

Article

A Secure and Intelligent Software-Defined Networking Framework for Future Smart Cities to Prevent DDoS Attack

Mohammed Mujib Alshahrani

College of Computing and Information Technology, University of Bisha, Bisha 61361, Saudi Arabia;
mmalshahrani@ub.edu.sa

Abstract: Smart cities have experienced significant growth in recent years, transforming people's lives into a more futuristic version. The smart city initiative includes a diverse collection of specifications, encompassing a large number of users whose requirements vary significantly and heterogeneously. Each device in smart cities generates a significant amount of data, which places a load on the gateways. Smart cities face a major challenge due to the enormous amount of data they generate. Through software-defined networking (SDN), network information paths are optimized, ensuring that traffic flow is evenly distributed across all network nodes. A considerable number of IoT devices with limited resources are susceptible to various security threats, such as device hijacking, ransomware, man-in-the-middle (MiM) attacks, and distributed denial-of-service (DDoS) attacks. These threats can pose a severe challenge to network security. Additionally, DDoS attacks have disrupted web businesses, resulting in the loss of valuable data. To counter DDoS attacks in a smart city, several options exist, yet many challenges remain. This research presents a secure and intelligent system to combat DDoS attacks on smart cities. SDN security controllers and machine learning models with optimization are employed in this study to reduce the impact of common DDoS attacks on smart cities. This work utilizes an SDN based on security controllers and a detection mechanism rooted in a machine learning model with optimization to mitigate various types of prevalent DDoS attacks within smart cities. Employing binary classification, XGBoost achieved an accuracy of 99.99%, precision of 97%, recall of 99%, an F1 score of 98%, and a false-positive rate of 0.05. In multiclass classification, the average accuracy is 99.29%, precision is 97.7%, recall is 96.69%, and the F1 score is 97.51%. These results highlight the superiority of this approach over other existing machine learning techniques.



Citation: Alshahrani, M.M. A Secure and Intelligent Software-Defined Networking Framework for Future Smart Cities to Prevent DDoS Attack. *Appl. Sci.* **2023**, *13*, 9822. <https://doi.org/10.3390/app13179822>

Academic Editor: Andrea Prati

Received: 4 August 2023

Revised: 29 August 2023

Accepted: 29 August 2023

Published: 30 August 2023



Copyright: © 2023 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: smart city; software-defined networking; DDoS attack; cybersecurity

1. Introduction

It is anticipated that by 2050, about 66% of the world's population will be living in cities [1]. The “smart city” concept focuses on ICT-based solutions to improve the everyday lives of people, government, economics, mobility, environment, and living conditions in urban areas [2]. In a smart city, many disparate systems and services are involved in complex connections with other systems in order to provide new data-oriented intelligent functions that leverage physical and cyberspaces. Governments, institutions, and private companies are all interested in learning about the potential benefits of smart city initiatives, as they solve many of the existing issues that affect densely populated areas. Future smart cities will have more technology to facilitate and improve the quality of life of their citizens. On the technological side, smart city initiatives require systems that can support large numbers of people using a diverse range of devices. The environment of a smart city is characterized by heterogeneity, with many different systems that need to be able to interoperate with one another and efficiently accomplish their functions. The interoperability aspect in a smart city is still elusive. This is because smart cities are distributed architectures, which necessitate a certain degree of interoperability and interoperation for managing heterogeneous systems that comprise diverse platforms. The

nature of smart cities necessitates this requirement. These heterogeneous systems are designed independently of one another, and each one features a unique operating system, programming platform, and tier of service.

These heterogeneous devices produce a significant amount of data on the network, thereby increasing the load on the gateways. The advancement of the internet of things (IoT) has spurred its adoption in domains like smart homes, smart cities, and related sectors, contributing to its exponential growth in recent years. Conversely, due to this development, IoT networks [3–6] are witnessing an upswing in security concerns, notably botnet attacks, often appearing as network anomalies. In a similar vein, providing security solutions proves challenging due to the limited resources accessible to devices connected to IoT networks in general. This challenge has been addressed through the utilization of the software-defined networking (SDN) computing paradigm, creating an environment that offers extra resources and adaptability for any anomaly mitigation systems.

This complexity and loading challenge can be surmounted by adopting SDN architecture for an efficient network management platform. The segregation of the control and data planes in SDN facilitates streamlined management, control, dynamic rule updates, analysis, and a broader network perspective from a centralized control point. Measurement stands out as one of the most pivotal and intricate aspects of centralized network management. The overarching objective is to identify DDoS attacks before smart city systems become inaccessible. DDoS attacks inundate a smart city device or a set of devices with a substantial number of packets, rendering them vulnerable. Smart city switches fail to identify a match if the incoming packet's source addresses are spoofed, which is commonly the case, necessitating the transmission of the packet to the controller. There's a possibility that the resources of the smart city controller could be depleted due to the constant processing of genuine and DDoS-spoofed packets. If the controller becomes incapable of receiving new legitimate packets, the SDN architecture might falter, rendering the controller unreachable. The use of SDN for network assessment in smart cities has been explored in various studies. Detecting DDoS attacks with speed and precision remains highly challenging. Some DDoS attackers employ packets that mimic normal traffic to disrupt the functioning of smart city systems. During a DDoS attack, the normal network conceals the traffic, leading traditional packet-based intrusion detection systems (IDS) to fail to identify it. Through the application of advanced machine learning algorithms and optimization approaches, a novel countermeasure will be devised through the monitoring and prompt identification of instantaneous network changes.

Cyber-attacks on smart cities are escalating, posing a threat to the advancements achieved in technological innovation. Smart cities are more susceptible to DDoS attacks, regardless of the nature of the targeted traffic. Smart cities that rely heavily on software-driven, complex digital networks to operate numerous city systems and services are vulnerable to cyber-attacks. When a DDoS attack is launched against a target in an SDN network, the controller triggers multiple flow entries due to the large volume of spoofed source addresses. This study addresses the following questions:

1. What is the best SDN framework and testbed for smart cities that can be used for evaluation?
2. How is the effectiveness of the anomaly mitigation schemes determined?
3. How is the best multiclass classification determined?

The failure of the controller can lead to the breakdown of entire or partial smart city systems. As a result, this study proposes a secure and intelligent framework based on an SDN scheme to protect smart cities from DDoS attacks.

The main goals of this work are to introduce an intelligent learning-based DDoS detection framework that mitigates the DDoS attacks on smart cities. The detailed objectives of this work can be summarized as follows:

- To propose an intelligent learning-based DDoS framework for eliminating DDoS attacks on smart cities.
- To improve the efficiency of the smart cities' SDN-enabled centralized network by further enhancing the model.

- To simulate the proposed framework in smart city systems for evaluation and benchmarking.

The remainder of this paper is organized as follows. Section 2 summarizes and discusses related work on DDoS attacks in smart cities. Section 3 describes the framework, algorithm, and features of the proposed work. Section 4 describes the evaluation and performance metrics of the proposed framework. In Section 5, we make final remarks and directions for future work.

2. Background and Related Work

DDoS attacks, which have been extensively studied, are among the most prevalent and serious security concerns. A variety of projects are now using SDN-based security measures to prevent such assaults. Here is a summary of some important works in this area:

Jesús et al. [7] proposed an SDN-based solution to combat DoS and DDoS attacks in IoT networks. The solution relies on OpenState, a potential technique for network monitoring, as it does not send packets to the controller. The SDN controller can identify DoS and DDoS attacks based on their entropy levels, as has been shown. The impact of this parameter was better understood by evaluating the application's performance in three different scenarios. The first scenario measures the attack's bandwidth and entropy values in a generic testbed, while the second and third scenarios are more focused on IoT scenarios. According to the experimental findings, an attack can be detected by comparing the entropy levels of different aspects. Chuanfeng et al. [8] presented a method for improving DDoS protection and data management security in SDN-enabled smart cities. A DDoS attack defense approach based on traffic classification was presented in this paper. The authors used a software-defined network function, virtualization design, and traffic categorization approach to improve SDN flexibility and load against DDoS attacks. According to the experimental results, the suggested approach can not only identify DDoS attacks quickly, but it can also correctly pinpoint the origins of DDoS attacks. As a result, SDN controllers are less vulnerable to attack, and the system is more efficient.

