Detection of DDoS attacks in the IoT devices Using Machine Learning Models on Urban IoT Dataset

MSc. Research Project
CyberSecurity

Simon Onyebuchi Obetta
Student ID: x19152272

School of Computing
National College of Ireland

Supervisor: Arghir Nicolae Moldovan

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Simon Onyebuchi Obetta		
Student ID:	x19152272		
Programme:	Cybersecurity	**Year:**	2021
Module:	MSc. Research Project		
Supervisor:	Arghir Nicolae Moldovan		
Submission Due Date:	16th December 2021		
Project Title:	Detection of DDoS attacks in the IoT devices using Machine Learning Models on Urban IoT Dataset		
Word Count:	7640	**Page Count**	22

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: [illegible]

Date: 30th January 2022

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Detection of DDoS attacks in the IoT devices Using Machine Learning Models on Urban IoT Dataset

Simon Onyebuchi Obetta
x19152272

Abstract

As the Internet of Things (IoT) has grown in popularity in recent years, attackers are increasingly targeting IoT environments to perform malicious attacks such as DDoS. This is due to the inadequate security implementation and management of IoT devices. Sometimes, the infected IoT devices can be used as bots by attackers to launch a DDoS attack on a target. Although various security methods have been introduced recently for IoT devices, an effective detection method is still required. This motivates the development of four machine learning models for DDoS detection. Using three modern neural network algorithms such as Feedforward Neural Network (FNN), Deep Neural Network (DNN), and Autoencoder and the conventional Random Forest for DDoS detection performance comparison. The detection system uses a public dataset, Urban IoT to detect attacks using these four algorithms. Experiment results show that DNN achieved the highest accuracy of 95.9%, while the Random Forest achieved the lowest accuracy of 94.2%.

1 Introduction

The Internet of Things (IoT) is a new technology innovation that connects smart electronic devices and gadgets to the internet for data collecting and transfer without human intervention (Steve, 2020). Presently a large number of IoT systems are interconnected with several sensors and maintain communication and exchange massive volumes of data. This is due to an increase in technological advancements. For instance, in the context of smart-home applications, large-scale IoT systems with numerous sensor nodes are being used and proposed. Most common network architectures that utilize IoT services include healthcare systems, institutions, organizations, and home network systems. For communication between the IoT devices and the controller, the majority of IoT implementations in smart homes rely heavily on home internet networks, either wireless or cable. IoT devices enable smarter and more efficient homes by allowing for automatic and remote control of household equipment. For example, modern CCTV cameras can now be monitored from afar using smartphones.

With security, vulnerabilities associated with IoT devices, and it has been predicted to grow with time. This is the reason IoT technology is expected to become a major research focus in the realm of cybersecurity. The most prevalent IoT security threats comprise code injection, middle-man attack, sinkhole, Sybil attack, Denial of Service (DoS), and Distributed Denial of Service (DDoS) Vashi *et al*. (2017). According to Cloudflare (2021), DDoS attacks occur when an attacker floods the target's network or application with fake requests from a compromised bot within a short period. It takes advantage of the vulnerabilities in the internet system infrastructures like unsecured ports, use of default passwords, lack of security updates, and so on to penetrate the targets' system (Douligeris and Mitrokotsa, 2004). The DDoS attackers aim to deny access to networks or applications from legitimate users by bringdown or slowing the network or application.

Comparing the difference in the performance of different machine learning algorithms (MLA) have shown potential in detecting malware especially DDoS in the Internet traffic in modern anomaly-based detection studies. However, this is not enough in detecting real-life or Slow DDoS attacks, especially in IoT devices.

This research aims to find an improvement in detecting DDoS attacks in IoT devices using the Urban IoT dataset. This will be performed by comparing the metrics performance of four MLA which include, simple Feedforward Neural Network (FNN), Deep Neural Network (DNN), Autoencoder Neural Network, and Random Forest. The first three belongs to Neural Network Machine Learning which is a branch of artificial intelligence, while the Random Forest is a traditional MLA. The goal is to generate and compare the results of the performance of the MLA on DDoS detection using the Urban IoT dataset. A model with high accuracy, precision, and recall rate will be proposed. These are some of the most important parameters in the neural network DDoS detection study right now.

1.1. Motivation

The IoT has grown in popularity as these technologies are employed for a variety of reasons while employing poor security standards. This increases the likelihood of DDoS attacks and other security threats. Because of insufficient security standards and policies, most security analysts regard IoT as a susceptible point for cyber-attacks (Tawalbeh *et al*., 2020). For example, the largest DDoS attack that happened in 2016 on the Dyn's company server leveraged IoT devices using default username/password and unsecured ports vulnerabilities available to the attacker to launch an attack to one of the major web hosting company's servers (Kumar and Lim, 2019). Because of these reasons, I have been motivated to carry out my research in this field. Nevertheless, numerous studies and procedures have been undertaken to detect and mitigate this from occurring, one of which is the employment of an MLA with excellent accuracy, recall, and precision.

1.2. Research Question

This paper aims to find a solution to these questions.
"How well do Machine Learning Models perform at detecting DDoS attacks on the Internet of Things Devices using Urban IoT dataset?"

This paper focuses on comparing different neural networks and Random Forest models for DDoS data classification. This strategy is based on the detection anomalies system. According to Tonkal *et al*. (2021), a neural network is a classification system that consists of multiple processing units called neurons. The units are organized by each layer. Each unit that occurs in a layer has a link to the previous layer. Random forest, on the other hand, is a traditional machine learning algorithm that works by building a decision tree out of many smaller trees. The results of each smaller tree are merged with a weighted value to generate an outcome using the bagging method (Pande *et al*. 2021).

1.3. Contribution

My research aims to make the following contributions:

- A comparison in performance of different machine learning algorithms in an attempt to detect DDoS attacks in an Urban IoT dataset.
- Build two approaches for the models' comparison; The first one is by using two different attack ratios, 0.8 and 1. While the second approach is by using different numbers of models to train 20 IoT nodes.

The following are the layouts of the paper. Firstly, the related research works done in DDoS detection are addressed in Section 2. The theoretical basis and structure of the proposed models are presented in Section 3. The test design is also presented in section 4. While the implementation and evaluation will be found in sections 5 and 6 respectively. The last section also contains the future work that can be considered.

2. Related Work

There have been numerous studies and efforts into using machine learning in DDoS detection. This section provides an overview of several public DDoS datasets as well as previous research papers on DDoS detection via machine learning methods.

2.1. DDoS Datasets

2.1.1. CIC DoS dataset (2017)

This is an application layer DDoS attacks dataset generated by the University of New Brunswick team and it is available in the Canadian Institute for Cybersecurity database (Hadian *et al*., 2017). The dataset is a slow-rate DoS attack dataset that frequently exhibits as slow transmit and slow read. The attacks happened very slow with a slight effect on the targeted system, as the basic assumption of low-rate DDoS attacks. The resulting application-layer DDoS attacks were mixed with attack-free traces from the ISCX-IDS dataset, yielding 8 distinct application-layer DoS attack traces. The final generated dataset is 4.6 GB in size and contains 24 hours of network traffic.

2.1.2. CIC-DDoS2019 dataset

This dataset is also available in the Canadian Institute for Cybersecurity database created by team University of New Brunswick (Sharafaldin *et al.,* 2019). The dataset includes benign and close-to real-life DDoS attacks. It also has different modern-day DDoS attacks such as UDP, SYN, DNS, and so on. The attacks were then carried out over some time, with 12 DDoS attacks on the training day and 7 attacks on the testing day. The DDoS traffic volume was so low, and the port scan had only been run on the testing day. The actual size of the dataset is 6.7Gb.

2.1.3. N-BaIoT dataset

Meidan et al. (2018) from the Department of Software and Information Systems Engineering Ben-Gurion University of the Negev acquired these IoT botnet data in raw network traffic data from 9 different IoT devices from two families. The IoT devices are; babies' monitors, security cameras, doorbells, and thermostats. They captured the traffics from these devices to the central switch in pcap format by using a port mirroring tool.

2.1.4. Urban IoT Data dataset

The Stevens Centre for Innovation team at the University of Southern California created their first dataset from a genuine Urban IoT system in a large metropolis, consisting of 4060 spatially distributed IoT nodes or sensors (Hekmatic *et al.*, 2021). The data comprises nodes' binary activity status at a granularity of 30 seconds over a month in a benign (non-attacked) environment. The original dataset contains the node ID, the geolocation coordinates, and the time of the IoT node's activity status for a record of 1 month.

2.2. Machine Learning Approach

A Machine learning algorithm is classified into four sections; supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning, and their implementation is grouped into traditional machine learning (shallow learning) and deep learning (Wei, *et al*., 2019). Several studies have been conducted using these two implementation groups for DDoS detection.

2.2.1. Traditional Machine Learning Approach in DDoS Detection

A machine learning classifier was proposed by Ashi, et al. (2020) to detect DDoS attacks with an emphasis on cloud computing architecture. After collecting 256 Uniform Resource Locators, the authors used four different systems to simulate a DDoS attack simultaneously (URLs). After that, a dataset comprising the simulation's network traffic flow was created. After the data had been pre-processed and assessed, the Random Forest (RF) technique was utilized for model testing.

