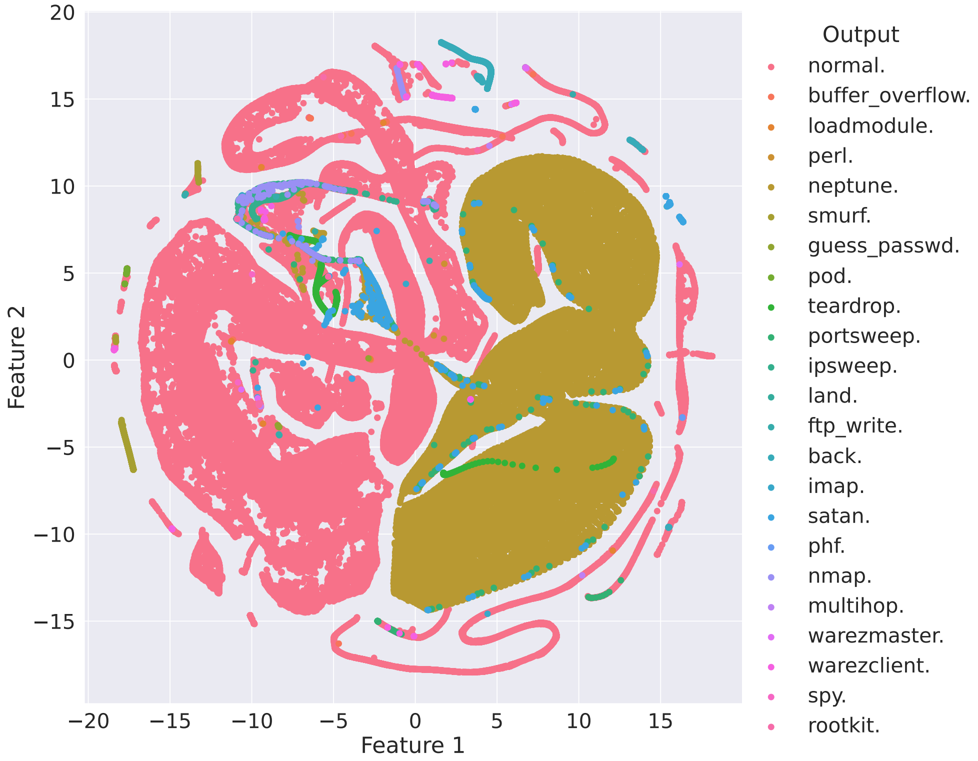


Figure 8. t-SNE plot with 100 perplexity and 500 iteration levels.

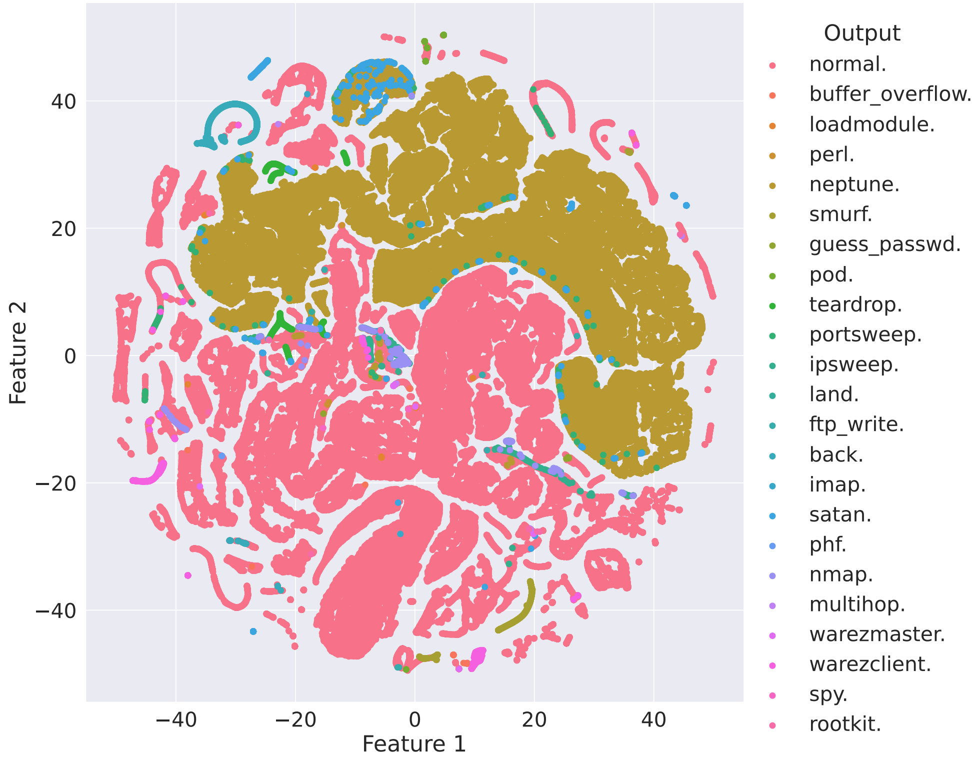


Figure 9. t-SNE plot with 50 perplexity and 1000 iteration levels.

4.4.2. Accuracy

Accuracy is a benchmark to assess the classification models. It measures the fraction of predictions that have been accurately classified. The accuracy of the model is calculated according to Equation (1) [39].

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

4.4.3. Precision, Recall and F1-Score

Precision is defined as the benchmark that measures the number of a correct optimistic predictions made that calculate the accuracy for the minority class. Equation (2) defines the precision [40].

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

The outcome is a value between 0.0 for no precision and 1.0 for full or ideal precision. The recall is a benchmark that quantifies the number of accurate optimistic predictions made from all positive predictions that could have been made. Equation (3) defines the recall [40].

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

The outcome is a value between 0.0 for no recall and 1.0 for full or ideal recall.

F-measure or F1 score combines precision and recalls into a sole measure that inculcates both properties. It is the harmonic mean of the two fractions and is illustrated in Equation (4) [40].

$$\text{F1 Measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

5. Experimental Analysis

For experimental analysis, Scikit learns python library is used along with GridSearchCV (cross-validated grid-search over a parameter grid) for determining optimal hyper-parameters from a given range for a particular machine learning model. Simulation is performed on a machine with Intel 1.70 GHz Core i7 CPU and 8 GB of RAM. Several ML classification algorithms were used to classify the data into the APT or normal type of analogous traffic. The following subsections explain the performance of classifiers.

5.1. Decision Tree

This is a supervised learning algorithm that is used for solving regression and classification problems [41]. The input parameters for the decision tree algorithm are listed in Table 4. Figure 10 shows the confusion matrix for the DT classifier. The labels on the *x*-axis show predicted results, whereas the *y*-axis refers to actual results. The DT model's average accuracy is 96.6%, with a 1.021 s execution time.

Table 4. Input Parameters for Decision Tree.

Hyper-Parameters	Value
max_depth	5, 10, 20, 50, 100, 500
Min_samples_split	5, 10, 100, 500
N_estimators	5, 10, 50, 100, 500
Criterion	gini
Splitter	best
Class_weight	Balanced
Cross validation (cv)	3
N_jobs	1