Da et al. [9] used the SDx paradigm to propose an overall framework for software-defined internet of things (SD-IoT). The controller pool in the proposed system consists of SD-IoT controllers, switches connected to an IoT gateway, and IoT devices. Researchers developed a method for identifying and mitigating DDoS attacks using the SD-IoT architecture. The similarity measured by the cosine of the packet message rates at the boundary SD-IoT switch ports is employed in the algorithm to assess whether DDoS attacks are present in the IoT. Experiments conducted in this study demonstrate that the suggested algorithm performs well and that the proposed framework can be easily adapted to increase IoT security for diverse and susceptible devices. Narmeen et al. [10] proposed an efficient DDoS attack detection system based on DDoS detection techniques. According to the findings provided in this work, DDoS attacks may be detected and mitigated in a large-scale network that includes a smart city based on SDN architecture. This framework can meet all DDoS attack detection and mitigation requirements. This study first surveys and classifies SDN-based DDoS attack identification and resolution systems based on the detection approach. Second, the authors presented an SDN-based DDoS Framework, which utilizes the SDN's properties for network security. Applications for smart cities may be secured with the framework presented here.

Application identification in network architecture has been advocated by Suh et al. [11]. This can help to prevent distributed denial-of-service (DDoS) attacks by restricting data flow. YuHunag et al. [12] proposed a DDoS attack identification technique that relies on network traffic statistics. The controller monitors the amount of traffic and the frequency of specific events that are associated with DoS attacks. Braga et al. [13] introduced a technique based on self-organizing maps. Researchers have developed a way to detect fraudulent internet patterns using this method. Zhang et al. [14] discussed the current state of prior network behavior and analysis. They developed a flow count identification technique for anomaly detection. This approach can be used to detect anomalous traffic flows that are associated with DDoS attacks. In another work, the authors compared the known

testbeds to the one that is used to create the IoT-Bot dataset, which is a simulated network environment that simulates real-world traffic [15].

A variety of datasets have been presented in the literature to assist researchers in modeling botnet operations and generating attack traffic statistics [16–19]. These datasets were generated using a variety of testbeds. Alomari et al. [17] built the DDoS Botnet traffic testbed using a high-tier server and virtual machine. The number of bots in their testbed was larger, but their produced botnet activities were restricted to HTTP DDoS attacks exclusively. They did not use machine learning algorithms. The data generated by this network testbed was intended to provide evidence of malicious activity on the internet. However, it is difficult to verify any results due to the lack of comprehensive network packet capture. The amount of data that can be retrieved and analyzed is also limited by network traces.

Bhatia et al. [19] built their testbed using real devices connected to a local network in a different way. They used Botloader and IP-Aliasing, two pieces of specialist software, to replicate flash events and other DDoS attacks on their testbed. The decision to use physical computers instead of a virtualized testbed has some drawbacks. It is more expensive, more difficult to install, and does not offer the same level of robustness as a virtualized environment. However, their methodology also encompasses a wider range of botnet operations, including port scanning using machine learning methods. In another study, Sharafaldin et al. [20] developed two independent testbeds based on physical computers. One testbed was for the network of the victim, and the other was for the network of the attacks. The authors chose to use the Ubuntu/Kali Linux platform for malware detection, and they used a wide range of popular operating systems in their approach. Moustafa et al. [15] built their UNSW-NB15 dataset using IXIA's Perfect Storm testbed to create both normal and malicious network traces. The researchers used IXIA to set up three virtual servers, two of which generated regular traffic and the third of which carried out attacks on the other two. Most of the existing work in this area has been addressing DoS/DDoS and IoT. However, there are fewer works that address the applicability of SDN to smart cities.

Jagtap et al. [21] proposed a novel intrusion detection and prevention system to prevent DDoS attacks. The authors introduced a long short-term memory (LSTM) and graded rated unit (GRU) deep learning model as the "block-attack" model, where the LSTM and GRU contribute to enhancing the rate of accuracy in detecting DDoS attacks in an SDN environment. They used the CICDDoS2019 dataset for the experiments. They achieved 98.5% accuracy in detecting and preventing the DDoS attacks and 95.5% accuracy for the SVM-based method.

Recently, Negera et al. [22] introduced a lightweight model for botnet attack detection in software-defined network-orchestrated IoT. The aim of this model is to enhance the security framework of IoT by harnessing the capabilities of IoT devices to efficiently thwart botnet malware attacks. By dynamically allocating computational resources, the model achieves rapid response times. Notably, empirical evaluation showcases the model's exceptional performance, yielding remarkable metrics such as 99% precision, recall, and F1 score, in conjunction with an impressive accuracy rate of 99.4%. Furthermore, the model's size, a mere 118 KB, coupled with its minimal parameter count of 19,414, contributes to its agile execution time of a mere 0.108 milliseconds. A comparative analysis of the existing work is presented in Table 1.

Table 1. Comparative analysis of existing works.

Authors	Detection Methodology	SDN	Machine Learning	Test Strategy	Attacks Type	Dataset
Jesús Galeano-Brajones [7]	Entropy-based solution	SDN	No	Experiments	DoS/DDoS	Bot-IoT
Chuanfeng Xu [8]	Entropy-based solution	SDN	Yes	Simulation	DDoS	UNB ISCX
Da Yin [9]	Anomaly-based	SDx	No	Experiments	DDoS	N/A
Narmeen Zakaria [10]	Entropy-based solution	SDN	No	Simulation	DDoS	N/A
J. Suh [11].	CONA	No	No	NetFPGA-OpenFlow	DDoS	N/A
C. YuHunag [12]	Anomaly-based	No	No	OpenFlow	DDoS	N/A
R. Braga [13]	Anomaly-based	No	No	OpenFlow	DDoS	N/A
Y. Zhang [14]	Anomaly-based	Yes	No	OpenWatch	No	N/A
Moustafa, N [15]	Anomaly-based	NO	No	IXIA PerfectStorm	DoS	UNSW-NB15
Alomari et [17]	Anomaly-based	No	No	Simulation	DDoS	HTTP Botnet
Bhatia [19]	Entropy-based solution	No	No	Simulation	DDoS	CAIDA
Sharafaldin [20]	Anomaly-based	No	Yes	CICFlowMeter	DDoS	Many

3. Proposed SDN Simulation Framework

The proposed SDN simulation framework for smart cities is shown in Figure 1. The framework has three layers: the infrastructure layer, the secure and intelligent SDN layer, and the service layer. The suggested architecture enables diverse networks that include IoT devices, RFID, WSN, ZigBee, sensors, and other network devices. The infrastructure layer includes both the IoT devices and the forwarding devices. The IoT devices, such as RFID, ZigBee, sensors, and WSN, create a variety of IoT applications suitable for smart cities. These wireless devices collect massive volumes of network data, which are then sent to an SDN-based smart city controller for processing. The forwarding devices sublayer consists of MQ telemetry transport (MQTT) gateways, which make it easier for the SDN controller to receive control and data packets. The secure and intelligent SDN layer consists of the global and local SDN controllers. The global SDN controller is responsible for controlling and monitoring communications between the global control center and the IoT application domains. The local SDN controller manages and monitors the communications within an application domain. The service layer makes IoT services possible through the use of SDN controllers. It also provides network services such as routing, security, and quality of service throughout the city. The control plane, which is the low-level details of the configuration and operation of typical network devices such as switches and routers, is traditionally dependent on the operating system of the device in question. This can make it difficult and time-consuming to dynamically reconfigure a network.