Rahman, *et al.* (2019) created an SDN framework to identify and defend against DDoS attacks on the controller and the switch. To predict DDoS attacks, this framework requires training a machine learning model with recorded data. The mitigation script then uses the prediction to make decisions on their SDN network. With an open-source DDoS dataset, they tested and compared the results for Support Vector Machine (SVM), K-Nearest neighbours (K-NN), J48, and Random Forest. The results of their experiment revealed that J48 is the best classifier for their network with accuracy, F-1, and recall rate of 100%.

Reddy and Thilagam, (2020) recommended using the Naive Bayes classifier to detect DDoS attack traffic by taking into account the five most influential DDoS attack network factors. Based on the probability of the DDoS attack value, the proposed DDoS attack classifier is applied on all monitor nodes to process valid traffic and remove DDoS attack traffic. According to their simulation results, the proposed strategy reduces the intensity of DDoS attacks and allows network nodes to handle up to 80% of legal traffic.

Misbahuddin and Zaidi, (2021) classified DDoS attacks by using a semi-supervised machine learning approach on the CICDS2017 dataset. They began with unlabelled traffic information collected against three aspects for victim-end defence, namely the webserver. Two distinct clustering methods cluster the unlabelled data, and a voting procedure determines the final classification of traffic flows. To detect DDoS attacks, the supervised learning algorithms K-Nearest Neighbors (K-NN), Support Vector Machine (SVM), and Random Forest (RF) is applied to labelled data. The accuracy scores were 95%, 92%, and 96.66% respectively in the experiments.

Rios, *et al.* (2021) tested and compared the Multi-Layer Perceptron (MLP), K-Nearest Neighbors (K-NN), Support Vector Machine (SVM), and Multinomial Naive Bayes (MNB) machine learning methods for detecting reduction of quality (RoQ) attacks. They also suggested a method for detecting RoQ attacks that combines three models: Fuzzy Logic (FL), Multi-Layer Perceptron (MLP), and Euclidean Distance (ED). They tested these methods using both simulated and real-world traffic patterns. They demonstrated that using three parameters, namely the number of packets, entropy, and average inter-arrival time, results in the better categorization of the four machine learning algorithms than using only entropy. And found that MLP outperforms the other four machine learning algorithms when it comes to detecting RoQ attacks.

Doshi, *et al*. (2018) proposed the approach of using multiple machine learning algorithms like K-Nearest neighbours (K-NN), Linear Support Vector Machine (LSVM), Decision Tree (DT), Random Forest (RF), and Neural Network (NN) for DDoS detection for the consumer IoT on the generated dataset. Their classification algorithm was based on the idea that system traffic conditions from these IoT nodes differ from those from well-studied non-IoT network nodes. They used data from a consumer IoT device that included both normal and DoS attack traffic to test five different machine learning classifiers. The results show variations in accuracy, F1, recall, and precision across the models. With K-NN, DT, RF, and NN having 99.9% accuracy while LSVM is 99.1%.

Singh, (2021), proposed a comparative analysis of the DDoS detection method using machine learning algorithms. The author adopted Support Vector Machine (SVM) using linear and Radial Blasts function kernel, Decision Tree (DT), k-Nearest Neighbors (K-NN), Multi-layer Perceptron (MLP), Gaussian Naïve Bayes (GNB), and Random Forest (RF). For the simulation of this approach, Minimet emulation software was used while IPv4, Transmission Control Protocol (TCP), Internet Control Message Protocol (ICMP), and user Diagram protocol (UDP) were used for data extraction. SVM emerged as the most effective method for detecting DDoS attacks, as the result show accuracy, precision, and recall of 100%. K-NN on the other hand gave the slowest rate out of all the machine learning algorithms used.

Pérez-Díaz1 *et al*. (2020) proposed a flexible SDN-Based Architecture for the identification and mitigation of low-rate DDoS attacks. SDN framework enables more flexible and manageable environments by decoupling control and data planes. The adopted approach makes it possible for machine learning to fully make use of GPU which increases training and classification speed. Their research was divided into two phases. The first phase is the intrusion prevention system (IPS) which consists of the flow management module which detects HTTP flows, the suspicious attackers' management for managing the blacklist of potential attackers, and the mitigation management module. The second phase is the intrusion detection system (IDS) which consists of the identification API, the machine learning model selection, and the identification. The architecture simulation was carried out using the Minimet simulator. From the result, Multi-Layer Perceptron (MLP) gave an accuracy of 95%.

Ali *et al*. (2020) presented DDoS detection using Support Vector Machine (SVM) and the SVM utilizing PCA filters on 200 DARPA datasets including solely DDoS attacks and 1998 DARPA datasets containing normal traffic. Because SVM takes a longer time to train and develop the detection model, the authors proposed a novel architecture that uses PCA for dimensionality reduction. An approach for determining the best value for principal component selection, which will increase the SVM's performance in detecting DDoS attacks.

Mishra *et al* (2021) proposed a classification-based machine learning for detecting DDoS attacks in cloud computing. The machine learning algorithms adopted for classification were K-Nearest Neighbour (K-NN), Naïve Bayes (NB), and Random Forest (RF). They generated a long feature vector by merging all feature vectors of interest. Their focus was more on supervised learning with the Random Forest having the best performance of 99.58% while Naive Bayes and K-NN having 93.69% and 97.89% respectively.

Chen *et al*. (2020) presented a multi-layer DDoS detection system using Decision Tree (DT) machine learning for DDoS prevention in IoT gateways by numerically extracting aspects of four types of DDoS attacks, including sensor data flood, ICMP flood, SYN flood, and UDP flood. They demonstrate that the multi-layer DDoS detection system can accurately segregate normal packets from DDoS attack packets from IoT devices by launching DDoS attacks from eight smart poles in a genuine IoT scenario. The proposed system can detect DDoS attacks with a 97% accuracy using DT.

Pande *et al*. (2021*)* proposed the use of a Random Forest (RF) algorithm to detect DDoS attacks in the generated DDoS attack dataset. The trained model resulted in 99.76% of correctly classified instances.

Hekmatic *et al. (*2021) proposed a simple Feed-forward Neural Network for DDoS detection employing 20 nodes out of 4060 in the original dataset for the Urban IoT DDoS Data-main dataset. They also provide a script for creating a benign dataset from the original dataset to eliminate bias toward nodes with higher activity. The authors used attacked emulator to generate an artificial DDoS attack for the attack ratio of 1 on the 20 selected IoT nodes. They also generated the train and test datasets from the generated attacked data. The simple Feedforward Neural Network was applied on the train and test dataset with a mean accuracy of Testing and Train data of 94% and 88%, respectively.

2.2.2. Deep Learning Approach in DDoS Detection

Abdullah et al. (2020) used the CICDDoS2019 dataset which is divided into two categories: reflection and exploitation to propose the detection of DDoS attacks with a Feed-

Forward Deep Neural Network algorithm. The model's accuracy for DDoS detection and classification is 99.9% for dataset1 and 94.5 percent for dataset2.

Shaaban et al. (2019) proposed the use of a Convolutional Neural Network (CNN) for DDoS detection. For their research, the authors used two datasets: the generated dataset and the NSL-KDD dataset. CNN is used for the classification of normal traffic from DDoS attacks. According to the study and results, CNN performance for the classification is 99% accurate. This result was compared other machine models like Decision Trees (DT), Support Vector Machines (SVM), K-Nearest Neighbors (K-NN), and Neural Networks (NN). the results of the comparison reveal that CNN is better at detecting DDoS.

Ray *et al.* (2021) proposed a privacy-preserving methodology for detecting IoT malware that uses Federated Learning (FL) to train and analyse supervised and unsupervised models without exposing sensitive data in the N-BaIoT dataset. The result performance was compared as follow based on the model architecture: 1) A federated approach, in which each device owner trains their model, which is then aggregated in a server regularly; 2) A non-privacy-preserving setup, in which the entire dataset is centralized and trained by the server; 3) A local setup, in which each device owner trains one isolated and individual model. The utilization of more diversified and larger data, as done in the federated and centralized techniques, has a significant favourable impact on model performance in both supervised and unsupervised scenarios, according to this comparison.

Soe *et al.* (2019) proposed the use of an Artificial Neural Network (ANN) for DDoS detection on the Bot-IoT dataset. The authors recommend using an ANN model for DDoS detection to achieve reliable detection performance. The attempt was to overcome the data imbalance problem. Before using neural network design, they applied the data re-sampling approach, Synthetic Minority Over-sampling Technique (SMOTE). From the result of the evaluation, the system is capable of identifying DDoS attacks with a single hidden layer and single output node artificial neural network with 100% accuracy.

Meidan *et al*. (2018) and Tsimbalist, (2019) proposed using an Autoencoder deep learning model to detect Mirai and BASHLITE attacks in the N-BaIoT dataset containing 9 IoT devices. They presumed that the predictability of traffic behaviour on IoT devices in the dataset may be immediately converted into anomaly detection performance measures. For example, an IoT device with a high level of traffic predictability would highlight any aberrant action, causing the True-Positive Rate (TPR) to rise and detection times to fall.