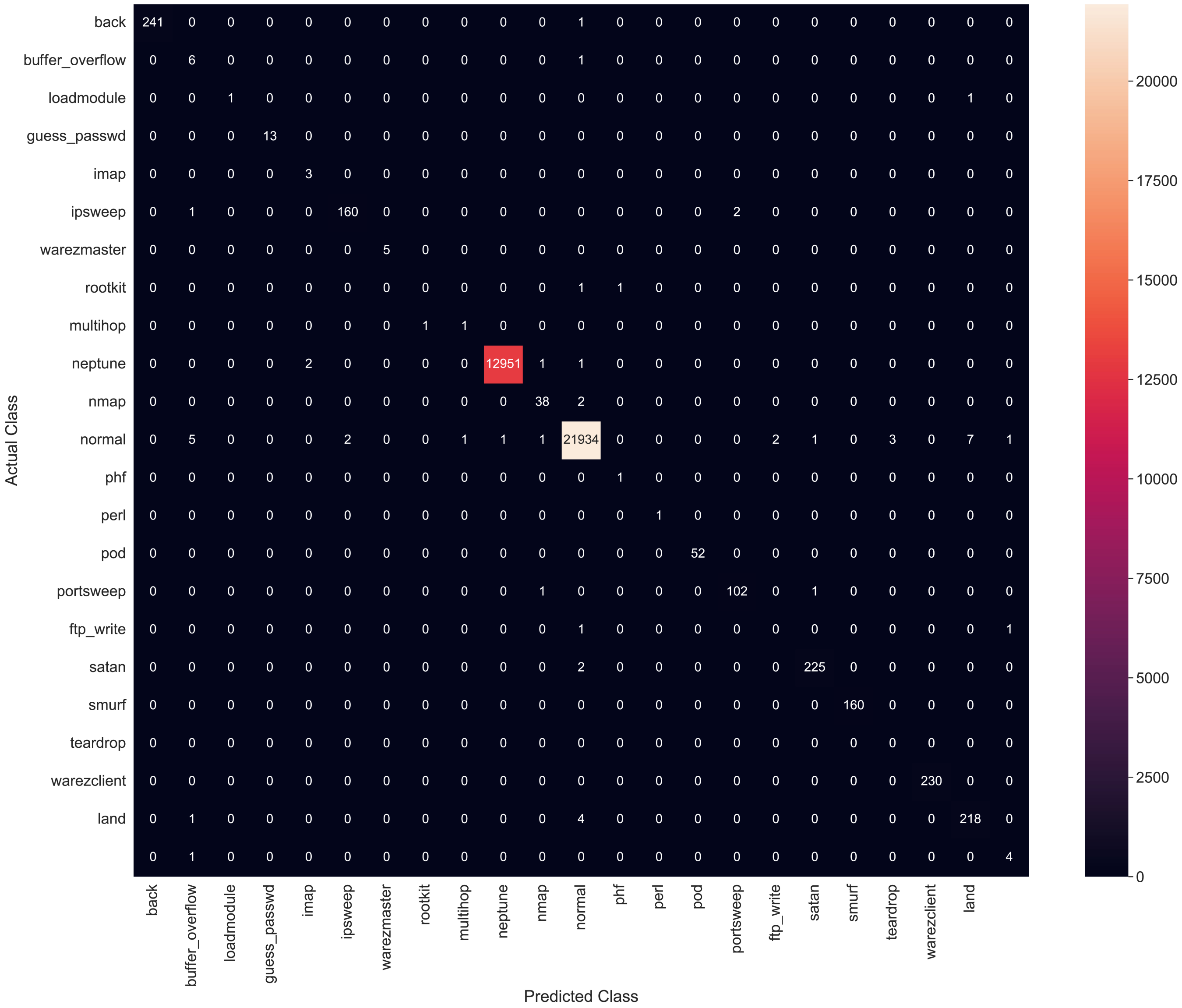


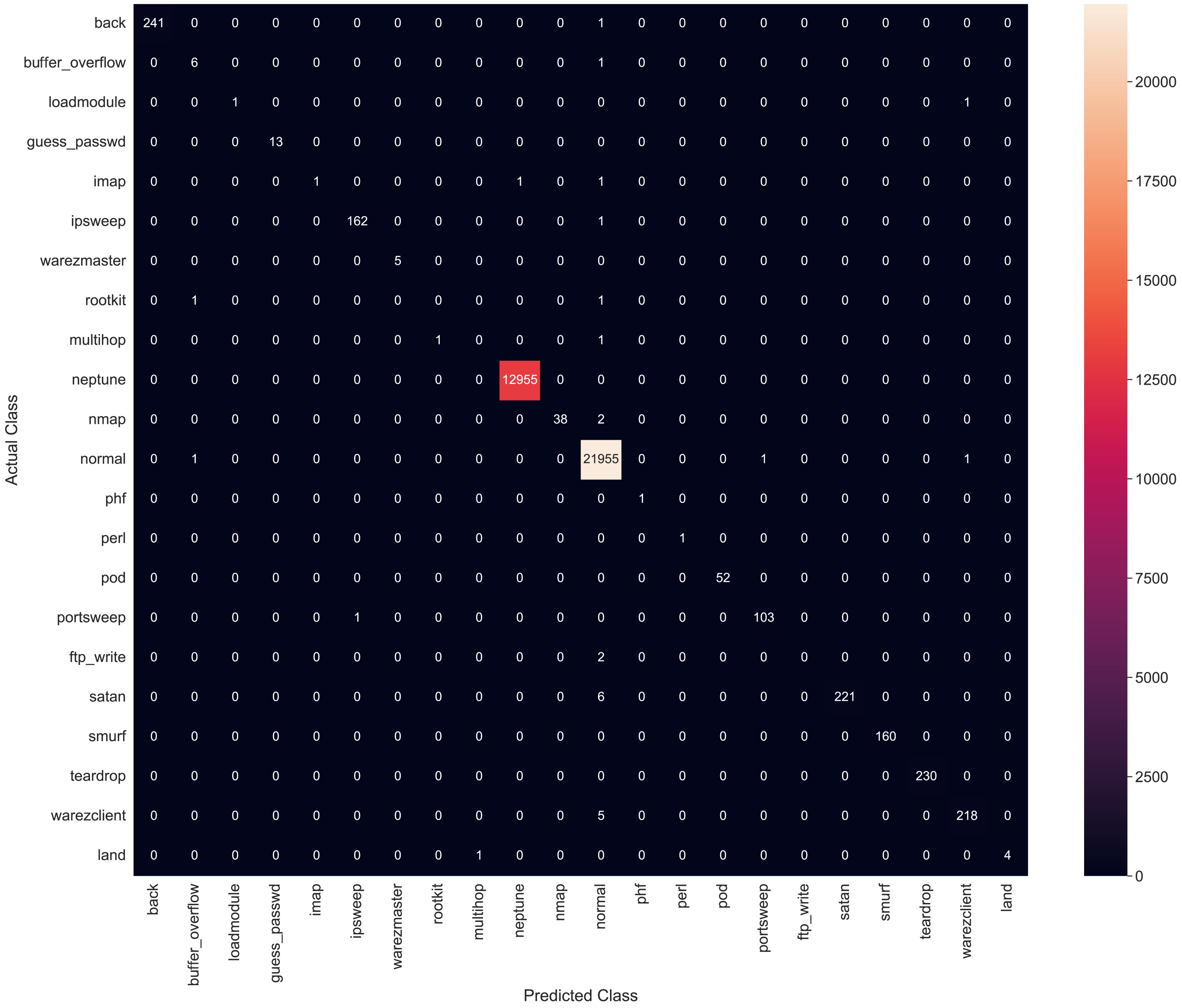
Figure 10. Confusion matrix for Decision Tree.

5.2. Random Forest

Random Forest is an ensemble learning algorithm that constructs numerous decision trees on various sets of features. It is widely used for classification and regression problems by constructing a multitude of DT during the training phase and generating results that are the mean prediction of each tree. They are well suited for the ensemble domain as they are fast and stable concerning the creation of synergy in the decision tree or forest. In this method, the technique uses trees that arbitrarily choose the attributes to divide the data [42]. Furthermore, this method has better accuracy than standard entropy reducing DT on many datasets in a large ensemble domain. The ensemble learning technique is specifically developed to aggregate the results of numerous decorrelated decision trees that are collected in a forest and classify the output results. RF is then trained using training data against the test dataset. Table 5 shows the input parameters for Random Forest classifier. After applying the classifiers, the experimental prediction accuracy of 98.5%, with an execution speed of 1.092 s, is achieved. Figure 11 shows the confusion matrix for RF.

Table 5. Input Parameters for Random Forest.

Hyper-Parameters	Value
max_depth	5, 10, 100, 500, 1000
Min_samples_split	5, 10, 100, 500
N_estimators	5, 10, 50, 100, 500
Criterion	gini
Class_weight	Balanced
Cross Validation (CV)	3
N_jobs	−1



	back	buffer_overflow	loadmodule	guess_passwd	imap	ipsweep	warezmaster	rootkit	multihop	neptune	nmap	normal	phf	perl	pod	portsweep	ftp_write	satan	smurf	teardrop	warezclient	land
back	241	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
buffer_overflow	0	6	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
loadmodule	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
guess_passwd	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
imap	0	0	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
ipsweep	0	0	0	0	0	162	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
warezmaster	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rootkit	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
multihop	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0
neptune	0	0	0	0	0	0	0	0	0	12955	0	0	0	0	0	0	0	0	0	0	0	0
nmap	0	0	0	0	0	0	0	0	0	0	38	2	0	0	0	0	0	0	0	0	0	0
normal	0	1	0	0	0	0	0	0	0	0	0	21955	0	0	0	1	0	0	0	0	1	0
phf	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
perl	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
pod	0	0	0	0	0	0	0	0	0	0	0	0	0	0	52	0	0	0	0	0	0	0
portsweep	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	103	0	0	0	0	0	0
ftp_write	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
satan	0	0	0	0	0	0	0	0	0	0	0	6	0	0	0	0	0	221	0	0	0	0
smurf	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	160	0	0	0
teardrop	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	230	0	0
warezclient	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	218	0
land	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	4