SDN is intended to address this issue, according to the article [23], by the idea of separation of control and data planes, which will allow for software-based device design, as seen in Figure 1. As a result of these principles, a software-based element may be used to operate and configure the network, providing all accompanying benefits, such as dynamic control. Using a logically centralized controller, network performance may be monitored and dynamically adjusted.

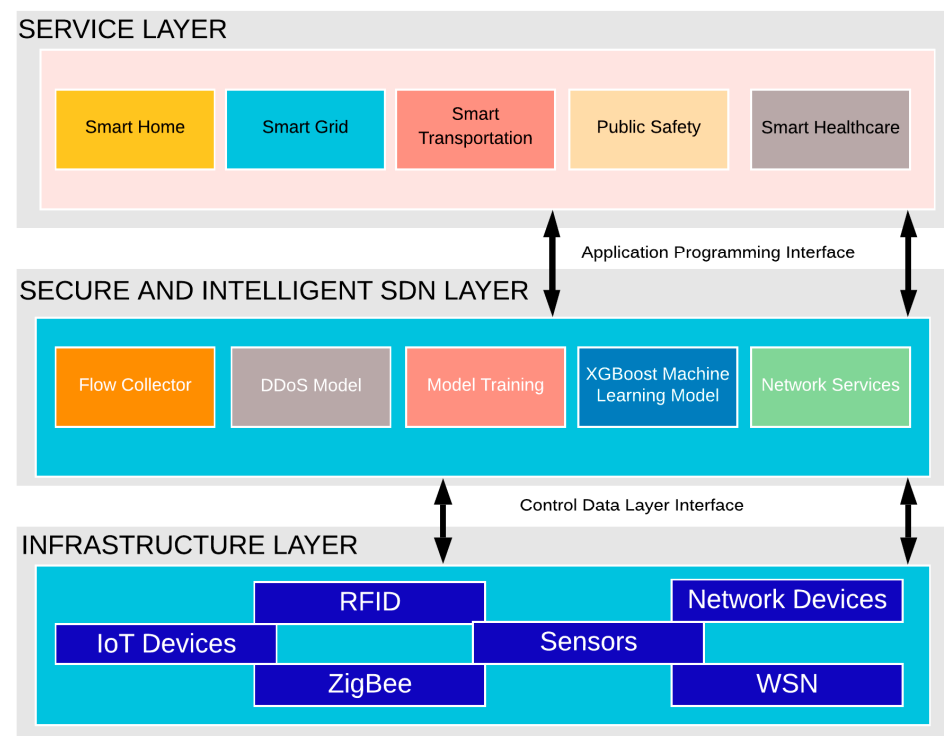


Figure 1. Proposed secure and intelligent SDN framework for smart cities.

3.1. Feature Space and Classification Model

The proposed test framework has three major components: the network platform, the simulated IoT services, and the feature extraction and attack analysis. The network platform consists of both normal and malicious virtual machines, including a firewall. The simulated IoT services include certain IoT services that sense the data in the network. The Node-RED tool [24] is used to represent the flow model of MQTT. Node-RED is a flow-based visual programming tool built on NodeJS, which is commonly used in IoT system development. Developers have the freedom to use Node-RED in a variety of ways, and the same system can be constructed in a variety of methods. The BoT-IoT dataset [25] has been used to obtain the data features. The XGBoost methods were then used to analyze the feature vectors to differentiate between normal and abnormal cases. The packets are collected using the pcap utility to produce the required network flow. The extracted features from the network flow, as shown in Table 2, are stored in the database.

Table 2. Feature set used in the framework [26].

Sr. No.	Name	Description
1.	seq	Sequence number assigned by the Argus extractors
2.	stddev	Standard deviation
3.	min	The minimum timeframe for all instances
4.	mean	The average timeframe for all instances
5.	max	Maximum timeframe for all instances
6.	N_IN_Conn_P_DstIP	No. of incoming connections/destination IP
7.	N_IN_Conn_P_SrcIP	No. of incoming connections/Source IP
8.	drate	Packets/second destination to source
9.	State_number	Numeric value for feature state

The BoT-IoT dataset is a composition of normal traffic, probing attacks, DDoS and DoS attacks, and information theft, as shown in Figure 2. The normal traffic consists of legal network transactions. Virtual machine traffic flows are included in the data collection. The

probing attack, also known as an information-gathering attack, is carried out by malicious individuals who use scanning or fingerprinting techniques to illegitimately obtain data from remote computers. Port scanning and OS fingerprinting are two forms of probing attacks found in the BoT-IoT dataset. In a DDoS attack, a malicious user overwhelms resources or services with invalid requests. Botnets, which are collections of hacked nodes on the network, are often used to carry out these attacks. The dataset includes HTTP, TCP, and UDP DDoS attacks.

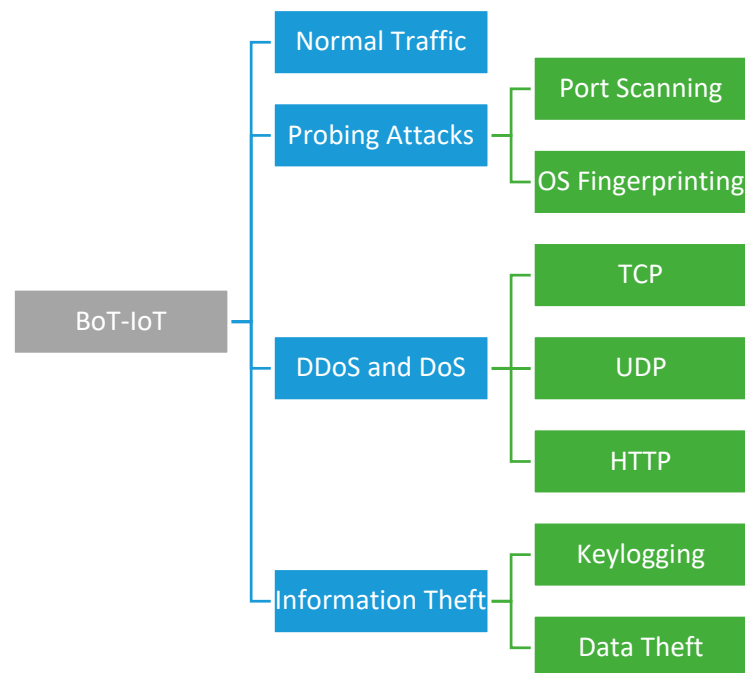


Figure 2. Structure of BoT-IoT dataset.

While attackers can use an information theft attack to gain access to sensitive or secret information, malevolent users may also use it. Data theft and keylogging are two types of information theft attacks in the dataset. The BoT-IoT data collection contains 9543 legitimate instances and 73,360,900 non-legitimate traffic flow instances. In this experiment, only 740,637 randomly selected examples were used. However, the retrieved instances include all forms of attacks, except for theft attacks, which have a negligible number in the BoT-IoT dataset.

To validate the proposed model, a typical smart parking configuration was selected using ten simulated IoT devices. The Node-RED root is used to establish the connection between smart devices and Amazon Web Services (AWS) that generate normal traffic using the MQTT protocol. MQTT was created to connect devices in remote locations where there is not much network bandwidth or where a “small code footprint” is required. It is a good choice for wireless networks with varying levels of latency due to sometimes limited bandwidth or unreliable connections.