Table 1: Summary of DDoS detection research studies

Reference	Dataset	Metrics	Algorithms	Accuracy
Ashi *et al.*, (2020)	Generated dataset by simulating DDoS attack from 4 different servers to a target	Accuracy, Recall, Precision, and F1-Score	RF	100%
Rahman *et a*l. (2019)	The simulation was done on a Virtual machine using Tshark to capture traffic in the RYU controller	Accuracy, Recall, Precision, and F1-Score	J48	100%
			RF	100%
			SVM	100%
			K-NN	100%
Misbahuddin and Zaidi (2021)	CICIDS2017	Accuracy and Precision	K-NN	95%,
			SVM	92%
			RF	96.66%
Rios *et al*. (2021)	Generated dataset captured from the emulated environment	FP, TP, TN, FN, Recall, F1-score, and precision	MLP	99.87%
			K-NN	99.58%
			SVM	99.49%
			MNB	96.02%
Doshi *et al. (2018)*	Generated dataset captured from IoT-specific traffic in gateway router	Precision, Recall, F1-score	K-NN	96.7%
			LSVM	92%
			DT	97.7%
			RT	98.1%
			NN	93.9%
Singh (2021)	SDN-DDoS (ICMP, TCP, UDP)	Accuracy, Precision, recall, and F1-score	RSVM	100%
			LSVM	100%
			KNN	95.6%
			DTC	89.7%
			RFC	80.7%
Pérez-Díaz1 *et al*. (2020)	CAIDA DDoS 2007	Precision, recall, F1-score, and false alarm rate	RF	90.4%
			MLP	95%
			SVM	93.0%
Ali *et al.* (2019*)*	DARPA	FP, Training Time, and Accuracy	SVM	95.1%
			PCA+SVM	97.5%
Mishra *et al.* (2021)	Generated dataset captured virtual machines in the cloud environment	FP, FN, Recall, F1-score, and precision	NB	93.58%
			K-NN	97.69%
			RF	99.68%

Chen *et al.* (2020)	Generated dataset from IoT devices	True-positive (TP), False-Positive (FP), False-Negative (FN), and True-Negative (TN)	Sensor Data flood	97.39%
			Network data flood	99.98%
Hekmatic *et al.* (2021)	Urban_IoT_DDoS_Data-main	Mean Accuracy and Mean Recall	NN	94%
Soe *et al.* (2019)	Bot-IoT dataset	Precision, Recall, and Accuracy	ANN	100%
Abdullah *et al.* (2020)	CICDDoS2019	Precision, Recall, and Accuracy F-Score	DNN Dataset1	99.9%
			DNN Dataset2	94.5%
Shaaban *et al.* (2019)	Generated dataset and the NSL-KDD dataset	Accuracy and Loss	CNN	99%
Pande *et al.* (2021)	Generated dataset	TP rate, FP rate, Precision, Recall	RF	99%

3. Methodology

This section covers the proposed methodology for detecting DDoS attacks using, simple Feed-Forward Neural Network (FNN), Deep Neural Network (DNN), Autoencoder Neural Network, and Random Forest (RF). Figure 1 below shows the steps performed starting from dataset selection, data pre-processing, models classifiers, and models evaluations.

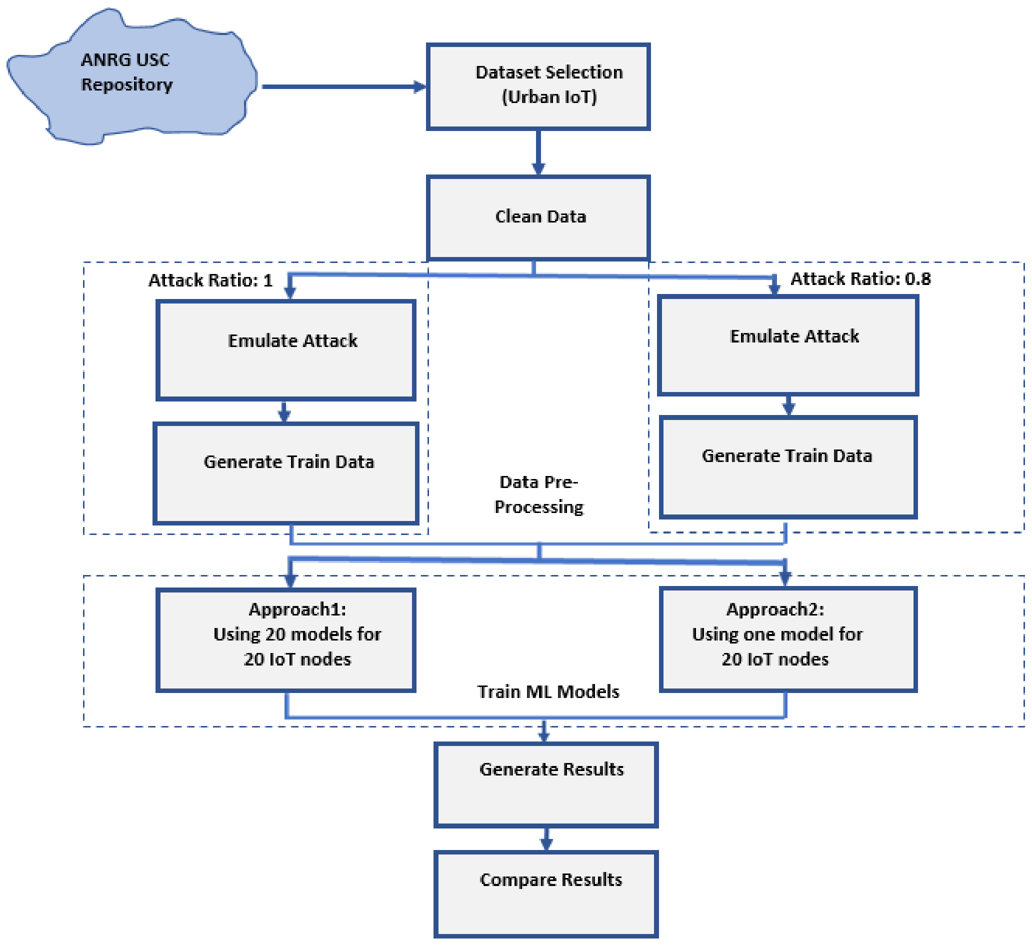

Figure 1: Process flow of the methodology

3.1. Dataset Selection

The dataset used to perform this research work is Urban IoT. It was downloaded online in an open-source ANRG USC GitHub repository (github.com). The original data was captured from the activity status of genuine event-driven IoT nodes installed in a city (Hekmatic *et al.*, 2021). The data captured is the activity of 4060 IoT devices (nodes) for one month. The dataset was used to perform this research because it is a very realistic dataset from IoT networks, which is appropriate for the proposed research question. In addition, it is very recent and was published on the GitHub website on 14th September 2021. The only paper publishment that used the dataset to conduct DDoS detection using simple Feedforward Neural Network was published recently on ACM on 15th November 2021 (Hekmatic *et al.*, 2021). As a result, there are open researches in this dataset that need to be done.

The original dataset contains the following records; node ID, node's geolocations (Latitude and Longitude), and timestamp of the node's activity status. The dataset also comprises each node's binary activity status at 30 seconds throughout a month in a benign (non-attacked) environment. When a node's activity status changes, a record is appended to the original dataset. It was later supplemented with artificial attack emulation to make it usable for training machine learning models for DDoS detection.

From the dataset statistics, up to 65% of the nodes are activated at the mid-day, but by midnight, only approximately 20% of the nodes are active as shown in figure 2 below.

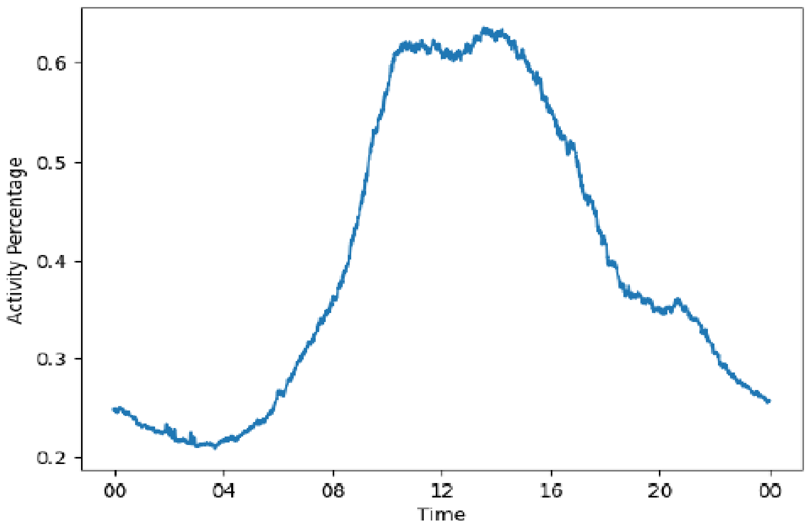

Figure 2: Active Nodes Percentage vs Time of the day

3.2. Data Cleaning

Data cleaning is part of the pre-processing activity carried out on the machine learning datasets. The reason why data cleaning is required in this method is that it has the potential to dramatically increase the efficiency and effectiveness of the training process. To emphasize the role of data cleaning in modern Machine Learning processes, it has been discovered that even when utilizing robust statistical techniques, the data cleaning methodology chosen can have a considerable impact on overall results (Krishnan, *et al*, 2016). It comprises work such as removing extraneous data, dealing with missing values, label conversion, categorization, and data standardization. For example, this paper concentrated on 20 IoT nodes, therefore only important features were retained and extraneous ones were removed. Following the identification of significant characteristics, the next duty in the pre-processing step is to emulate the attack.