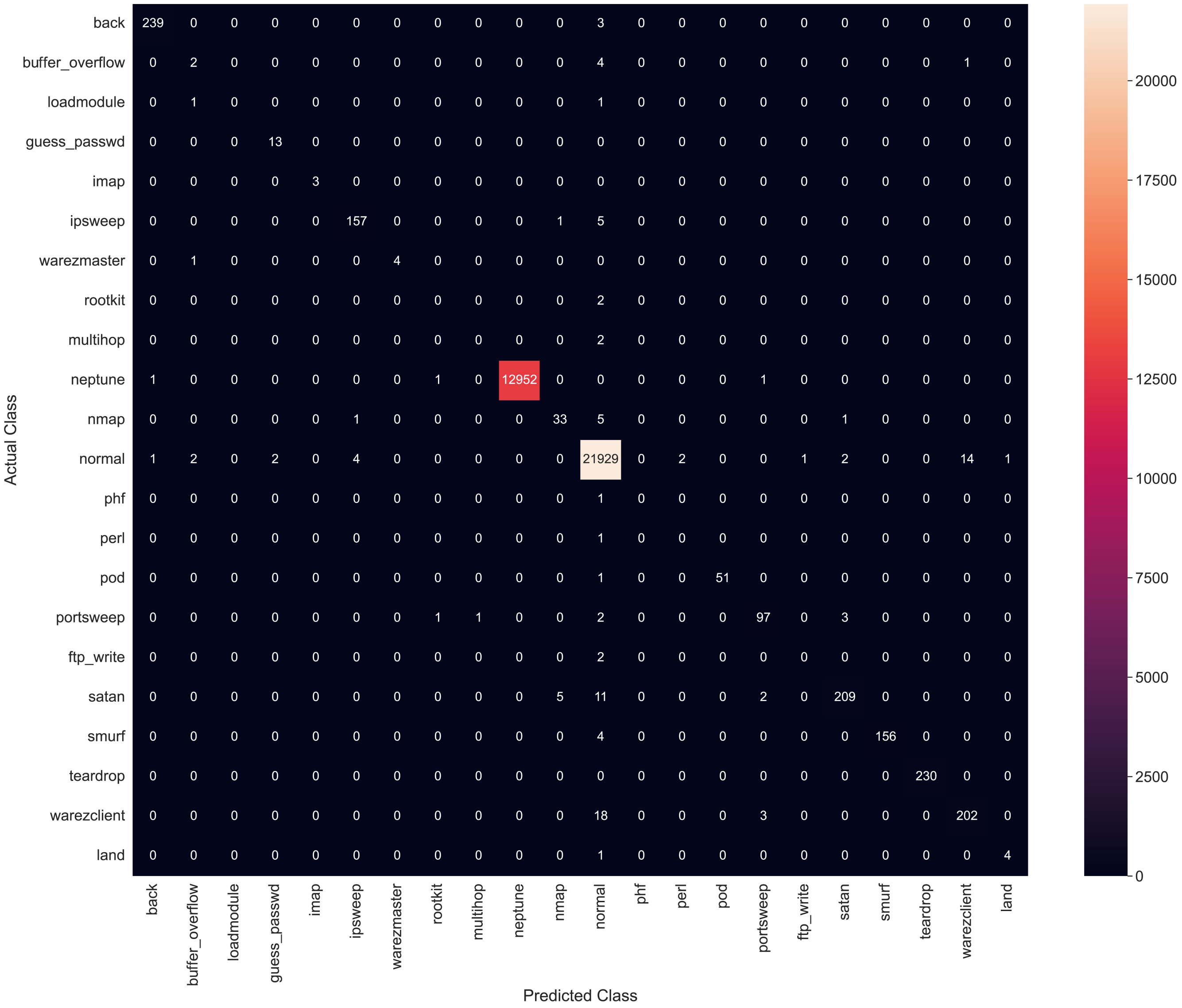
Figure 11. Confusion matrix for Random Forest.

5.3. Support Vector Machine

This is a linear supervised ML algorithm that analyzes high-dimensional and sparse data for pattern recognition, classification and regression [13,43]. SVM has been applied to classify heterogeneous data of the proposed dataset, as we consider our problem to be in line with multi-class pattern classification. Each data sample is classified into two categories, i.e., regular traffic and APT or malicious data. The model can solve linear and non-linear classification problems by creating a hyper-plane that segregates data into classes. The input parameters for SVM classifier are listed in Table 6. After applying the classifiers, the experimental prediction accuracy is 98.7%; an execution speed of 0.976 s is achieved. Figure 12 shows the confusion matrix of results by applying SVM classifiers.

Table 6. Input Parameters for SVM.

Hyper-Parameters	**Value**
Alpha	10^{-8}–10^{3}
penalty	L1, L2
loss	hinge
Cross validation (CV)	5
N_jobs	−1



	back	buffer_overflow	loadmodule	guess_passwd	imap	ipsweep	warezmaster	rootkit	multihop	neptune	nmap	normal	phf	perl	pod	portsweep	ftp_write	satan	smurf	teardrop	warezclient	land
back	239	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0
buffer_overflow	0	2	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	1	0
loadmodule	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
guess_passwd	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
imap	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ipsweep	0	0	0	0	0	157	0	0	0	0	1	5	0	0	0	0	0	0	0	0	0	0
warezmaster	0	1	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
rootkit	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
multihop	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
neptune	1	0	0	0	0	0	0	1	0	12952	0	0	0	0	0	1	0	0	0	0	0	0
nmap	0	0	0	0	0	1	0	0	0	0	33	5	0	0	0	0	0	1	0	0	0	0
normal	1	2	0	2	0	4	0	0	0	0	0	21929	0	2	0	0	1	2	0	0	14	1
phf	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
perl	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
pod	0	0	0	0	0	0	0	0	0	0	0	1	0	0	51	0	0	0	0	0	0	0
portsweep	0	0	0	0	0	0	0	1	1	0	0	2	0	0	0	97	0	3	0	0	0	0
ftp_write	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
satan	0	0	0	0	0	0	0	0	0	0	5	11	0	0	0	2	0	209	0	0	0	0
smurf	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	156	0	0	0
teardrop	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	230	0	0
warezclient	0	0	0	0	0	0	0	0	0	0	0	18	0	0	0	3	0	0	0	0	202	0
land	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	4

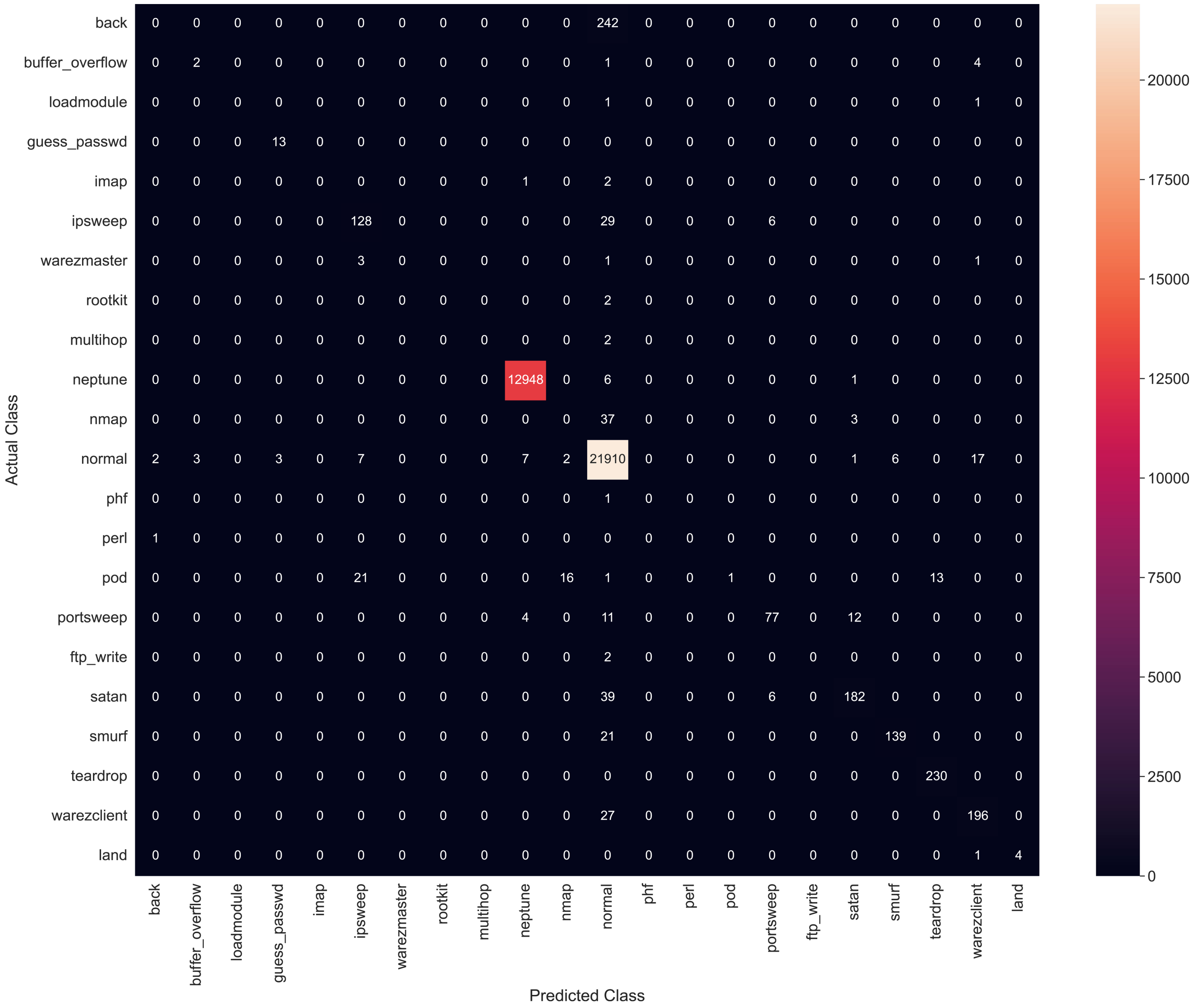
Figure 12. Confusion matrix for Support Vector Machine.