MQTT started as an IBM-owned protocol that was used to communicate with SCADA systems in the Oil and Gas industry. It is now an open-source protocol that is run by the Organization for the Advancement of Structured Information Standards (OASIS). The MQ in MQTT stands for “message queuing,” but in MQTT communication, there is no longer any message queuing. The protocol now has publish-and-subscribe messaging, and smart automation systems are using it more and more. Today, MQTT is one of the most popular open-source protocols used in fog and edge computing and to connect the internet of things (IoT). Aside from MQTT, there are other well-known messaging protocols that IoT applications can use.

The gradient-boosted tree (GBT) approach is categorized as supervised learning based on the approximation method through the optimization of certain loss functions as well as the use of multiple regularization strategies [27]. In our analysis, we are looking for a function that can improve the performance of the proposed model. Therefore, loss function L becomes a good indicator of how accurate our model's predictions are. If the prediction results $\hat{y}_i$ are close to the actual values y_i , then the loss will be the smallest, and if the predictions are completely off from the original values, then the loss will be the greatest. The loss can be defined by using Equation (1).

$$L = |y_i - \hat{y}_i| \quad (1)$$

Based on the value of L , the model is iterated for update until the best result is achieved. To obtain the classification, the binary cross-entropy (Log loss) has to be employed. In XGBoost, we have many numbers of trees [23,24]. Let us assume that we have G trees, then the prediction model can be defined as $\sum_{g=1}^G f_g$. The f_g represents the prediction of the decision. We create predictions using all decision trees that we have, as shown in Equation (2):

$$\hat{y}_i = \sum_{g=1}^G f_g(x_i) \quad (2)$$

The x_i represents the feature vector for the data point i . We can also define the prediction using this model at any step t , as shown in Equation (3).

$$\hat{y}_i^t = \sum_{g=1}^t f_g(x_i) \quad (3)$$

We must optimize the loss function in order to train the model. For binary classification, we use LogLoss, as shown in Equation (4).

$$L = -\frac{1}{N} (\sum_{i=1}^N y_i \log(p_i) + (1 - y_i) \log(1 - p_i)) \quad (4)$$

The multiclassification can be represented as shown in Equation (5):

$$L = -\frac{1}{N} (\sum_{i=1}^N \sum_{j=1}^M y_{i,j} \log(p_{i,j})) \quad (5)$$

where N represents the total rows, and M is the collection of classes. The objective function consists of a loss function and regularization at the iteration t that has to be minimized. We can define the objective function of XGBoost by Equation (6) [27].

$$\mathcal{L}^t = \sum_{i=1}^n l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) ; \text{ s.t } x = \hat{y}_i^{(t-1)} \quad (6)$$

The simple approximation function can be computed by Equation (7). The $f(x)$ represents the loss function l , while a is the predicted value obtained from the previous step ($t-1$). The Δx is defined as a new learner in this context for iteration t . Now, applying the second-order approximation, Equations (8) and (9) can be obtained as follows.

$$f(x) \approx f(a) + f'(a)(x - a), \Delta x = f_t(x_i), \quad (7)$$

$$f(x) \approx f(a) + f'(a)(x - a) + 1/2 f''(a)(x - a)^2 \quad (8)$$

$$\mathcal{L}^t = \sum_{i=1}^n [l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + 1/2 h_i f_t^2(x_i)] + \Omega(f_t) \quad (9)$$

where $g_i = \partial_{\hat{y}_i^{(t-1)}} l(y_i, \hat{y}_i^{(t-1)})$, and $h_i = \partial_{\hat{y}_i^{(t-1)}}^2 l(y_i, \hat{y}_i^{(t-1)})$.

To generalize the above equation, the constant parts can be removed, as shown in Equation (10).

$$\tilde{\mathcal{L}}^t = \sum_{i=1}^n [g_i f_t(x_i) + 1/2 h_i f_t^2(x_i)] + \Omega(f_t) \quad (10)$$

Equation (10) is the sum of quadratic functions that has one variable, which can be minimized further, as shown in Equations (11) and (12).

$$\operatorname{argmin}_x G_x + \frac{1}{2} H x^2 = -\frac{G}{H}, H > 0 \quad (11)$$

$$\min_x G_x + \frac{1}{2} H x^2 = -\frac{G^2}{H} \quad (12)$$

3.2. Simulated Attack Model

To simulate the attack model, we deployed six virtual machines with the Kali Linux operating system to generate a botnet-simulated attack model, as shown in Figure 3. In this work, we modeled denial-of-service (DoS) attacks [17,24,26–28] that can disrupt normal service and make it unavailable to users. The virtual machines act as bots to target remote servers or machines. The attack can be identified by the large volume of data generated, which can prevent legitimate users from accessing the service. Additionally, the attack can crash the system by increasing the request load, making the provided service unavailable. This type of attack can also be carried out through a protocol that abuses the working of the internet protocol (IP) to deplete the computing resources of the target machines so that they cannot respond to requests from legitimate users. To set up the test scenario, TCP, UDP, and HTTP were used.

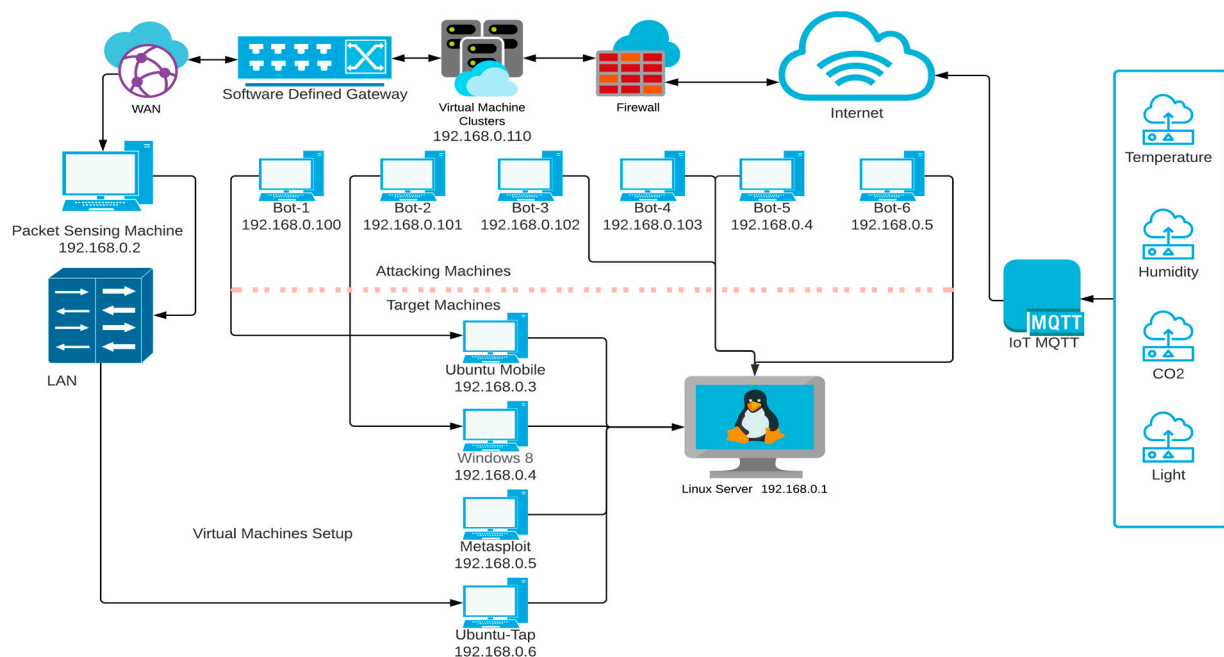


Figure 3. Modified test framework for SDN-based smart city [29].