3.3. **Attack Emulator**

To enhance the dataset an artificial attack emulation is required to make it suitable for training machine learning models for DDoS detection. By modelling and emulating current and emerging DDoS attacks and the detection techniques, I believe it is possible to provide an answer to the best DDoS attack detection method. It is critical to employ appropriate models and emulation that allow researchers to thoroughly study various modes of DDoS attacks and detection mechanisms, implement novel approaches and assess detection performance (False-Negatives, False-Positive Precision, Recall, and Accuracy).

Emulation allows for the use of real nodes in an active state to carry out a DDoS attack. It creates real-time legitimate attack traffic to analyse attacks. Although emulation provides a realistic environment, it does have several drawbacks, such as a lack of hardware diversity, a lengthy setup time, and code errors.

3.4. Train Machine Learning Models

Machine learning is a subset of Artificial Intelligence that allows a computer program to learn from massive amounts of historical data. When the software has analysed and processed the data, it can create future predictions (Merlin, 2018). The classification models employed in the experiment included one of the traditional machine learning algorithms, Random Forest (RF), and three neural network models; Feedforward Neural Network (FNN), Deep Neural Network (DNN), and Autoencoder. Each method has its hyper-parameter, which will be illustrated in section 5.

Feed-Forward Neural Network Model: This is a type of Neural Network with a single hidden layer. It is the simplest type of neural network (Goyal *et al*, 2017). The input layer sends a multi-dimensional request to the single hidden layer and is processed using a weighted summation and an activation function. It is trained using labelled data and a learning algorithm that optimizes the summation model's weights. The hidden layer is linked to the input layer and the output layer link to the hidden layer (Dertat, 2017).

The aim of using this model is to reproduce the DDoS detection done by Hekmatic et al., (2021) using the same dataset (Urban_IoT). The major difference between the FNN and the other two neural networks used is that it has only one hidden layer of units (Jurafsky and Martin, 2021). The model's inability to deal with the complex dataset is the reason I thought of other models with multiple hidden layers that can produce better performance than the original work described below.

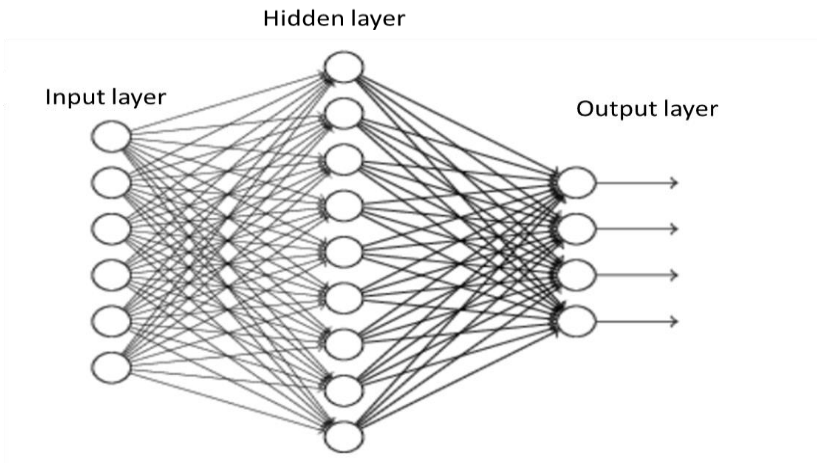

Figure 3: Simple Feed-Forward Neural Network Architecture (Nielsen, 2019)

Deep Neural Network Model: This model is made up of Feedforward Neural Networks (FNN) that do not have any feedback connections. Just like the FNN, the DNN consists of the input and output layers, as well as the hidden layers (more than one). Each layer contains units with weights. The activation processes of the units from the previous layer are carried out by these units (Pande *et al*., 2021).

Because the DNN model's structure combines feature extraction and classification operations, it benefits from both supervised and unsupervised learning (Pande *et al*., 2021).

The DNN model in Figure 1 has an input layer with the same unit as the selected feature. For the input layer, the ReLU activation function is utilized. The second layer in the DNN model is the hidden layers which can be more than one. Each hidden layer also uses the ReLU activation. The output layer normally uses the Sigmoid or Softmax activation function.

DNN is used to train the model because it can extract different abstraction levels at different processing layers without requiring human interaction, which is one of its primary advantages over traditional machine learning (Ortet Lopes *et al.*, 2021). In addition, the multiple hidden layers architecture can automatically uncover complex correlations and mappings from input to output data that are not compatible with other none deep neural networks. This can lead to an increase in the overall performance.

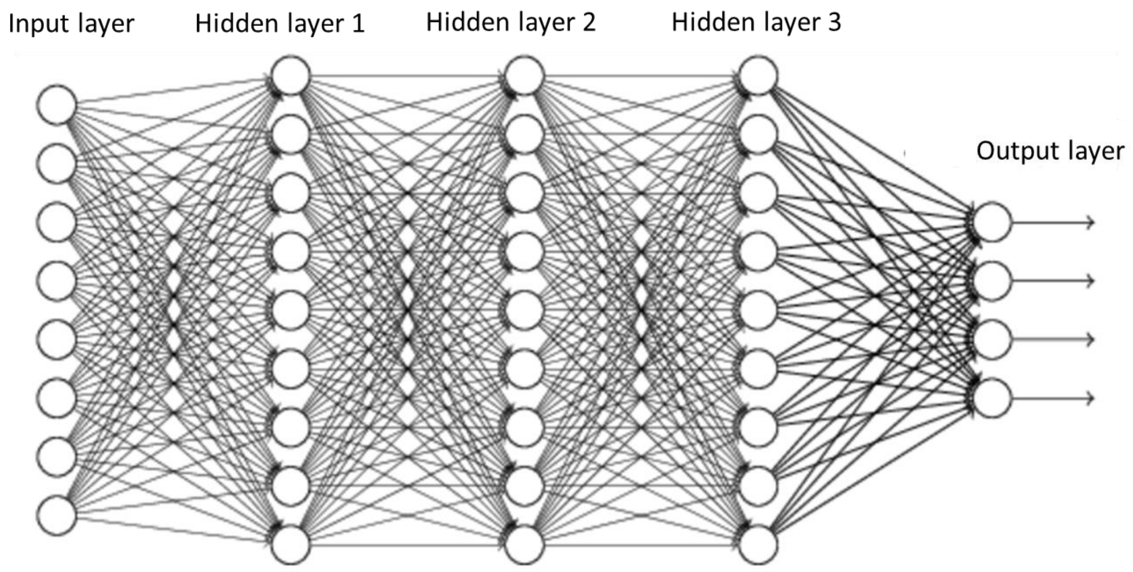

Figure 4: Deep Neural Network Architecture (Nielsen, 2019)

Autoencoder Neural Network Model: This is a type of neural network that shrinks multidimensional input data within a hidden region before reconstructing the data from the hidden region (Ozgur and Fatih, 2019). Figure 5 depicts a simple autoencoder with one n-unit input layer, three hidden layers of varying units, one reconstructs layer with n units, and an activation function that can be any nonlinear function. For example, in this research, the tanh activation function was used for model training. The autoencoder is divided into encoder, code, and decoder. The encoder is the region that sits between the input layer and the hidden layer. The encoding region enables the reduction of multidimensional data to a lower size. The decoding region on the other hand is located between the hidden space layer and the output layer. And the code region is between the encoder and decoder. By increasing the size of the shrunk hidden layers, the decoder attempts to reconstruct the input.

The reason why an Autoencoder model is used is that it is also a multi-hidden layer Neural Network just like DNN which can uncover complex correlations and mappings of data and increase the metrics performance. For example, Ozgur and Fatih, (2019) used this model to propose DDoS attack detection in their study using the kdd99 dataset because the model has an advantage in terms of removing outliers and fixing complexes in a dataset

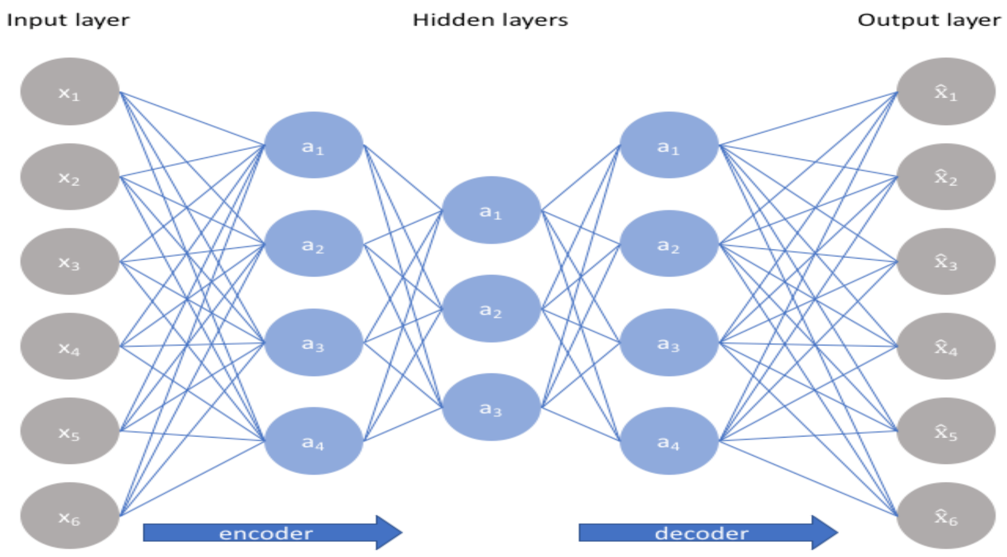

Figure 5: Autoencoder neural network architecture (Jordan, 2018)

Random Forest Model: This is one of the traditional machine learning algorithms that is based on the construction of numerous small trees in a decision tree (Pande et al. 2021). By using a bagging method, the results of each small tree are combined with a weighted value to provide a final prediction outcome. To reach the final predicted conclusion, this approach employs the mean of the individual small trees.