5.4. Logistic Regression

This is a supervised ML algorithm used for classification and regression problems [44]. The input parameters for the Logistic Regression algorithm are listed in Table 7. The average observed detection accuracy in LR is 98.5%, with an execution time of 2.954 s. Figure 13 shows the confusion matrix of results by applying LR classifiers.

Table 7. Input Parameters for Logistic Regression.

Hyper-Parameters	**Value**
Alpha	0.001, 0.01, 0.1, 1, 10, 20, 30
penalty	L1, L2
loss	log
Class_weight	Balanced
Cross validation (CV)	5
N_jobs	−1



	back	buffer_overflow	loadmodule	guess_passwd	imap	ipsweep	warezmaster	rootkit	multihop	neptune	nmap	normal	phf	perl	pod	portsweep	ftp_write	satan	smurf	teardrop	warezclient	land
back	0	0	0	0	0	0	0	0	0	0	0	242	0	0	0	0	0	0	0	0	0	0
buffer_overflow	0	2	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	4	0
loadmodule	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
guess_passwd	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
imap	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0
ipsweep	0	0	0	0	0	128	0	0	0	0	0	29	0	0	0	6	0	0	0	0	0	0
warezmaster	0	0	0	0	0	3	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
rootkit	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
multihop	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
neptune	0	0	0	0	0	0	0	0	0	12948	0	6	0	0	0	0	0	1	0	0	0	0
nmap	0	0	0	0	0	0	0	0	0	0	0	37	0	0	0	0	0	3	0	0	0	0
normal	2	3	0	3	0	7	0	0	0	7	2	21910	0	0	0	0	0	1	6	0	17	0
phf	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
perl	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
pod	0	0	0	0	0	21	0	0	0	0	16	1	0	0	1	0	0	0	0	13	0	0
portsweep	0	0	0	0	0	0	0	0	0	4	0	11	0	0	0	77	0	12	0	0	0	0
ftp_write	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0
satan	0	0	0	0	0	0	0	0	0	0	0	39	0	0	0	6	0	182	0	0	0	0
smurf	0	0	0	0	0	0	0	0	0	0	0	21	0	0	0	0	0	0	139	0	0	0
teardrop	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	230	0	0
warezclient	0	0	0	0	0	0	0	0	0	0	0	27	0	0	0	0	0	0	0	0	196	0
land	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4

Figure 13. Confusion matrix for Logistic Regression.

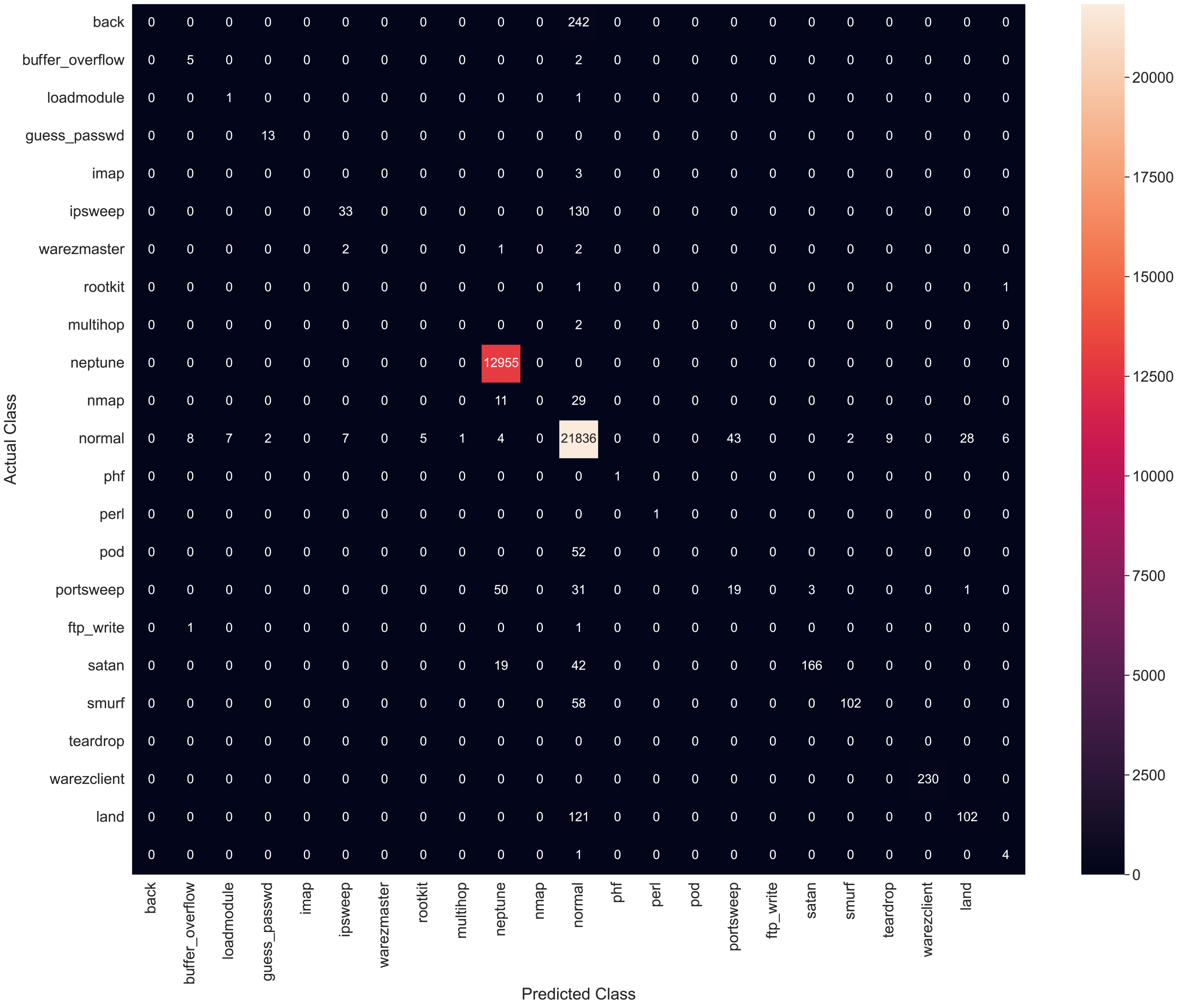
5.5. Gaussian Naive Bayes

This algorithm is based upon a probabilistic approach. A prior and posterior probability calculation of the classes in the dataset is involved in this type of ML algorithm [45]. The input parameters for the GNB classifier are listed in Table 8. The model accuracy of

GNB classifiers has been observed as 97.2%, with an execution time of 0.145 s. Figure 14 shows the confusion matrix of results by applying GNB classifiers.

Table 8. Input Parameters for GNB.

Hyper-Parameters	Value
Var_smoothing	10^{-9}–10^{3}
Cross validation (CV)	5
N_jobs	−1



	back	buffer_overflow	loadmodule	guess_passwd	imap	ipsweep	warezmaster	rootkit	multihop	neptune	nmap	normal	phf	perl	pod	portsweep	ftp_write	satan	smurf	teardrop	warezclient	land	
back	0	0	0	0	0	0	0	0	0	0	0	242	0	0	0	0	0	0	0	0	0	0	0
buffer_overflow	0	5	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
loadmodule	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
guess_passwd	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
imap	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0	0	0
ipsweep	0	0	0	0	0	33	0	0	0	0	0	130	0	0	0	0	0	0	0	0	0	0	0
warezmaster	0	0	0	0	0	2	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0
rootkit	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
multihop	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
neptune	0	0	0	0	0	0	0	0	0	12955	0	0	0	0	0	0	0	0	0	0	0	0	0
nmap	0	0	0	0	0	0	0	0	0	11	0	29	0	0	0	0	0	0	0	0	0	0	0
normal	0	8	7	2	0	7	0	5	1	4	0	21836	0	0	0	43	0	0	2	9	0	28	6
phf	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
perl	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
pod	0	0	0	0	0	0	0	0	0	0	0	52	0	0	0	0	0	0	0	0	0	0	0
portsweep	0	0	0	0	0	0	0	0	0	50	0	31	0	0	0	19	0	3	0	0	0	1	0
ftp_write	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
satan	0	0	0	0	0	0	0	0	0	19	0	42	0	0	0	0	0	166	0	0	0	0	0
smurf	0	0	0	0	0	0	0	0	0	0	0	58	0	0	0	0	0	0	102	0	0	0	0
teardrop	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
warezclient	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	230	0	0
land	0	0	0	0	0	0	0	0	0	0	0	121	0	0	0	0	0	0	0	0	0	102	0
	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	4