4. Results and Discussion

Anomaly mitigation schemes must be tested to ensure their effectiveness. Evaluating a plan helps to determine its effectiveness. Therefore, it is necessary to use a structured dataset that effectively represents the environmental trace flow characteristics in which the model will be implemented, as this cannot be achieved on a real-world network. The BoT-IoT dataset provides an appropriate setting for traffic flow in a connected IoT environment, so we will use it for this purpose [26]. The dataset contains both real and simulated

IoT traffic, as well as threats. The dataset was created on an IoT testbed using feature extraction methods and network platforms. Virtual machines were used to generate valid and malicious traffic on the network platforms. The MQTT protocol is used to simulate smart city network traffic [29]. A temperature sensor station, humidity sensor, CO₂, and smart lights are some of the scenarios simulated. Fourteen new features were derived from the thirty features to improve the predictive power of the classifiers that will be used. The BoT-IoT consists of normal traffic, probing attacks, DDoS, and information theft. The BoT-IoT contains 9543 instances of legitimate traffic flow, while it has 73,360,900 illegitimate instances. However, only 740,637 cases were randomly selected for this simulation. The theft attacks, which are negligible, are not included in the extracted occurrences. As illustrated in Figure 4, the model's performance is evaluated using the confusion matrix to quantify the accuracy of its predictions.

		Predicted class		
		Malicious	Legitimate	
Actual class	+	TP	FN	Sensitivity
	−	FP	TN	Specificity
		Precision	NPV	Accuracy

Figure 4. Performance measurement through confusion matrix.

The performance of the model can be accessed through metrics such as accuracy, detection rate, false-positive rate, false-negative rate, F1 score, etc. We will present a brief definition of these metrics.

- a. Accuracy: This is the measure of all positive classifications of all instances in the dataset of an intrusion detection system that can be derived as follows:

$$(Acc = TN + TP) / (TP + FP + TN + FN)$$

where TN represents the total negative classified instances, TP represents the positive classified instances, FP represents false-positive classified, and FN represents false-negative classification of the data.

- b. Detection Rate (DR): The true-positive rate (TPR) is the number of properly recognized malicious observations to the total number of malicious observations in the dataset, which can be derived as follows:

$$DR = TP / (FN + TP)$$

- c. False-Positive Rate (FPR): This is the percentage of normal observations to the total number of normal data that are incorrectly labeled as attacks:

$$FPR = FP / (FP + TN)$$

- d. False-Negative Rate (FNR): An attack's misclassification rate, which is also known as precision, is calculated as follows:

$$FNR = FN / (FN + TP)$$

- e. F1: Recall and accuracy may be weighted averaged to calculate F1 using the formula:

$$F1\ Score = 2 \times (Recall \times Precision) / (Recall + Precision)$$

To see how well it stacks up with existing classification algorithms, k-fold cross-validation tests validate that the XGBoost classifier performs supervisor decision tree, k-nearest neighbor, naïve Bayes, and gradient boosting.

In the 10-fold cross-validation procedure, a random number generator divides the dataset into ten equal pieces. Only a component of the split dataset is used for each assessment, and the remaining portions of the dataset are used for training purposes. Once for each segment of the partitioned dataset, the procedure is repeated to obtain superior results. The ratio of features in the dataset and the ratio of the training instances have been balanced to avoid overfitting. It ensures low variance and bias, as well as the avoidance of overfitting. In light of the dataset's asymmetry in terms of instances of traffic flows, the classifiers are evaluated in terms of the entire classification report, rather than on individual instances. The classifier's classification capability will be captured this way, allowing us to categorize distinct instances of binary classification (normal and attack instances), multiclass classification, and port scan. The classification report contains information on the accuracy, precision, recall, and F1 score.

Regarding detection accuracy, all classifiers achieved a satisfactory level of success, with XGBoost achieving the highest level of success and NB achieving the lowest level of success. Conversely, the XGBoost classifier achieved the lowest false-positive rate (FPR) with a value of 0.06, as shown in Table 3. This demonstrates that it is able to categorize network traffic cases with the lowest possible number of false positives. A detailed performance analysis of the multiclass classification performance is shown in Appendix A.

Table 3. Average performance metrics of binary classification.

No	Algorithms	Accuracy (%)	Precision	Recall	F1 Score	FPR
1	NB	96	55	50	53	0.90
2	k-NN	97.1	61	85	75	0.64
3	DT	97.9	93	93	93	0.16
4	GRB	98.3	57	88	63	0.87
5	XGBoost	99.9	97	99	98	0.06

Figure 5 depicts the performance of the classifiers in binary classes, such as normal and attack examples, which provides a better insight into the classifier's performance. The XGBoost classifier outperformed the other classifiers in terms of average recall, F1 score, and precision. It is possible to have a better understanding of how well the classifiers work by looking at the average performance of the binary classifiers in two classes. With respect to average recall accuracy and F1 score, the XGBoost classifier is found to be superior to other classifiers.

Figure 6 shows the classification accuracy for multiclass classification. A high percentage of attacks are detected by all classifiers in multiclass classification, where the performance is similar to the binary class classification. The k-NN and NB classifiers have the lowest detection accuracy for TCP, UDP, OS fingerprinting, and keylogging. Both the XGBoost and DT produced similar findings; however, the XGBoost had the best detection accuracy for both attack and nonattack.

The precision of the classifiers in multiclass classification is presented in Figure 7. In all attacks and typical cases, the XGBoost, GRB, and DT classifiers attained a precision of between 84 and 99.99. The naïve Bayes and k-NN have obtained a precision of 5–72 percent in all types of attack and normal instances, excluding the exception of DDoS TCP attacks, where naïve Bayes has reached a precision of 82.1 percent, and keylogging attacks, where k-NN achieved a precision of 98.7.

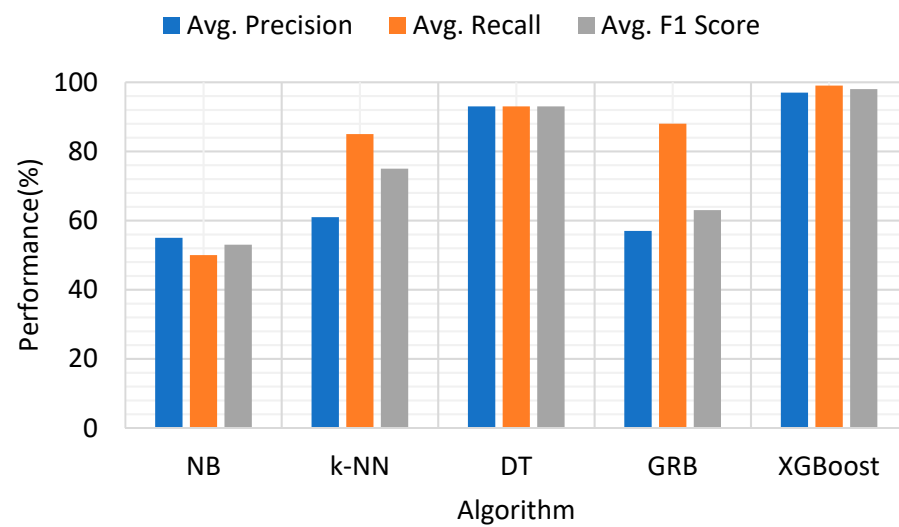


Figure 5. Average recall, precision, and F1 measure of the classifiers.