The project, analysis is on the binary classification of the dataset (Y-value = 1 or 0), which is a type of supervised learning in which an algorithm tries to figure out which group an input belongs to. According to Mishra et al., (2021), Random Forest is recommended for the supervised learning approach since it produces much better results than other machine learning algorithms. This is because Random Forest is less prone to overfitting than the alternative Decision Tree since it employs an ensemble of Decision Trees, with the values in the tree being a random, independent sample.

3.5. Generate Models Results

The results are based on the metrics used to determine which of the four models achieved the best performance in the detection of DDoS attacks. They are; True-Positive (TP) and True-Negative (TN), False-Positive (FP) and False-Negative (FN), Accuracy, and Recall. The metrics derived from these elements are explained in the following sections.

True-Positive and True-Negative: A True-Positive result is one in which the model accurately predicts the positive result. A true negative, on the other hand, is a result in which the model predicts the negative result.

False Positive and False-Negative: A false positive occurs when the model forecasts the positive class inaccurately. While a false negative on the other hand is an outcome in which the model forecasts the negative class inaccurately.

Accuracy: The metric measures the model's accuracy, or all right predictions. The model can correctly anticipate both positive and negative outcomes.

Accuracy = (TN + TP) / (TN + TP + FP + FN) (1)

Recall: This metric measures the performance of a positive result prediction. It may be similar to Accuracy only that this measures the positive results.

Recall = TP/ (TP + FN) (2)

Precision: It is the percentage of correctly predicted True Positive value. It calculates the size of the relevant instance that is returned after classification has been done. A model with a high precision generates more accurate results than one with lower precision.

Precision = TP/(TP + FP) (3)

4. Design Specification

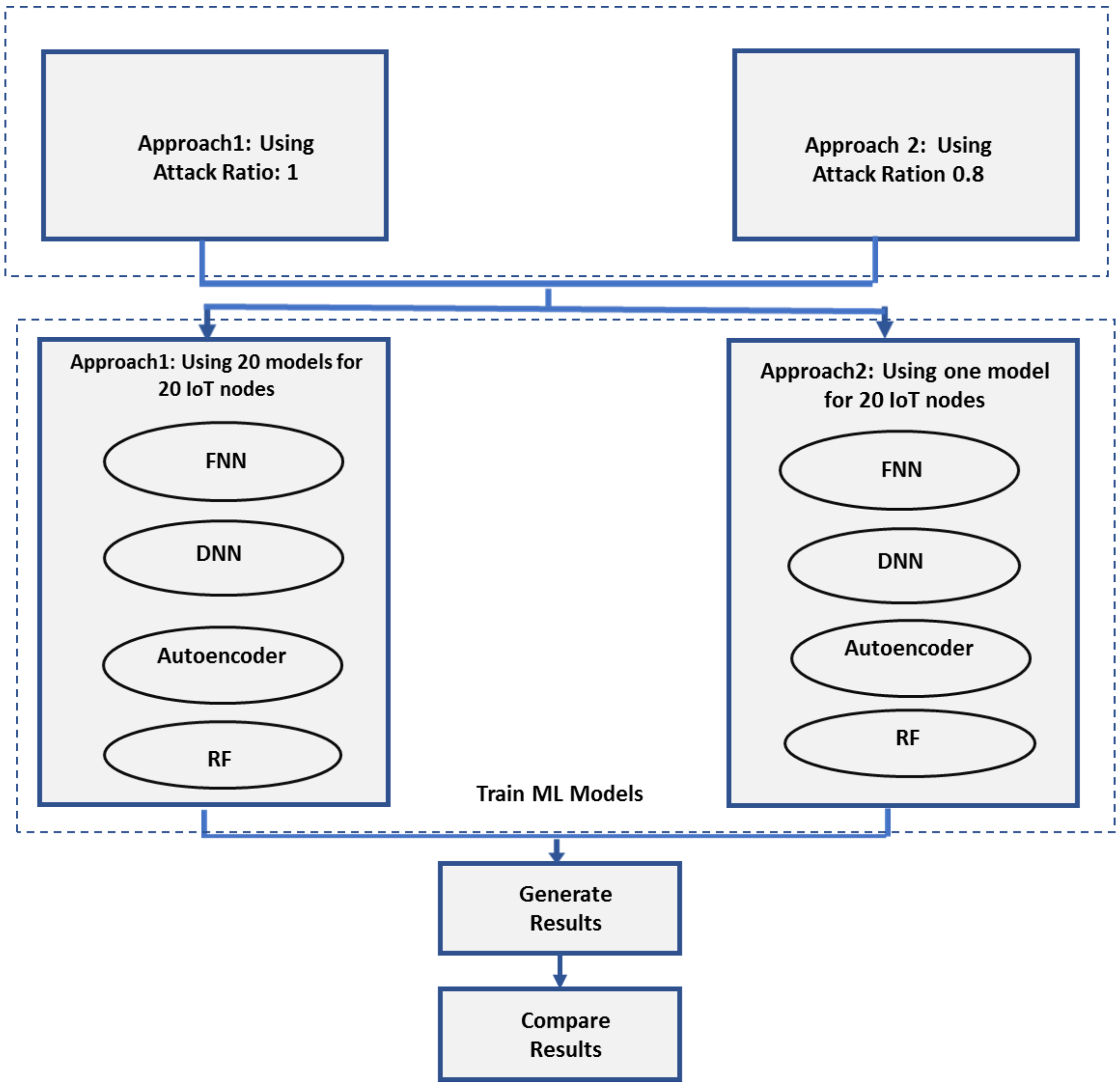
Approach1: Using Attack Ratio: 1
Approach 2: Using Attack Ration 0.8
Approach1: Using 20 models for 20 IoT nodes
FNN
DNN
Autoencoder
RF
Approach2: Using one model for 20 IoT nodes
FNN
DNN
Autoencoder
RF
Train ML Models
Generate Results
Compare Results

Figure 6: Project Implementation design

4.1. Data Pre-Process Design

At this stage, the design carried out are by employing two attack ratio approaches during attack emulation. The first approach is by using an attack ratio of 1 while the second approach has an attack ratio of 0.8. This is the ratio of the IoT nodes subjected to DDoS attacks. The reason for using high attack ratios is to have sufficient datasets for realistic and appropriate model analysis.

4.2. Machine Learning Model Design

Following the data selection and pre-processing for the two attack ratio groups, this paper recommended comparing the performance of DDoS detection in the generated trained and test datasets using four different machine learning models in two different techniques, as shown in figure 6.

Approach 1: This approach is to carry out the implementation for the four algorithms using 20 models for the 20 IoT nodes.

Approach 2: This approach is to carry out the implementation for four algorithms using one model for the 20 IoT nodes.

The reason for selecting these models and the design is to have a comprehensive performance comparison across the platforms.

5. Implementation

5.1. Tools Used

The project codes were written in Python version 3.7.9. This is because, python is a lightweight, adaptable, and easy-to-use programming language that can handle complicated scripting. It also provides code that is both concise and readable more suitable for machine learning projects than other programming languages like Java. For the step-by-step implementation of the codes, I used Spyder IDE 5.1.5 and Anaconda prompt. They are also lightweight software, which means it is faster and uses fewer system resources during code execution.

In addition, the test environment was performed on a Lenovo laptop on Windows 10 Pro operating system with 8.00GB of RAM.

5.2. Data Cleansing

This was implemented after the original dataset has been downloaded. The goal of this task was to create a benign dataset by selecting 20 IoT nodes at random from the original dataset over sometime. As a result, the original dataset was referenced and called in the python clean scripts. The next was to specify the beginning and end dates for the dataset that was generated. And also, setting the time step. This is the time between the beginning and the end date. The total number of IoT nodes to be trained was also set. In this research, 20 IoT nodes were used. After the script has been developed, it was executed in the Anaconda

Prompt, and benign data containing 20 IoT nodes for one-month activities records were generated and saved. The records include the IoT node's geo-locations, time, active status, and attack status. The count plot of the active state of the benign data is shown below. The number "0" represents inactive nodes, while "1" represents active nodes.