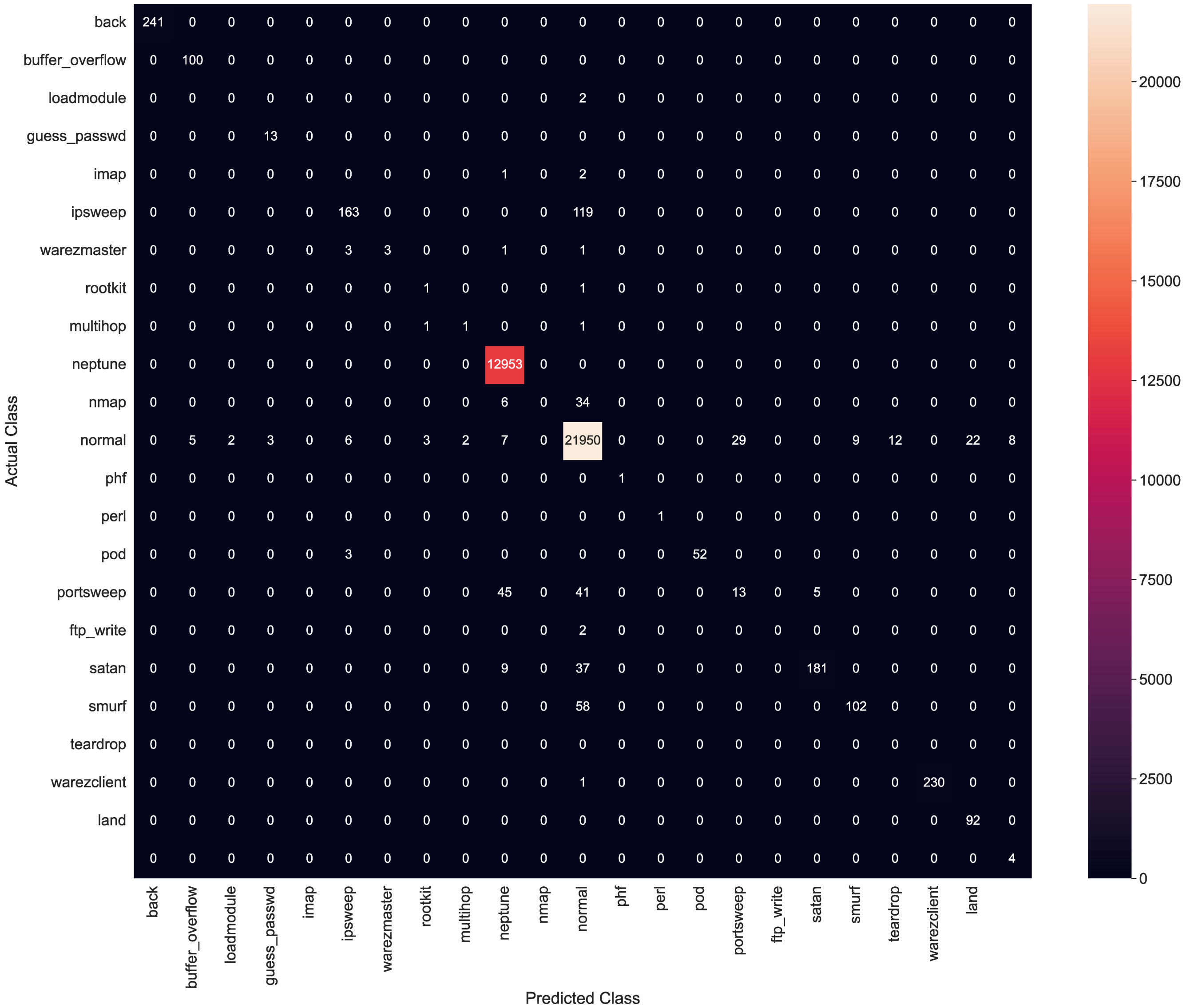
Figure 14. Confusion matrix for Gaussian Naive Bayes.

5.6. Bagging

Bagging is an ensemble meta-model to improve the stability and accuracy of ML algorithms that form the basis for prediction [46]. In this model, several independent models have been fitted. Bagging has minimal variance as a model, thereby reducing the inherent problem of overfitting [47]. The input parameters for the Bagging algorithm are listed in Table 9. The average observed detection accuracy in bagging is 97.5%, with an execution time of 1.092 s. Figure 15 shows the confusion matrix of results by applying Bagging classifiers.

Table 9. Input Parameters for Bagging.

Hyper-Parameters	Value
Var_smoothing	10^{-9}–10^{3}
Cross validation (CV)	5
N_jobs	None
max_samples	1
max_features	1



Actual Class	back	buffer_overflow	loadmodule	guess_passwd	imap	ipsweep	warezmaster	rootkit	multihop	neptune	nmap	normal	phf	perl	pod	portsweep	ftp_write	satan	smurf	teardrop	warezclient	land	
back	241	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
buffer_overflow	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
loadmodule	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
guess_passwd	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
imap	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0
ipsweep	0	0	0	0	0	163	0	0	0	0	0	119	0	0	0	0	0	0	0	0	0	0	0
warezmaster	0	0	0	0	0	3	3	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
rootkit	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
multihop	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0
neptune	0	0	0	0	0	0	0	0	0	12953	0	0	0	0	0	0	0	0	0	0	0	0	0
nmap	0	0	0	0	0	0	0	0	0	6	0	34	0	0	0	0	0	0	0	0	0	0	0
normal	0	5	2	3	0	6	0	3	2	7	0	21950	0	0	0	29	0	0	9	12	0	22	8
phf	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
perl	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
pod	0	0	0	0	0	3	0	0	0	0	0	0	0	0	52	0	0	0	0	0	0	0	0
portsweep	0	0	0	0	0	0	0	0	0	45	0	41	0	0	0	13	0	5	0	0	0	0	0
ftp_write	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
satan	0	0	0	0	0	0	0	0	0	9	0	37	0	0	0	0	0	181	0	0	0	0	0
smurf	0	0	0	0	0	0	0	0	0	0	0	58	0	0	0	0	0	0	102	0	0	0	0
teardrop	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
warezclient	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	230	0	0
land	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	92	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4

Figure 15. Confusion matrix for Bagging.

5.7. XGBoost

XGBoost is an ML classification technique that is often used in regression to generate a prediction model in the form of an ensemble for weaker prediction models like the decision. It generalizes weaker models by optimization fashion of arbitrary differentiable loss function [48]. The input parameters for the XGBoost algorithm are listed in Table 10.

The model accuracy of the XGBoost classifier was observed as 98.7%, with an execution time of 0.985 s. Figure 16 shows the confusion matrix of results for XGBoost.

Table 10. Input Parameters for XGBoost.

Hyper-Parameters	Value
max_depth	2, 3, 5, 7, 10
N_estimators	5, 10, 50, 100, 500
objective	Multi:softprob
Cross validation (CV)	3
N_jobs	−1