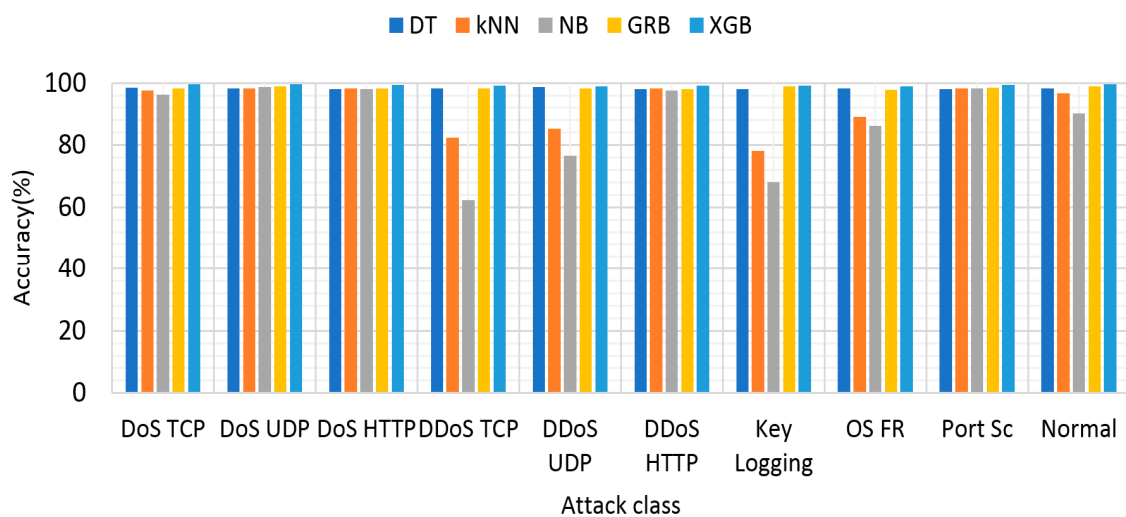


Figure 6. Accuracy of multiclass classification.

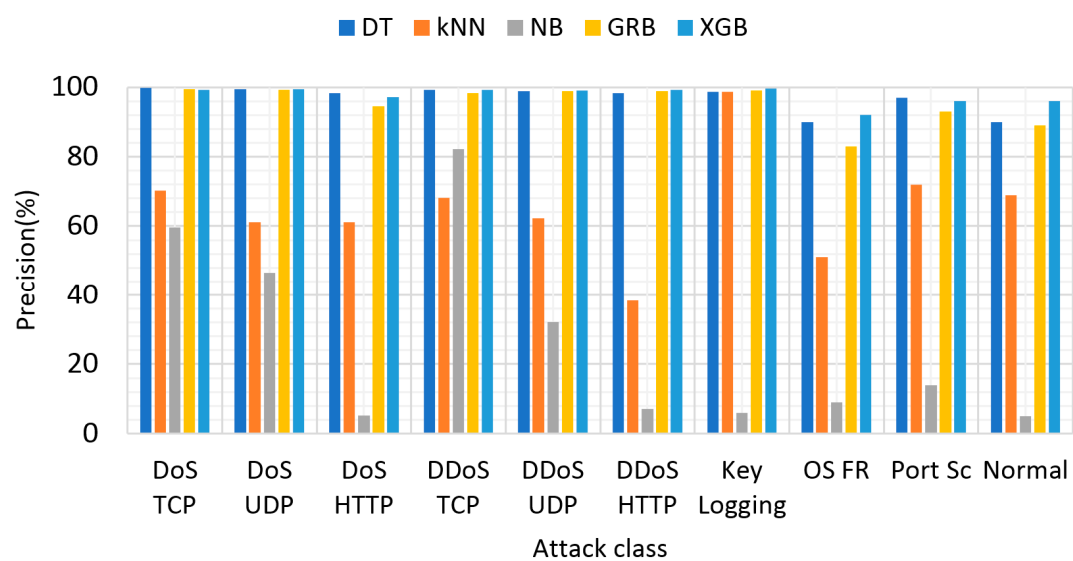


Figure 7. Precision of multiclass classification.

A multiclass classification system, except for keylogging attacks and normal cases, achieved recalls of 83–100% in all attacks and normal cases, respectively, in Figure 8, depicting the results of the multiclass classification system. There was a negligible difference in recall value between the two methods, except in the case of keylogging attacks (NB reported an 86% rate of recall).

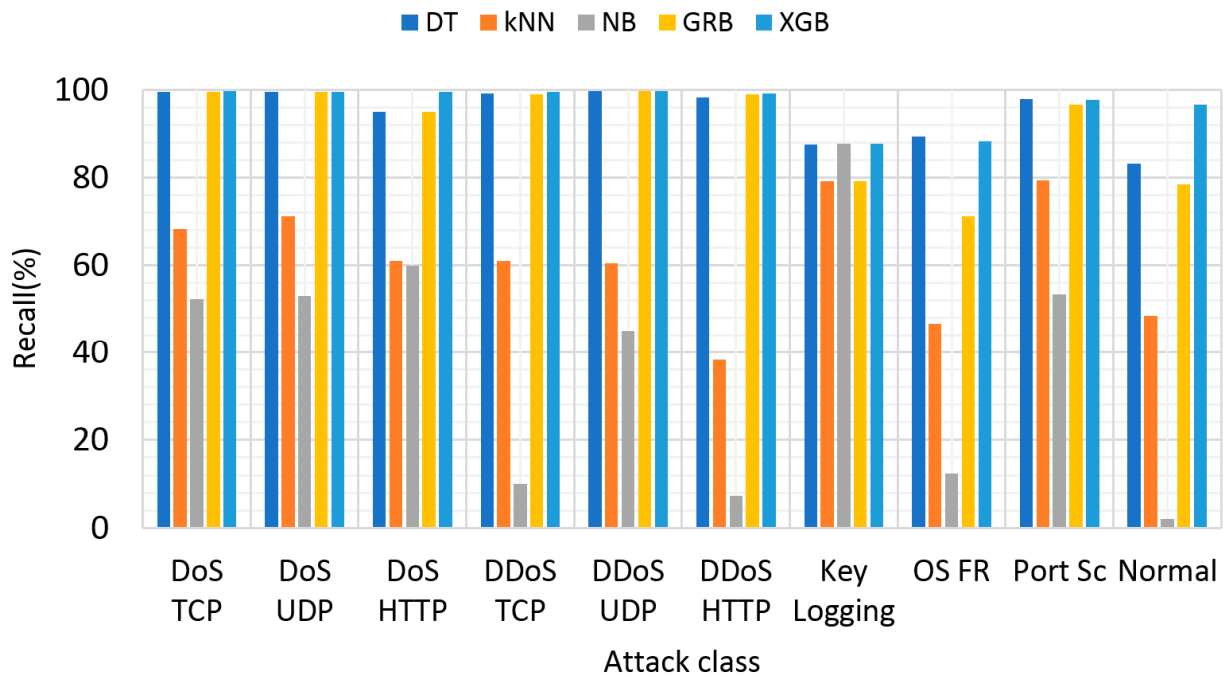


Figure 8. Recall of multiclass classification.

Regarding attacks and regular situations, classifiers such as the XGBoost, GRB, and DT classifiers all earned F1 scores of 82–100%, whereas NB and k-NN only received F1 scores of 4–70% in all but OS fingerprinting assaults, when k-NN scored an impressive 88%, as shown in Figure 9.