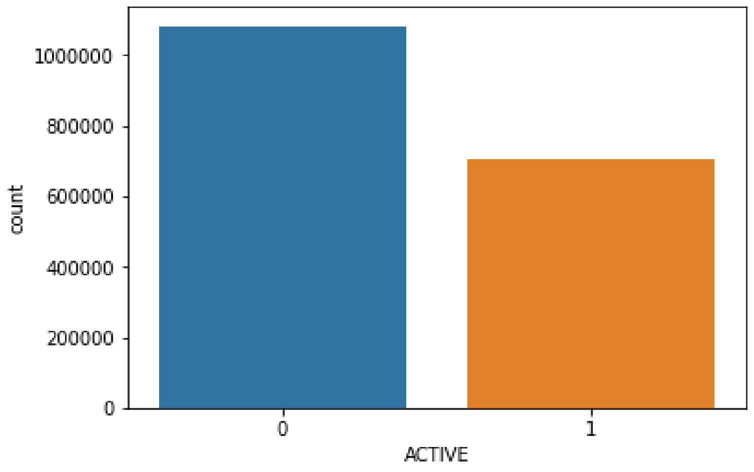

Figure 7: Node's active status's count plot

5.3. Attack Emulator

In this section, a DDoS attack was artificially initiated on the generated benign dataset. Although the focus was on active status nodes alone because the efficacy of detecting DDoS attacks is difficult from a single node's traffic. This attack took into account the ratio of nodes under attack, the attack length, and the attack start date. The python script generated attacks that contains the following:

Benign data: This is the dataset used for the attack.

Attack duration: This is the duration of the attack which is set to have occurred for 1, 2, 4, 8, and 16 hours every day on each node for one week of one month (80% train and 20% test data).

The attack begins and attacks end date: This is the begin/end date of the attacks. In this work, the start and end date of the attack is from 2nd January 2021 to 10th January 2021.

Attacked ratio nodes: This is the ratio of the nodes in the benign dataset to be attacked. In this research design, two attack ratio groups were set independently. They are 0.8 and 1.

The attack starts times: The start times of the attacks within the attack begin and attack end date. The attack started at 2 AM every day and the end time depend on the duration.

The output generated six attached data files on the train folder and test folders with the following records, Node ID, geolocation, time of the attack, node active status, attacked status, beginning and end date number of nodes, attacked ratios, and attacked duration.

5.4. Generate Train and Test dataset

After the attacked dataset has been generated, labelled training and test datasets were created from it by computing the average activity time of the IoT nodes in the chosen time intervals. To compute the mean activity time of the nodes, in this research I analysed a list of 12 different time frames, namely 1, 10, 30, in minutes interval and 1, 2, 4, 8, 16, 24, 30, 36, 42, 48 in hours interval. The result generated train and test datasets to be used for model analysis.

5.5. Train Machine Learning Models

As discussed, the machine learning models implementation is designed into two approaches. The first approach was by generating 20 models for the 20 IoT nodes. This was achieved by creating a for loop in the train models scripts for each node. The generated models' files were saved in separate folders using the node ID as the folder name. The second approach was by using a single model for the 20 IoT nodes. The loop function was removed and modified the training script was to generate the only model.

Simple Feed-Forward Neural Network Model: In this implementation stage, it consists of a 12-neuron input layer, followed by a single hidden layer with 8-neurons and ReLU activation. At the end of the hidden layer, a 20% dropout is employed, as well as batch normalization. A single neuron with the Sigmoid activation function is the output layer. To discover the attacked time slots in the dataset, the neural network model is trained for 500 epochs for each node.

Deep Neural Network Model: In this model, the input layer of the DNN implementation consists of 30 units with a ReLU activation function, followed by two hidden layers of 10 units each with a dropout of 0.4 and the ReLU activation function. The sigmoid activation function in the output layer.

Autoencoder Neural Network: The implementation, encoder is made up of three dense layers, which have 64, 32, and 16 units respectively, and the "tanh" activation function for each layer. The result of this encoder generates code that the decoder subsequently uses to reconstruct its input. The decoder on the other hand comprises three dense layers with the same units and tanh activation function.
The output function processes the result of the input function using a single layer sigmoid activation.

Random Forest: The random forest classifier uses 'n_samples=1000', 'n_features=20', 'random state=3' and n split=10, n repeats=3, and n jobs =-1 to measure the homogeneity supplied by each variable node and leaf. The outcome of the target variable is (0, 1), with the first index indicating the probability of the data being attacked and not attacked.

The results generated in each model are the initial and 20 nodes' model packages' or, initial and one model packages. It depends on the approach used.

5.6. Generate Result

The training and testing results have been created by merging data from time windows under DDoS attacks and normal conditions. The performance of the train and test were generated after the result generation script was executed for each 20 IoT nodes (accuracy, recall, and precision).

6. Evaluation

.

Following the generation of the results, it provides metrics such as Accuracy, Recall, and Precision for each of the IoT nodes. The average performance metrics were taken and compared among the four machine learning algorithms which were used to determine the best performing algorithm.

6.1. Experiment 1: Models Comparison for Attack Ratio 1

Table 2 is the mean results when using 20 training models for 20 IoT nodes while table 3 is the mean results when using one model for 20 IoT nodes for four machine learning algorithms when the attack ratio is 1.

Table 2: Comparison for 20 models on 20 IoT nodes

	Models	**Mean Accuracy**	**Mean Recall**	**Mean Precision**
Train dataset	FNN	0.943	0.936	0.794
	DNN	**0.959**	**0.942**	**0.842**
	Autoencoder	0.957	0.936	0.839
	RF	0.942	0.936	**0.842**
Test dataset	FNN	0.870	**0.835**	0.680
	DNN	**0.886**	0.824	**0.694**
	Autoencoder	0.883	0.791	0.687
	RF	0.870	**0.835**	**0.694**

Table 3: Comparison for one model on 20 IoT nodes

	Models	**Mean Accuracy**	**Mean Recall**	**Mean Precision**
Train dataset	FNN	0.834	**0.962**	0.661
	DNN	**0.874**	0.894	0.645
	Autoencoder	0.846	0.955	**0.666**
	RF	0.846	0.955	0.664
Test dataset	FNN	0.828	0.946	**0.699**
	DNN	0.452	0.875	0.640
	Autoencoder	**0.846**	**0.955**	0.664
	RF	0.823	0.839	0.643

6.2. Experiment 2: Models Comparison for Attack Ratios 0.8

This is also the mean performance results across the four machine learning algorithms for the second approach used. That is for the attack ratio of 0.8. Table 4 illustrates the mean results when using 20 training models for 20 IoT nodes while table 5 illustrates the mean results when using one model for 20 IoT nodes.

Table 4: Comparison for 20 models on 20 IoT nodes

	Models	**Mean Accuracy**	**Mean Recall**	**Mean Precision**
Train dataset	FNN	0.933	0.942	0.726
	DNN	0.956	**0.945**	0.789
	Autoencoder	**0.958**	0.937	0.802
	RF	**0.958**	0.938	**0.803**
Test dataset	FNN	0.857	**0.817**	0.582
	DNN	0.883	0.801	0.610
	Autoencoder	**0.885**	0.762	**0.616**
	RF	**0.885**	0.762	**0.616**

Table 5: Comparison for one model on 20 IoT nodes

	Models	**Mean Accuracy**	**Mean Recall**	**Mean Precision**
Train dataset	FNN	0.832	0.593	0.622
	DNN	0.854	**0.920**	**0.669**
	Autoencoder	**0.864**	0.905	0.591
	RF	0.854	**0.920**	0.569
Test dataset	FNN	0.827	0.624	**0.605**
	DNN	**0.832**	**0.917**	0.554
	Autoencoder	0.345	0.894	0.574
	RF	**0.832**	**0.917**	0.554

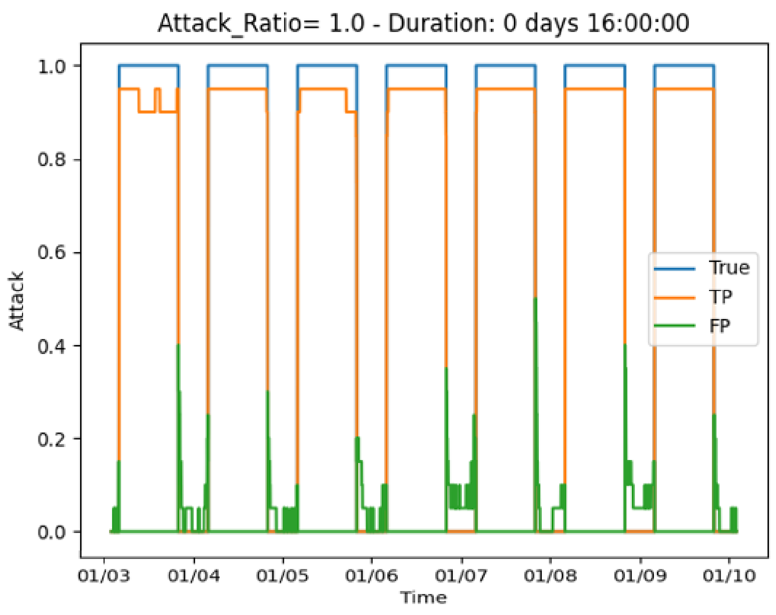

Figure 8: DNN training Dataset Attack Prediction vs Time for 1 attack ratio on 20 models