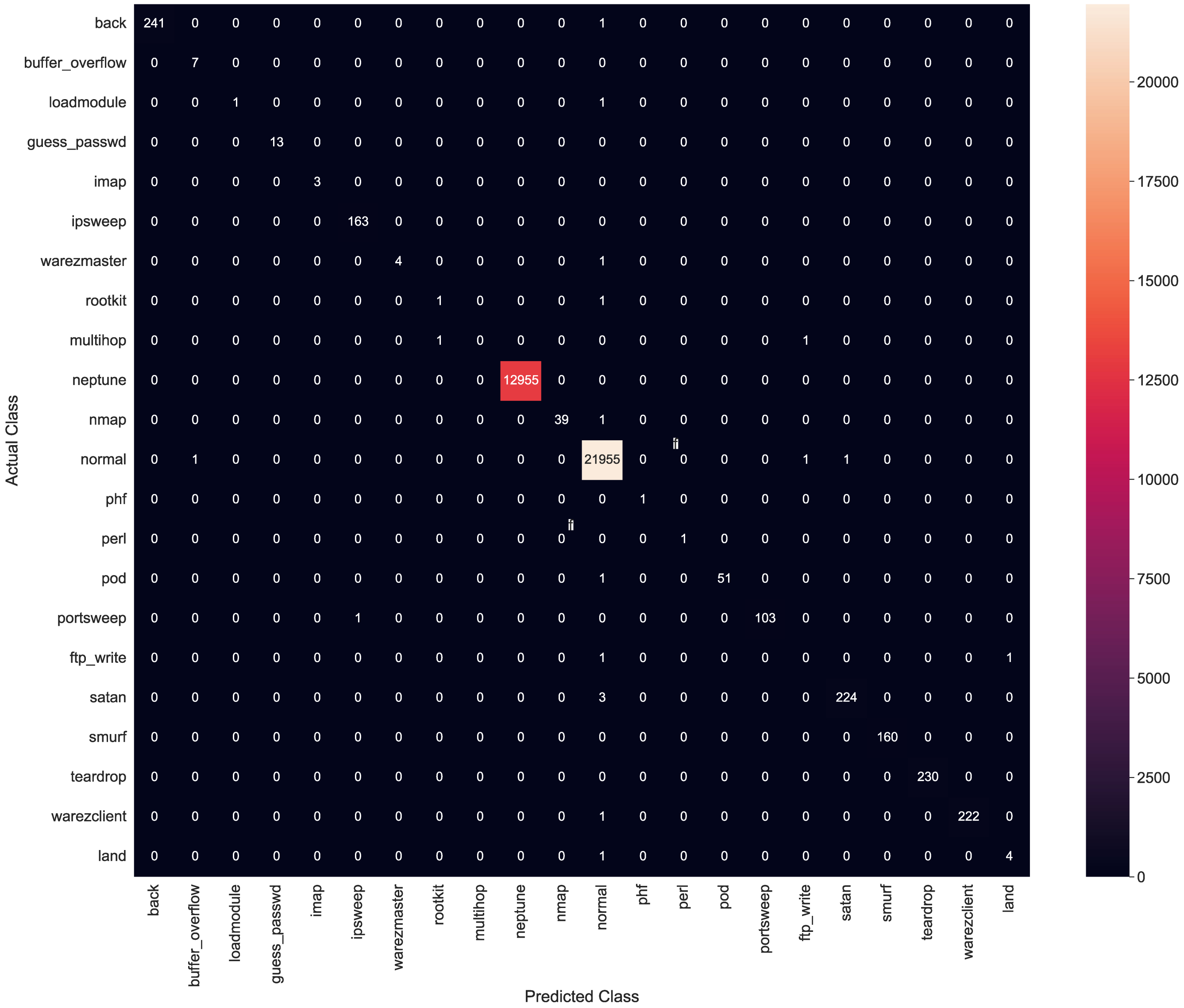


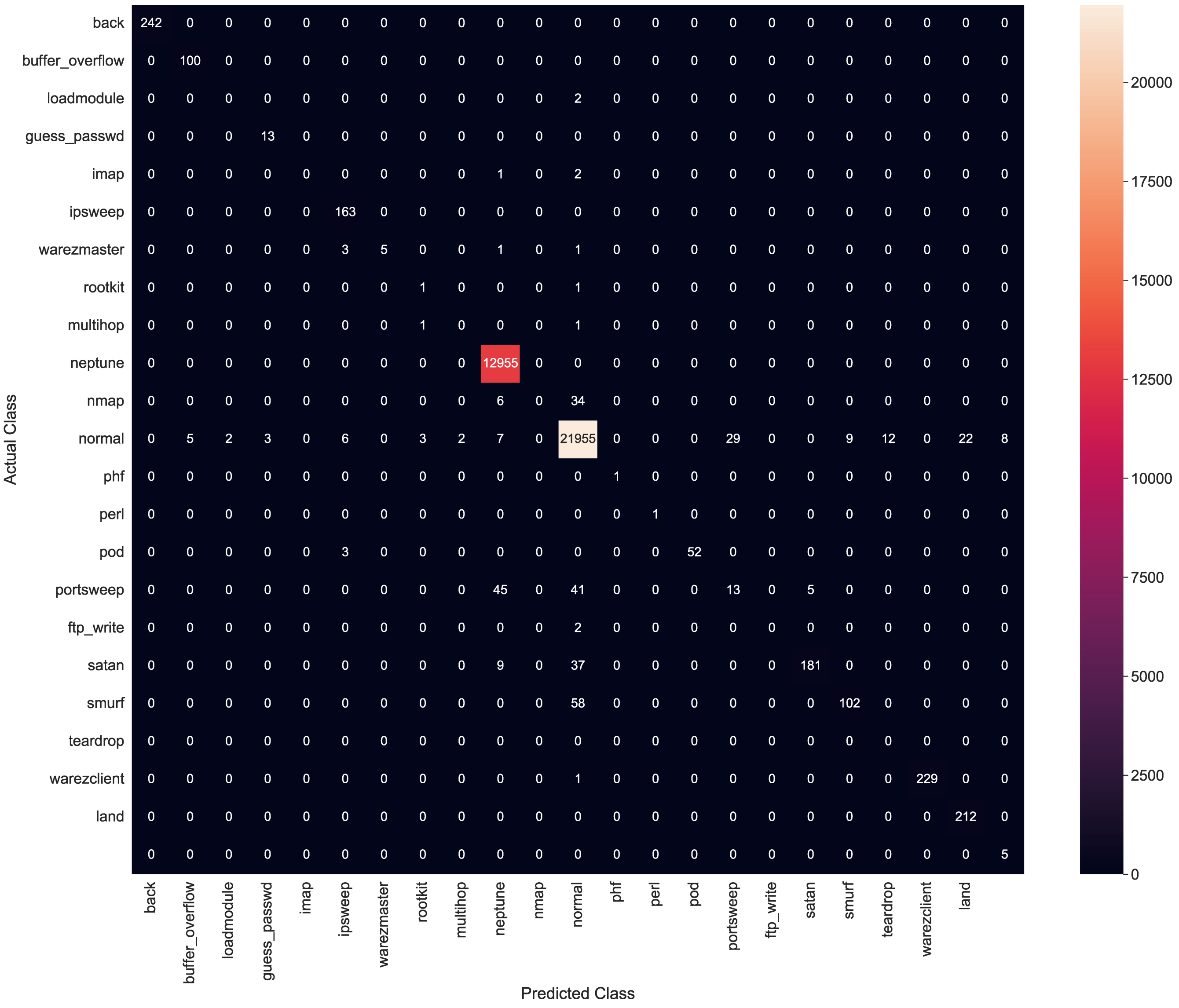
Figure 16. Confusion matrix for XGBoost.

5.8. Adaboost (An Ensemble Technique)

An ensemble technique is a compound model, which combines multiple low performing classifiers to develop an enhanced classifier. The Adaboost algorithm, as an ensemble approach to the classification models, also known as Adaptive boosting, is highly suited for classification purposes. It combines multiple weaker models to generate one stronger model. The input parameters for the Adaboost algorithm are listed in Table 11. The accuracy of the Adaboost classifier was observed as 99.9%, with an execution time of 0.012 s. Figure 17 shows the confusion matrix for AdaBoost classifiers.

Table 11. Input Parameters for Adaboost.

Hyper-Parameters	Value
algorithm	SAMME.R
Learning_rate	1.0
N_estimators	50
Cross validation (CV)	3



	back	buffer_overflow	loadmodule	guess_passwd	imap	ipsweep	warezmaster	rootkit	multihop	neptune	nmap	normal	phf	perl	pod	portsweep	ftp_write	satan	smurf	teardrop	warezclient	land	
back	242	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
buffer_overflow	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
loadmodule	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
guess_passwd	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
imap	0	0	0	0	0	0	0	0	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0
ipsweep	0	0	0	0	0	163	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
warezmaster	0	0	0	0	0	3	5	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0
rootkit	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
multihop	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
neptune	0	0	0	0	0	0	0	0	0	12955	0	0	0	0	0	0	0	0	0	0	0	0	0
nmap	0	0	0	0	0	0	0	0	0	6	0	34	0	0	0	0	0	0	0	0	0	0	0
normal	0	5	2	3	0	6	0	3	2	7	0	21955	0	0	0	29	0	0	9	12	0	22	8
phf	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
perl	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
pod	0	0	0	0	0	3	0	0	0	0	0	0	0	0	52	0	0	0	0	0	0	0	0
portsweep	0	0	0	0	0	0	0	0	0	45	0	41	0	0	0	13	0	5	0	0	0	0	0
ftp_write	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0
satan	0	0	0	0	0	0	0	0	0	9	0	37	0	0	0	0	0	181	0	0	0	0	0
smurf	0	0	0	0	0	0	0	0	0	0	0	58	0	0	0	0	0	0	102	0	0	0	0
teardrop	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
warezclient	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	229	0	0
land	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	212	0
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5

Figure 17. Confusion matrix for AdaBoost classifier.

5.9. Comparative Analysis

The experimental results for applied classification models in terms of evaluation parameters are listed in Table 12.

Table 12. Experimental Results.

Classification Models	Training Score	Accuracy	Recall	Precision	F1-Score	Execution Time	Error
Decision Tree	0.995	0.966	0.988	0.992	0.990	1.021	0.004
Random Forest	0.985	0.985	0.996	0.996	0.996	1.092	0.015
Support Vector Machine	0.987	0.987	0.973	0.946	0.958	0.976	0.013
Logistic Regression	0.984	0.985	0.975	0.960	0.966	2.954	0.015
Gaussian Naive Bayes	0.971	0.972	0.974	0.963	0.967	0.145	0.028
Bagging	0.954	0.975	0.973	0.994	0.995	1.092	0.025
Gradient Boost XGBoost	0.988	0.987	0.975	0.961	0.966	0.985	0.013
AdaBoost	0.999	0.999	1.000	0.999	1.000	0.012	0.001

The results show that Adaboost outperforms others with 99.9%, accuracy and execution time of 0.012 s. While the XGBoost classifier has also yielded promising results of 98.7%, its execution time is comparatively higher than Adaboost. Furthermore, other ML classifiers such as SVM, RF, LR and DT have also performed more or less similarly but not better than XGBoost and Adaboost. Figure 18 shows the comparison among different classifiers.