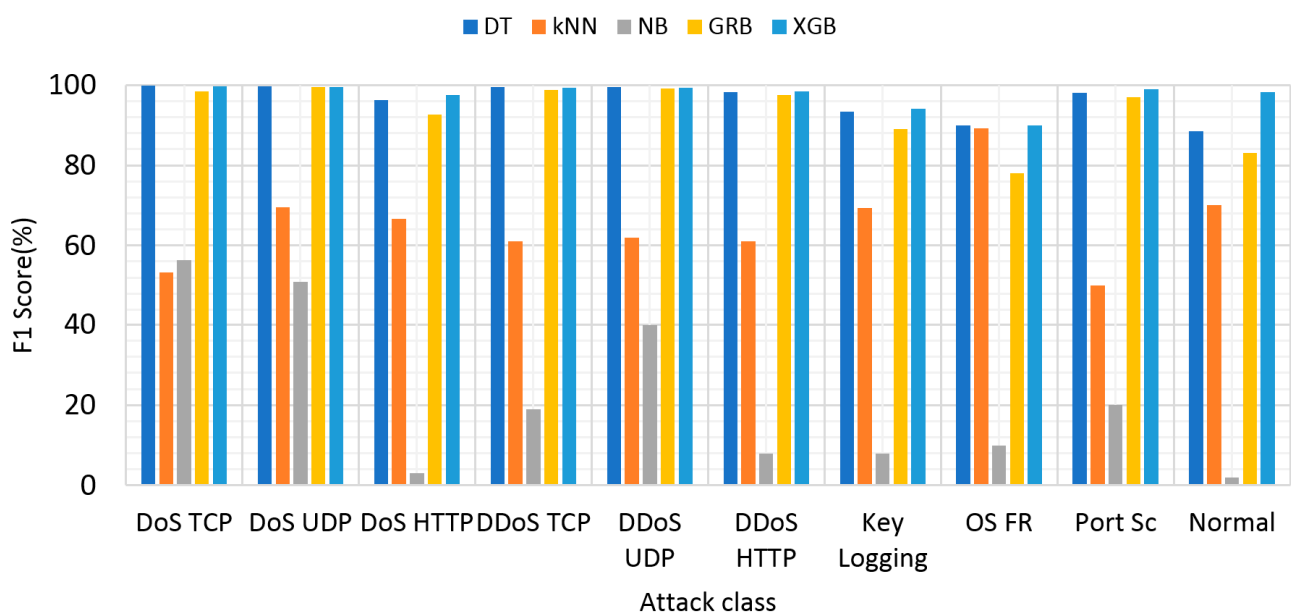


Figure 9. F1 score of multiclass classification.

The proposed model is compared with the existing work, as shown in Figure 10. In the reinforced learning approach, the random forest method [8] is applied, which has obtained an accuracy of 99.54%. In another work, the authors have applied ANN to obtain a high accuracy of 99.84% in the SDN environment. In our proposed work, we obtained slightly better than the existing work [8], which is 99.9%, as shown in Figure 10.

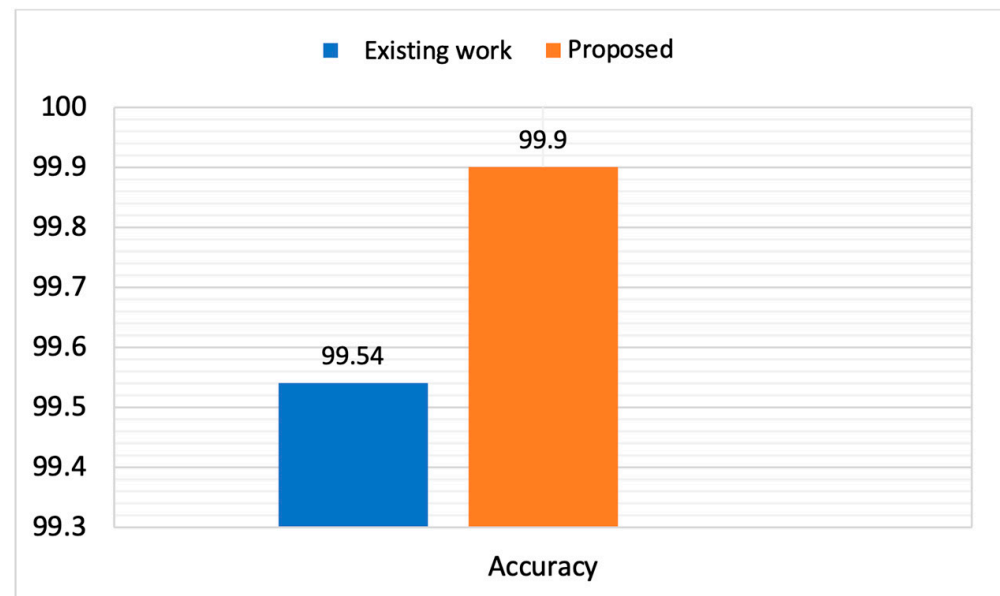


Figure 10. Comparative analysis of accuracy with existing work [8].

5. Conclusions

The heterogeneity and interoperability requirements must be met when designing a smart city application. These stringent requirements must be met by all smart city components, including devices, network equipment, vendor-proprietary software, communication technologies and protocols, and a variety of other smart services and smart city applications. SDN has emerged as a possible resilient future internet architecture in recent years. Numerous recent studies have shed light on how SDN can be used to improve the resilience and security of communication networks in smart cities. This work conducted a comprehensive and in-depth study to explain the essentials of SDN from the resilience perspective, followed by a proposal of a secure and intelligent framework. XGBoost achieved an accuracy of 99.99%, a precision of 97%, a recall of 99%, an F1 score of 98%, and an FPR of 0.05 using binary classification. In multiclass classification, the average accuracy was 99.29%, precision 97.7%, recall 96.69%, and F1 score 97.51%.

In our future work, we will explore and try the DDoS attack on high-performance targets.

Funding: This project was funded by the Deanship of Scientific Research, University of Bisha, Bisha, Kingdom of Saudi Arabia.

Data Availability Statement: The experiment has been performed on the open dataset, which is freely available at IEEE Dataport: Nour Moustafa, 16 October 2019, “The Bot-IoT dataset”, IEEE Dataport, <https://dx.doi.org/10.21227/r7v2-x988>, accessed on 25 April 2023.

Acknowledgments: The authors extend their appreciation to the Deanship of Scientific Research, University of Bisha, Saudi Arabia for funding this research work through the Promising Initiative Project under Grant Number (UB-Promising-08-1442).

Conflicts of Interest: The authors declare no conflict of interest between the authors or institutions.

Appendix A

Table A1. Multiclass Classification Performance Metrics.

Class	Algorithms																			
	DT				kNN				NB				GRB				XGB			
	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1	Acc	Pre	Rec	F1
DoS TCP	98.5	99.8	99.5	99.8	97.5	70.2	68.2	53.3	96.3	59.6	52.3	56.3	98.3	99.5	99.5	98.5	99.6	99.3	99.6	99.7
DoS UDP	98.2	99.5	99.5	99.7	98.3	61.1	71.2	69.6	98.6	46.5	53	51	98.9	99.2	99.4	99.5	99.5	99.5	99.5	99.5
DoS HTTP	98.1	98.2	95	96.2	98.2	61.2	61	66.6	98	5.2	60	3	98.3	94.5	95	92.6	99.4	97.2	99.5	97.5
DDoS TCP	98.3	99.2	99.2	99.6	82.3	68.2	61	61	62.3	82.1	10	19	98.3	98.3	99	98.8	99.2	99.2	99.5	99.3
DDoS UDP	98.6	98.9	99.6	99.6	85.2	62.2	60.5	62	76.5	32.2	45	40	98.2	98.9	99.6	99.2	99	99	99.6	99.3
DDoS HTTP	98.1	98.2	98.2	98.2	98.3	38.3	38.3	61	97.5	7.2	7.2	8	98	98.9	98.9	97.6	99.1	99.2	99.2	98.5
Key Logging	98	98.6	87.5	93.3	78.2	98.7	79.2	69.3	68.2	6	87.6	8	99	99	79.2	89	99.2	99.6	87.6	94.1
OS FR	98.3	90	89.3	90	89.2	51	46.6	89.2	86.2	9	12.3	10	97.8	83	71.2	78	99	92	88.2	90
Port Sc	98	97	97.8	98	98.2	72	79.3	50	98.3	14	53.3	20	98.4	93	96.5	97	99.3	96	97.7	98.9
Normal	98.3	90	83.2	88.5	96.6	69	48.5	70	90.2	5	2	2	98.9	89	78.5	83	99.6	96	96.5	98.3