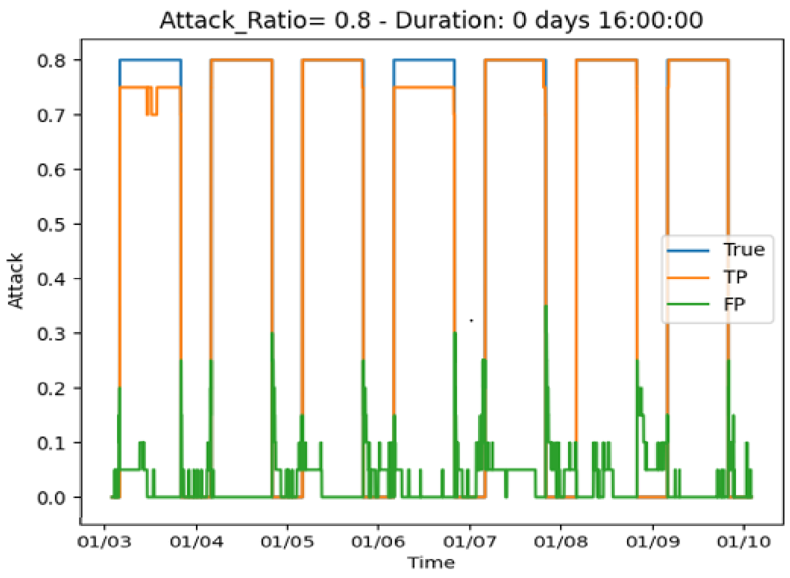

Figure 9: DNN training Dataset Attack Prediction vs Time for 0.8 attack ratio on 20 models

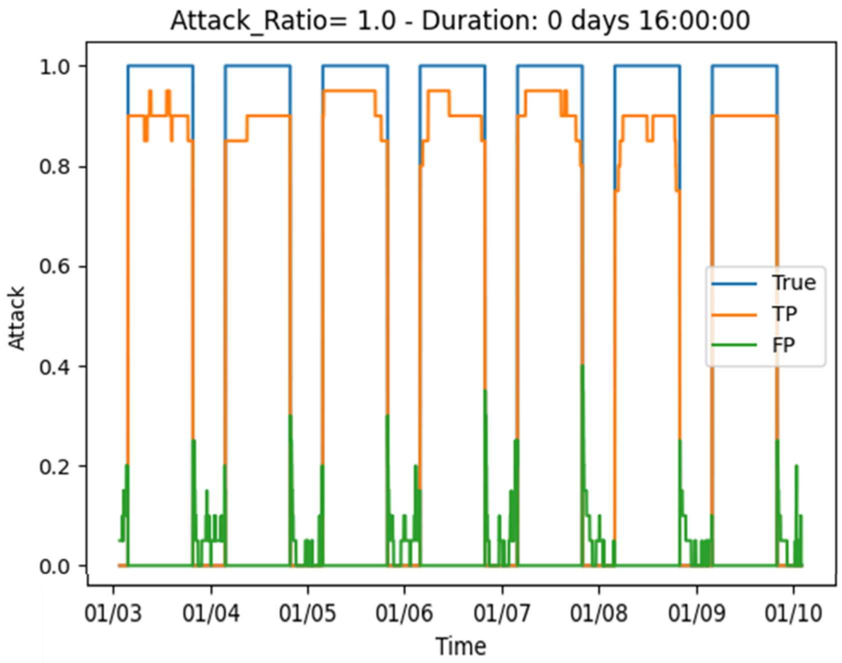

Figure 10: DNN training Dataset Attack Prediction vs Time for 1 attack ratio on one model

Figures 8 and 9 show the mean total nodes vs. time for true attack (T), attack predictions true positive (TP), and false positives (FP) when applying an average of 20 models on 20 nodes for both training datasets. And figure 10 shows the metrics when using one model on 20 nodes. The attack duration was set to 16 hours in these numbers. The attacked nodes are detected extremely effectively in both training datasets, but there are a few FP in figure 8, that is attack ratio of 1.

6.3. Discussion

Tables 2 and 3 illustrate the outcomes of the first approach for the four machine learning techniques used on the attack ratio of 1. And table 4 and 5 are on the attack ratio of 0.8. The results are separated into two. That is when 20 models are used to train 20 IoT nodes, and when one model is used to train 20 nodes. From the general performance, Deep Neural Network (DNN) on ratio 1 produced the accuracy of 95.9% and 88.6% for train and test datasets, respectively with a fewer False-Positive value within the attacked windows. These results also exceeded the results of prior work on the same dataset that used a Feedforward Neural Network to identify DDoS, with accuracy and recall of 94% and 88%, respectively (Hekmatic *et al.* 2021). Random Forest (RF) on the other hand, produced the lowest percentage of 94.2% and 87.0%.

Overall, the algorithms show high performance (accuracy, recall, and precision) when the approach of 20 models on 20 nodes than when the approach for one model on 20 nodes across the four machine learning algorithms used, as demonstrated in the tables and figures above. The training dataset's accuracy performance for 20 models on the 20 nodes approach, for example, ranges from 94.2% to 94.9%, whereas accuracy performance for one model on 20 nodes runs from 83.4% to 87.4%. In plot figure 10, the explanation for the poor performance of the one node 20 nodes architecture is as a result of low True-Positive (TP) value during the attack windows when compared to 20 models on 20 nodes in figures 8 and 9.

These results are based on data from 20 IoT nodes out of a total of 4060 IoT nodes, and only for two attack ratios. As a result, future studies are still required that could employ more IoT nodes and different attack ratios.

7. Conclusion and Future Work

The research addresses the improvement in the performance of machine learning models at detecting DDoS attacks on Internet of Things Devices using the Urban IoT dataset. To show the practical implementation of this project, four machine learning models were analysed: Feedforward Neural Network (FNN), Deep Neural Network (DNN), Autoencoder, and Random Forest (RF). The research was carried out to compare the performance of these four algorithms. This was conducted through the use of several design concepts. Firstly, a single model was used to train and test 20 nodes, followed by 20 models used to train and test 20 nodes. Secondly, by using different attack ratios (0.8 and 1). The project implementation processes entailed data cleaning, attack emulation, train and test data generation, models training, matrices results generation, and result comparisons.

It has been shown from the result that DNN can classify DDoS data with better accuracy than the other three algorithms. Therefore, this has provided an answer to the research question; that is Deep Neural Network (DNN) machine learning performs well in detecting DDoS attacks on the Internet of Things Devices using the Urban IoT dataset. However, when compared to the other algorithms, the DNN took a long time to train and test. As a result, there is an opportunity for improvement, and fine-tuning the model that may enable it to train faster.

In the future study, I intend to use a deep learning model such as Convolutional Neural Networks (CNN) with larger IoT nodes to do further research on the same dataset. However, I will build up the implementation in a more resource-intensive environment.

References

Abdullah, E., Kazim, Y. and Ali, B (2020). ‘Detection of DDoS Attacks with Feed Forward based Deep Neural Network Model’ *Expert Systems with Applications,* 169, pp.169 doi: 114520.10.1016/j.eswa.2020.114520.

Ali, J., Roh, B., Lee, B., and Ali, M. (2020) 'A Machine Learning Framework for Prevention of Software-Defined Networking controller from DDoS Attacks and dimensionality reduction of big data,' *International Conference on Information and Communication Technology Convergence (ICTC),* Jeju, Korea (South), October 2020, pp. 515-519, doi: 10.1109/ICTC49870.2020.9289504.

Ashi, Z., Aburashed, L., Al-Fawa’reh, M., and Qasaimeh, M. (2020) 'fast and Reliable DDoS Detection using Dimensionality Reduction and Machine Learning' *2020 15th International Conference for Internet Technology and Secured Transactions (ICITST*), London, United Kingdom, December 2020, pp. 1-10, doi: 10.23919/ICITST51030.2020.9351347.

Cil, A. E., Yildiz, K., Buldu, A. (2021) 'Detection of DDoS attacks with feed forward based deep neural network model' *Expert Systems with Applications*, 169, pp. 0957-4174, doi: 10.1016/j.eswa.2020.114520.

Chen, Y., Sheu, J., Kuo, Y. and Cuong, N. (2020) 'Design and Implementation of IoT DDoS Attacks Detection System based on Machine Learning' *2020 European Conference on Networks and Communications (EuCNC),* Dubrovnik Croatia, June 2020, pp.122-127, doi: 10.1109/EuCNC48522.2020.9200909.

Dertat, A. (2017) 'Applied Deep Learning - Part 3: Autoencoders' Available at: https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798 [Accessed 3 December 2021].

Cloudflare (2021) ‘What is a DDoS attack?’ Available at: https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/ [Accessed 3 December 2021].

Dennis, R., and Li, X. (2018) 'Machine-learning and statistical methods for DDoS attack detection and defense system in software defined networks' *Ryerson University. Thesis*. 1, doi /10.32920/ryerson.14657556.

Doshi, R., Apthorpe, N. and Feamster, N. (2018) ‘Machine Learning DDoS Detection for Consumer Internet of Things Devices’, *2018 IEEE Security and Privacy Workshops (SPW),* San Francisco, CA, USA, May 2018, pp. 29-35, doi: 10.1109/SPW.2018.00013.