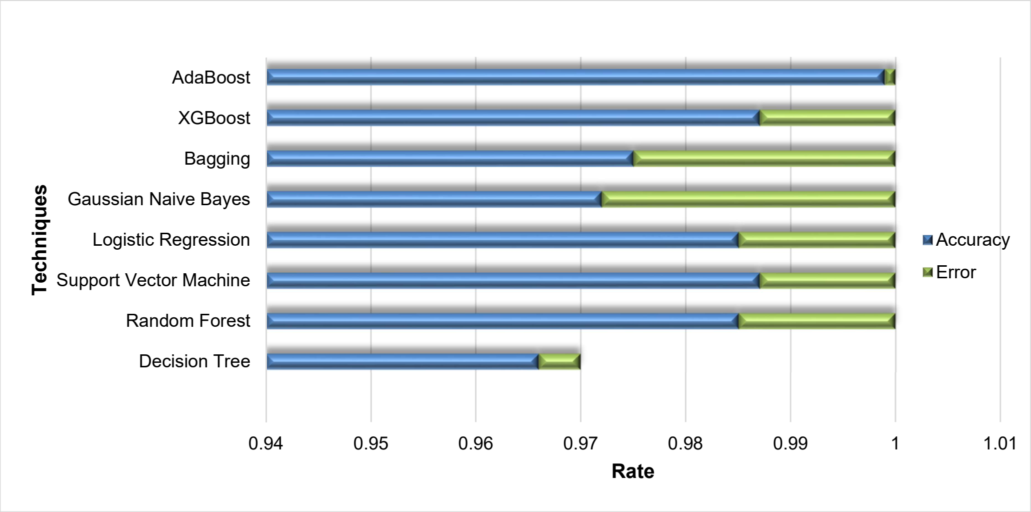


Figure 18. Classifier comparison based on accuracy.

Therefore, it can be stated that boosting algorithms can positively affect the ML algorithms' overall performance when classifying complex malware signatures, including the classification of APT data signatures. Table 13 states the comparative analysis of the proposed system with state of the art techniques for APT detection and classification. Results show the superiority of the proposed approach with respect to detection and classification accuracy. Various ML and autonomous approaches have been proposed by different researchers. While Brogi et al. [26] presented a non-ML autonomous approach providing accuracy of 100% TRP, the approach only covers four APT attack phases and that too on a simulated dataset. This approach may not provide an accurate and true representation of APT signatures when deployed on mission critical systems. Contrary to this and other approaches summarized in the table above, minimum computational power is required to process the anomaly data packets in the proposed system, facilitating the overall performance by minimizing comprehensive execution time and computation power consumed by the network that covers all APT attack phases.

Another main attribute of the proposed system is its scalability, as it is based on machine learning algorithms that make it highly scalable because of its capability to process huge amounts of data and perform massive computations in a cost- and time-effective way. The proposed system can be deployed on the I-IoT gateway to analyze the whole network traffic. This results in high scalability, productivity (faster deployment), modularity and portability.

Table 13. Comparison of proposed methodology with existing techniques.

Papers	Techniques	Dataset	Accuracy
Teyou et al. [23]	CNN	NSL-KDD	80.7% with 7 APT Life cycle phase.
Latif et al. [24]	RaNN	DS2OS	99.20% with 7 APT attack phases.
Wang et al. [25]	Autonomous approach	Simulated Dataset	83.3% TPR with 1 APT attack phase.
Brogi et al. [26]	Autonomous approach	Simulated Dataset	100% TPR with 4 APT attack phase.
Cheng et al. [27]	Autonomous approach	Simulated Dataset	83.7% TPR with 2 APT attack phase.
Parra et al. [28]	DL	NBaIoT	94.30% TPR with 7 APT attack phase.
Zheng et al. [29]	Linear discriminant analysis (LDA)	NSL-KDD	92.35% with 7 APT attack phase.
Khalvati et al. [30]	SVM and Bayesian	KDDcup99	91.50% with 7 APT attack phase.
Ghafir et al. [31]	Autonomous approach	Correlation clustered dataset	84.4%
Shudong et al. [32]	ML	Behavioural data	80.0%
Proposed System	NB, DT, RF, SVM, LR, Bagging,XGBoost and AdaBoost	KDDCup99	99.9% (Adaptive Boost) accuracy and reduced detection time

6. Conclusions and Future Directions

APT is a highly complex and enigmatic attack that targets traditional and enterprise networks such as I-IoT. Such types of attack must be detected at the earliest possible moment, as I-IoT networks are highly sophisticated and an attack on such a network can lead to a catastrophic disaster. This paper presented an intelligent APT detection and classification system for securing I-IoT. After pre-processing, several machine learning algorithms were applied to detect and classify malicious I-IoT nodes prone to APT attacks. To develop a system, a publicly available dataset KDDCup99 is used. The experimental results show that the Adaboost algorithm outperforms the rest of the applied machine learning algorithms by showing 99.9% accuracy, 100% model training score, 99.9% precision, 100% recall and 0.012 s execution time. Moreover, the comparative analysis of the proposed system with state-of-the-art systems depicts its superiority.

In future, the proposed system can be deployed on the I-IoT gateway for real-time traffic analysis of I-IoT networks, including smart cities, smart power plants, smart industries, smart communication systems, etc. The I-IoT gateway is directly connected to the Distributed Control System (DCS) of the I-IoT networks. Whenever a malicious packet is sent into the node that interfaces with the DCS, the system can detect APT traffic signatures and other attack vectors from the regular network traffic that help take the necessary actions to minimize the damage. Furthermore, the arrangement of such methodology may be expanded into other domains, besides I-IoT network, such as SCADA systems to determine new possibilities and limitations imposed by different kinds of datasets.

Author Contributions: Conceptualization, S.H.J.; methodology, S.H.J. and M.B.A.; software, S.H.J. and M.A.; validation, M.B.A. and M.A.; formal analysis, S.H.J. and M.B.A.; investigation, M.A.A.G.; resources, S.H.A.; data curation, K.M. and S.H.A.; writing—original draft preparation, S.H.J.; writing—review and editing, M.A.; visualization, K.M.; supervision, M.B.A.; project administration, M.A.A.G.; funding acquisition, M.A.A.G. All authors have read and agreed to the published version of the manuscript.

Funding: This research work is supported by Data and Artificial Intelligence Scientific Chair at Umm Al-Qura University, Makkah City, Saudi Arabia.