References

1. Quasim, M.T.; Khan, M.A.; Algarni, F.; Alshahrani, M.M. Fundamentals of Smart Cities. In *Smart Cities: A Data Analytics Perspective*; Khan, M.A., Algarni, F., Quasim, M.T., Eds.; Lecture Notes in Intelligent Transportation and Infrastructure; Springer: Cham, Switzerland, 2021. [\[CrossRef\]](#)
2. Alghamdi, N.S.; Khan, M.A. Energy-efficient and blockchain-enabled model for internet of things (IoT) in smart cities. *Comput. Mater. Contin.* **2021**, *66*, 2509–2524.
3. Liu, H.; Li, S.; Wang, H.; Sun, Y. Adaptive fuzzy control for a class of unknown fractional-order neural networks subject to input nonlinearities and dead-zones. *Inf. Sci.* **2018**, *454–455*, 30–45.
4. Liu, H.; Li, S.; Cao, J.; Li, G.; Alsaedi, A.; Alsaadi, F.E. Adaptive fuzzy prescribed performance controller design for a class of uncertain fractional-order nonlinear systems with external disturbances. *Neurocomputing* **2017**, *219*, 422–430.
5. Liu, H.; Pan, Y.; Li, S.; Chen, Y. Synchronization for fractional-order neural networks with full/under-actuation using fractional-order sliding mode control. *Int. J. Mach. Learn. Cybern.* **2018**, *9*, 1219–1232.
6. Han, Z.; Li, S.; Liu, H. Composite learning sliding mode synchronization of chaotic fractional-order neural networks. *J. Adv. Res.* **2020**, *25*, 87–96. [\[PubMed\]](#)
7. Galeano-Brajones, J.; Carmona-Murillo, J.; Valenzuela-Valdés, J.F.; Luna-Valero, F. Detection and Mitigation of DoS and DDoS Attacks in IoT-Based Stateful SDN: An Experimental Approach. *Sensors* **2020**, *20*, 816. [\[CrossRef\]](#) [\[PubMed\]](#)
8. Xu, C.; Lin, H.; Wu, Y.; Guo, X.; Lin, W. An SDNFV-Based DDoS Defense Technology for Smart Cities. *IEEE Access* **2019**, *7*, 137856–137874. [\[CrossRef\]](#)
9. Yin, D.; Zhang, L.; Yang, K. A DDoS Attack Detection and Mitigation with Software-Defined Internet of Things Framework. *IEEE Access* **2018**, *6*, 24694–24705. [\[CrossRef\]](#)
10. Bawany, N.Z.; Shamsi, J.A.; Salah, K. DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions. *Arab. J. Sci. Eng.* **2017**, *42*, 425–441. [\[CrossRef\]](#)
11. Suh, J.; Choi, H.-g.; Yoon, W.; You, T.; Kwon, T.; Choi, Y. Implementation of a content-oriented networking architecture (CONA): A focus on DDoS countermeasure. In Proceedings of the European NetFPGA Developers' Workshop, Cambridge, UK, 9–10 September 2010.
12. Chu, Y.; Tseng, M.; Chen, Y.; Chou, Y.; Chen, Y. A novel design for future on-demand service and security. In Proceedings of the IEEE 12th International Conference on Communication Technology, Nanjing, China, 11–14 November 2010; pp. 385–388.
13. Braga, R.; Mota, E.; Passito, A. Lightweight DDoS flooding attack detection using NOX/OpenFlow. In Proceedings of the IEEE 35th Conference on Local Computer Networks (LCN), Denver, CO, USA, 10–14 October 2010; pp. 408–415.
14. Zhang, Y. An adaptive flow counting method for anomaly detection in SDN. In Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies—CoNEXT '13, Santa Barbara, CA, USA, 9–12 December 2013; pp. 25–30.
15. Moustafa, N.; Slay, J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In Proceedings of the Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 10–12 November 2015; pp. 1–6.
16. Doshi, R.; Apthorpe, N.; Feamster, N. Machine learning DDoS detection for consumer internet of things devices. *arXiv* **2018**, arXiv:1804.04159.

17. Alomari, E.; Manickam, S.; Gupta, B.; Singh, P.; Anbar, M. Design, deployment and use of HTTP-based botnet (HBB) testbed. In Proceedings of the 16th International Conference on Advanced Communication Technology (ICACT), Pyeongchang, Republic of Korea, 16–19 February 2014; pp. 1265–1269.
18. Livadas, C.; Walsh, R.; Lapsley, D.; Strayer, W.T. Using machine learning techniques to identify botnet traffic. In Proceedings of the 31st IEEE Conference on Local Computer Networks, Tampa, FL, USA, 14–16 November 2006.
19. Bhatia, S.; Schmidt, D.; Mohay, G.; Tickle, A. A framework for generating realistic traffic for distributed denial-of-service attacks and ash events. *Comput. Secur.* **2014**, *40*, 95–107. [\[CrossRef\]](#)
20. Sharafaldin, I.; Lashkari, A.H.; Ghorbani, A.A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy, ICISSP 2018, Funchal, Portugal, 22–24 January 2018.
21. Jagtap, M.M.; Saravanan, R.D. Intelligent software defined networking: Long short term memory-graded rated unit enabled block-attack model to tackle distributed denial of service attacks. *Trans. Emerg. Telecommun. Technol.* **2022**, *33*, e4594. [\[CrossRef\]](#)
22. Negera, W.G.; Schwenker, F.; Debelee, T.G.; Melaku, H.M.; Feyisa, D.W. Lightweight Model for Botnet Attack Detection in Software Defined Network-Orchestrated IoT. *Appl. Sci.* **2023**, *13*, 4699. [\[CrossRef\]](#)
23. Jarraya, Y.; Madi, T.; Debbabi, M. A Survey and a Layered Taxonomy of Software-Defined Networking. *IEEE Commun. Surv. Tutor.* **2014**, *16*, 1955–1980. [\[CrossRef\]](#)
24. Node-RED Tool. Available online: <https://nodered.org/> (accessed on 4 August 2023).
25. Moustafa, N. The Bot-IoT Dataset. IEEE Dataport. 16 October 2019. Available online: <https://ieee-dataport.org/documents/bot-iot-dataset> (accessed on 25 April 2023). [\[CrossRef\]](#)
26. Koroniotis, N.; Moustafa, N.; Sitnikova, E.; Turnbull, B. Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset. *Future Gener. Comput. Syst.* **2019**, *100*, 779–796. [\[CrossRef\]](#)
27. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; ACM: New York, NY, USA, 2016; pp. 785–794.
28. Behal, S.; Kumar, K. Detection of DDoS attacks and ash events using information theory metrics—An empirical investigation. *Comput. Commun.* **2017**, *103*, 18–28. [\[CrossRef\]](#)
29. Eclipse, Mosquitto MQTT Broker. Available online: <https://mosquitto.org/> (accessed on 4 December 2021).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.