Douligeris, C. and Mitrokotsa, A. (2004) ‘DDoS attacks and defense mechanisms: classification and state-of-the-art’, *Proceedings of the 3rd IEEE International Symposium on Signal Processing and Information Technology (IEEE Cat. No.03EX795*), 44(5), pp. 190-193, doi: 10.1109/ISSPIT.2003.1341092.

github.com, ‘Urban_IoT_DDoS_Data’. Available at: https://github.com/ANRGUSC/Urban_IoT_DDoS_Data/tree/main/dataset [Accessed December 7, 2021].

Goyal, C., Majumder, P. and Argrawal, R. (2017) 'Creating & Visualizing Neural Network in R’ Available at: https://www.analyticsvidhya.com/blog/2017/09/creating-visualizing-neural-network-in-r/ [Accessed 29 January 2022].

Hadian, H., Gonzalez, J. H., Stakhanova, N. and Ghorbani, A. A. (2017) 'Detecting HTTP-based Application Layer DoS attacks on Web Servers in the presence of sampling.' *Computer Networks*, 121, pp.121. doi: 10.1016/j.comnet.2017.03.018.

Hekmati, A., Grippo, E., and Krishnamachari, B., (2021) 'Dataset: Large-scale Urban IoT Activity Data for DDoS Attack Emulation'. *ACM Conference on Embedded Networked Sensor Systems*, Coimbra, Portugal, November 2021, pp. 560–564 doi: org/10.1145/3485730.3493695.

Jordan, J. (2018) 'Introduction to autoencoders.' Available at: https://www.jeremyjordan.me/autoencoders/ [Accessed 3 December 2021].

Jurafsky, D and Martin, J., (2021) 'Neural Networks and Neural Language Models' Available at: https://web.stanford.edu/~jurafsky/slp3/7.pdf [Accessed 9 January 2022].

Krishnan, S., Franklin, J., Goldberg, K., Wang, J. and Wu, E. (2016) 'ActiveClean: An Interactive Data Cleaning Framework For Modern Machine Learning' *In Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16)*, New York, June 2016, pp. 2117–2120, doi: 10.1145/2882903.2899409.

Kumar, A. and Lim, T. (2019) ‘A Secure Contained Testbed for Analyzing IoT Botnets’. In: Gao H., Yin Y., Yang X., Miao H. (2018) (eds) ‘Testbeds and Research Infrastructures for the Development of Networks and Communities. TridentCom’ *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, 270, pp. 124-137, doi: 10.1007/978-3-030-12971-2_8.

Meidan, Y., Bohadana, M., Mathov, Y., Mirsky, Y., Shabtai, A., Breitenbacher, D. and Elovici, Y., (2018). 'N-BaIoT—Network-Based Detection of IoT Botnet Attacks Using Deep Autoencoders'. *IEEE Pervasive Computing*, 17(3), pp.12-22. doi: 10.1109/MPRV.2018.03367731.

Misbahuddin, M. and Zaidi, S. M., (2021) 'Clustering based semi-supervised machine learning for DDoS attack classification' *Journal of King Saud University – Computer and Information Sciences* 33, pp. 436–446, doi: 10.1016/j.jksuci.2019.02.003.

Mishra, A., Gupta, B., Perakovic, D., Penalvo, F. and Hsu, C. (2021) 'Classification Based Machine Learning for Detection of DDoS attack in Cloud Computing' *2021 IEEE International Conference on Consumer Electronics (ICCE)*, Las Vegas, NV, USA, January 2021, pp. 1-4, doi: 10.1109/ICCE50685.2021.9427665.

Nielsen, M. (2019) 'Why are deep neural networks hard to train?' Available at: http://neuralnetworksanddeeplearning.com/chap5.html [Accessed 3 December 2021].

Ortet Lopes, I., Zou, D., Ruambo, F., Akbar, S. and Yuan, B. (2021) 'Towards Effective Detection of Recent DDoS Attacks: A Deep Learning Approach'. *Security and Communication Networks*, November 2021, pp.1-14. doi: 10.1155/2021/5710028.

Ozgur, C. F. and Fatih, M. A. (2019) ‘Distributed Denial of Service Attack Detection Using Autoencoder and Deep Neural Networks’ *Journal of Intelligent & Fuzzy Systems*. 37. pp. 1-11, doi: 10.3233/JIFS-190159.

Pande S., Khamparia A., Gupta D. and Thanh D.N.H. (2021) ‘DDOS Detection Using Machine Learning Technique. In: Khanna A., Singh A.K., Swaroop A. (eds) ‘Recent Studies on Computational Intelligence. Studies in Computational Intelligence’, *Springer, Singapore. Studies in Computational Intelligence,* 921, pp. 59-68, doi: org/10.1007/978-981-15-8469-5_5

Pérez-Díaz1, J. A, Valdovinos, I. A., Choo, K. R., and Zhu, D. (2020) 'A Flexible SDN-Based Architecture for Identifying and Mitigating Low-Rate DDoS Attacks Using Machine Learning' *in IEEE Access*, 8, pp. 155859-155872, doi: 10.1109/ACCESS.2020.3019330.

Rahman, O., Ali, M., Quraishi, G. and Lung, C. (2019) 'DDoS Attacks Detection and Mitigation in SDN using Machine Learning' *2019 IEEE World Congress on Services*, Milan, Italy, July 2019, 2642-939X, pp. 184-189, doi: 10.1109/SERVICES.2019.00051.

Ranger, S (2020*)* 'What is the IoT*?* Everything you need to know about the Internet of Things right now. ZDNet'. Available at: https://www.zdnet.com/article/what-is-the-internet-of-things-everything-you-need-to-know-about-the-iot-right-now/ [Accessed 3 December 2021].

Reddy, K. and Thilagam, P. (2020) 'Naïve Bayes Classifier to Mitigate the DDoS Attacks Severity in Ad-Hoc Networks' *International Journal of Communication Networks and Information Security (IJCNIS)* 12(2), pp. 221-226, doi: 10.54039/ijcnis.v12i2.4574 ,

Rey, V., S'anchez, P.M., Celdr'an, A.H., Bovet, G., & Jaggi, M. (2021). 'Federated Learning for Malware Detection in IoT Devices' *Computer Networks*, 204, pp. 108693, doi: abs/2104.09994.

Rios, V. M., Pedro, R.M., Magoni, D. and Freire, M. M. (2021) 'Detection of reduction-of-quality DDoS attacks using Fuzzy Logic and machine learning algorithms', *Computer Networks*, 186, pp.107792, doi: 10.1016/j.comnet.2020.107792

Shaaban, A. R., Abd-Elwanis, E. and Hussein, M. (2019) 'DDoS attack detection and classification via Convolutional Neural Network (CNN)' *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*, Cairo, Egypt, December 2019, pp. 233-238, doi: 10.1109/ICICIS46948.2019.9014826.

Sharafaldin, I., Lashkari, A. H., Hakak, S. and Ghorbani, A. A. (2019) 'Developing Realistic Distributed Denial of Service (DDoS) Attack Dataset and Taxonomy'. *IEEE 53rd International Carnahan Conference on Security Technology*, Chennai, India. October 2019, pp. 1-8, doi: 10.1109/CCST.2019.8888419.

Singh, A. K. (2021) 'Machine Learning in OpenFlow Network: Comparative Analysis of DDoS Detection Techniques' *The International Arab Journal of Information Technology*, (18)2. pp. 152-157, doi: 10.1109/ISRITI51436.2020.9315510.

Soe, Y. N., Santosa, P. I. and Hartanto, R. (2019) 'DDoS Attack Detection Based on Simple ANN with SMOTE for IoT Environment,' *2019 Fourth International Conference on Informatics and Computing (ICIC)*, Semarang, Indonesia, October 2019, pp. 1-5, doi: 10.1109/ICIC47613.2019.8985853.

Tawalbeh, L., Muheidat, F., Tawalbeh, M. and Quwaider, M. (2020) 'IoT Privacy and Security: Challenges and Solutions'. *Appl. Sci*. 10(12), pp. 4102; doi: org/10.3390/app10124102

Tonkal, O., Polat, H., Başaran, E., Cömert. Z., and Kocaoğlu, R., (2021). 'Machine learning approach equipped with neighbourhood component analysis for ddos attack detection in software-defined networking' *Electronics (Switzerland*), 10(11), pp. 1227. doi: 10.3390/electronics10111227

Tsimbalist, S., (2019) 'Detecting, classifying and explaining IoT botnet attacks using Deep Learning Methods based on network data' *Bachelor degree thesis*, Tallinn University Of Technology.

Vashi, S., Verma, S., Ram, J., Modi, J. and Prakash, C. (2017) 'Internet of Things (IoT): A vision, architectural elements, and security issues' *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC).* Palladam, India, February 2017, pp. 492-496. doi: 10.1109/I-SMAC.2017.8058399.

Wei, J., Chu, X., Sun, X-Y., Xu K., Deng, H., Chen, J., Wei, Z and Lei, M., (2019) 'Machine learning in materials science'. *Environmental Science & Technology* 55(19), pp. 338– 358. doi: 10.1002/inf2.12028.