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

References

1. Sisinni, E.; Saifullah, A.; Han, S.; Jennehag, U.; Gidlund, M. Industrial internet of things: Challenges, opportunities and directions. *IEEE Trans. Ind. Inform.* **2018**, *14*, 4724–4734. [CrossRef]
2. Cheng, J.; Chen, W.; Tao, F.; Lin, C. Industrial iot in 5g environment towards smart manufacturing. *J. Ind. Inf. Integr.* **2018**, *10*, 10–19. [CrossRef]
3. Lund, D.; MacGillivray, C.; Turner, V.; Morales, M. Worldwide and regional internet of things (iot) 2014–2020 forecast: A virtuous circle of proven value and demand. *Int. Data Corp. (IDC) Tech. Rep.* **2014**, *1*, 9. [CrossRef]
4. Greengard, S. The worsening state of ransomware. *Commun. ACM* **2021**, *64*, 15–17. [CrossRef]
5. Forestiero, A. Metaheuristic algorithm for anomaly detection in Internet of Things leveraging on a neural-driven multiagent system. *Knowl.-Based Syst.* **2021**, *228*, 107241. [CrossRef]
6. Forestiero, A. Self-organizing anomaly detection in data streams. *Inf. Sci.* **2016**, *373*, 321–326. [CrossRef]
7. Ali, B.; Awad, A. Cyber and physical security vulnerability assessment for iot based smart homes. *Sensors* **2018**, *18*, 817. [CrossRef]
8. Saad, M.; Ahmad, M.B.; Asif, M.; Masood, K.; Alghamdi, M.A. Social Distancing and Isolation Management Using Machine-to-Machine Technologies to Prevent Pandemics. *Comput. Mater. Contin. (CMC) J.* **2021**, *67*, 3545–3562. [CrossRef]
9. Thomson, K.; Von Solms, R.; Louw, L. Cultivating an organizational information security culture. *Comput. Fraud Secur.* **2006**, *2006*, 7–11. [CrossRef]
10. Pedreira, V.; Barros, D.; Pinto, P. A Review of Attacks, Vulnerabilities,and Defenses in Industry 4.0 with New Challenges on Data Sovereignty Ahead. *Sensors* **2021**, *21*, 5189. [CrossRef]
11. Ahmad, M.B.; Akram, A.; Asif, M.; Rehman, S.U. Using Genetic Algorithm to Minimize False Alarms in Insider Threats Detection of Information Misuse in Windows Environment. *Math. Probl. Eng. J.* **2014**, *2014*, 12.
12. Hajda, J.; Jakuszewski, R.; Ogonowski, S. Security Challenges in Industry 4.0 PLC Systems. *Appl. Sci.* **2021**, *11*, 9785. [CrossRef]
13. Oueslati, E.; Mrabet, H.; Jemai, A.; Alhomoud, A. Comparative Study of the Common Cyber-physical Attacks in Industry 4.0. In Proceedings of the International Conference on IoT, Embedded Systems and Communications (IINTEC), Tunis, Tunisia, 20–22 December 2019; pp. 1–7. [CrossRef]
14. Yeboah-Ofori, A.; Islam, S. Cyber Security Threat Modelling for Supply Chain Organizational Environments. *Future Internet* **2019**, *11*, 63. [CrossRef]
15. Sami, I.U.; Ahmad, M.B.; Asif, M.; Ullah, R. DoS/DDoS Detection for Healthcare in Internet of Things. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 297–300.
16. Wanasinghe, T.; Wroblewski, L.; Petersen, B.; Gosine, R.; James, L.; De Silva, O.; Mann, G.; Warrian, P. Digital twin for the oil and gas industry: Overview, research trends, opportunities and challenges. *IEEE Access* **2020**, *8*, 104175–104197. [CrossRef]
17. Khan, W.; Rehman, M.; Zangoti, H.; Afzal, M.K.; Armi, N.; Salah, K. Industrial Internet of Things: Recent advances, enabling technologies and open challenges. *Comput. Electr. Eng.* **2020**, *81*, 106522. [CrossRef]
18. Ghosh, S.; Sampalli, S. A survey of security in SCADA networks: Current issues and future challenges. *IEEE Access* **2019**, *7*, 135812–135831. [CrossRef]
19. Mumtaz, S.; Alsohaily, A.; Pang, Z.; Rayes, A.; Tsang, K.; Rodriguez, J. Massive internet of things for industrial applications: Addressing wireless IIoT connectivity challenges and ecosystem fragmentation. *IEEE Ind. Electron. Mag.* **2017**, *11*, 28–33. 10.1109/MIE.2016.2618724. [CrossRef]
20. Cheng, X.; Luo, Q.; Pan, Y.; Li, Z.; Zhang, J.; Chen, B. Predicting the APT for Cyber Situation Comprehension in 5G-Enabled IoT Scenarios Based on Differentially Private Federated Learning. *Secur. Commun. Netw.* **2021**, *8814068*, 14. [CrossRef]
21. Taylor, P.; Dargahi, T.; Dehghantanha, A. Analysis of apt actors targeting iot and big data systems: Shell crew, net-traveler, projectsauron, copykittens, volatile cedar and transparent tribe as a case study. In *Handbook of Big Data and IoT Security*; Springer: Cham, Switzerland, 2019; pp. 257–272. [CrossRef]
22. Fielder, A.; Li, T.; Hankin, C. Defense-in-depth vs. critical component defense for industrial control systems. In Proceedings of the 4th International Symposium for ICS & SCADA Cyber Security Research 2016 (ICS-CSR), Belfast, UK, 23–25 August 2016. [CrossRef]
23. Teyou, D.; Kamdem, G.; Ziazet, J. Convolutional neural network for intrusion detection system in cyber physical systems. *arXiv* **2019**, arXiv:1905.03168.
24. Latif, S.; Zou, Z.; Idrees, Z.; Ahmad, J. A novel attack detection scheme for the industrial internet of things using a light-weight random neural network. *IEEE Access* **2020**, *8*, 89337–89350. [CrossRef]
25. Wang, X.; Zheng, K.; Niu, X.; Wu, B.; Wu, C. Detection of command and control in advanced persistent threat based on independent access. In Proceedings of the 2016 IEEE International Conference on Communications (ICC), Kuala Lumpur, Malaysia, 22–27 May 2016; pp. 1–6. [CrossRef]
26. Brogi, G.; Tong, V. Terminaptor: Highlighting advanced persistent threats through information flow tracking. In Proceedings of the 2016 8th IFIP International Conference on New Technologies, Mobility and Security (NTMS), Larnaca, Cyprus, 21–23 November 2016; pp. 1–5. [CrossRef]
27. Sigholm, J.; Bang, M. Towards offensive cyber counterintelligence: Adopting a target-centric view on advanced persistent threats. In Proceedings of the 2013 European Intelligence and Security Informatics Conference, Uppsala, Sweden, 12–14 August 2013; pp. 166–171. [CrossRef]

28. Cheng, X.; Zhang, J.; Chen, B. Cyber situation comprehension for iot systems based on apt alerts and logs correlation. *Sensors* **2019**, *19*, 4045. [CrossRef] [PubMed]
29. Parra, T.; Rad, P.; Choo, K.; Beebe, N. Detecting internet of things attacks using distributed deep learning. *J. Netw. Comput. Appl.* **2020**, *163*, 102662. [CrossRef]
30. Zheng, D.; Hong, Z.; Wang, N.; Chen, P. An improved ldabased elm classification for intrusion detection algorithm in iot application. *Sensors* **2020**, *20*, 1706. [CrossRef] [PubMed]
31. Khalvati, L.; Keshtgary, M.; Rikhtegar, N. Intrusion detection based on a novel hybrid learning approach. *J. AI Data Min.* **2018**, *6*, 157–162. [CrossRef]
32. Ghafir, I.; Hammoudeh, M.; Prenosil, V.; Han, L.; Hegarty, R.; Rabie, K.; Aparicio-Navarro, F. Detection of advanced persistent threat using machine-learning correlation analysis. *Future Gener. Comput. Syst.* **2018**, *89*, 349–359. j.future.2018.06.055. [CrossRef]
33. Shudong, L.; Qianqing, Z.; Xiaobo, W.; Weihong, H.; Zhihong, T. Attribution Classification Method of APT Malware in IoT Using Machine Learning Techniques. *Secur. Commun. Netw.* **2021**, *2021*, 9396141. [CrossRef]
34. Janarthanan, T.; Zargari, S. Feature selection in unsw-nb15 and kddcup'99 datasets. In Proceedings of the 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE), Edinburgh, UK, 19–21 June 2017; pp. 1881–1886. [CrossRef]
35. Yang, Q.; Liu, Y.; Chen, T.; Tong, Y. Federated machine learning: Concept and applications. *ACM Trans. Intell. Syst. Technol. (TIST)* **2019**, *10*, 1–19. [CrossRef]
36. Gottwalt, F.; Chang, E.; Dillon, T. Analysis of feature selection techniques for correlation-based network anomaly detection. In Proceedings of the International Conference on Applications and Techniques in Cyber Security and Intelligence, Shanghai, China, 22–24 June 2018; pp. 10–16. [CrossRef]
37. Hoffbeck, J.; Landgrebe, D. Covariance matrix estimation and classification with limited training data. *IEEE Trans. Pattern Anal. Mach. Intell.* **1996**, *18*, 763–767. [CrossRef]
38. Van der Maaten, L.; Hinton, G. Visualizing data using tsne. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
39. Alkasassbeh, M.; Al-Naymat, G.; Hassanat, A.; Almseidin, M. Detecting distributed denial of service attacks using data mining techniques. *Int. J. Adv. Comput. Sci. Appl.* **2016**, *7*, 436–445. [CrossRef]
40. Dada, E.; Bassi, J.; Hurcha, Y.; Alkali, A. Performance evaluation of machine learning algorithms for detection and prevention of malware attacks. *IOSR J. Comput. Eng.* **2019**, *21*, 18–27. [CrossRef]
41. Safavian, S.; Landgrebe, D. A survey of decision tree classifier methodology. *IEEE Trans. Syst. Man Cybern.* **1991**, *21*, 660–674. [CrossRef]
42. Ali, J.; Khan, R.; Ahmad, N.; Maqsood, I. Random forests and decision trees. *Int. J. Comput. Sci. Issues (IJCSI)* **2012**, *9*, 272.
43. Agarap, A. Towards building an intelligent anti-malware system: A deep learning approach using support vector machine (svm) for malware classification. *arXiv* **2017**, arXiv:1801.00318.
44. Peng, C.; Lee, K.; Ingersoll, G. An introduction to logistic regression analysis and reporting. *J. Educ. Res.* **2002**, *96*, 3–14. [CrossRef]
45. Jahromi, A.; Taheri, M. A non-parametric mixture of gaussian naive bayes classifiers based on local independent features. In Proceedings of the 2017 Artificial Intelligence and Signal Processing Conference (AISP), Shiraz, Iran, 25–27 October 2017; pp. 209–212. [CrossRef]
46. Liang, G.; Zhu, X.; Zhang, C. An empirical study of bagging predictors for different learning algorithms. In Proceedings of the AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 7–11 August 2011; Volume 25. [CrossRef]
47. Lower, N.; Zhan, F. A study of ensemble methods for cyber security. In Proceedings of the 2020 10th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 6–8 January 2020; Volume 25. [CrossRef]
48. Peter, S.; Diego, F.; Hamprecht, F.; Nadler, B. Cost efficient gradient boosting. *Adv. Neural Inf. Process.* **2017**, *30*, 1551–1561.