

# Detection of DDoS Attacks on Urban IoT Devices Using Neural Networks

Simon Onyebuchi Obetta and Arghir-Nicolae Moldovan<sup>a</sup>  
*School of Computing, National College of Ireland, Dublin, Ireland*

**Keywords:** IoT, DDoS Attacks, Intrusion Detection, Machine Learning, Neural Networks.

**Abstract:** As the Internet of Things (IoT) has grown in recent years, attackers are increasingly targeting IoT devices to perform malicious attacks such as DDoS. Often, this is due to 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 for IoT devices, effective DDoS detection methods are still required. This paper compares the performance of four machine learning algorithms for DDoS detection on a recent Urban IoT dataset: Feedforward Neural Network (FNN), Deep Neural Network (DNN), Autoencoder (AEN) and Random Forest (RF). The results show that DNN achieved the highest accuracy of 95.9% on train data and 88.6% on test data.

## 1 INTRODUCTION


The Internet of Things (IoT) is a technology that connects smart electronic devices to the Internet for data collection and transfer without human intervention. Presently many IoT systems are interconnected with several sensors and maintain communication and exchange massive volumes of data. For instance, in the context of smart-home applications, large-scale IoT systems with numerous sensor nodes are being used and proposed. 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.

IoT devices are also prone to security vulnerabilities and increasingly targeted by attackers, thus, IoT is becoming a major research area 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 botnet. It takes advantage of vulnerabilities such as unsecured ports, use of default passwords, unpatched software, to penetrate the targets' system (Douligeris and Mitrokotsa, 2004). The DDoS attackers aim to deny access to legitimate users by taking down or slowing the network or application.

Many recent anomaly-based detection studies have compared the performance of different machine learning algorithms (MLA) and showed they have good potential in detecting malware and DDoS in computer networks traffic. However, more research is needed on detecting real-life or slow DDoS attacks, especially in IoT devices.

This research aims to investigate DDoS attacks detection in IoT devices using the recent Urban IoT dataset (Hekmati et al., 2021). This will be performed by doing a comprehensive performance comparison of three Neural Network-based algorithms (i.e., Feedforward Neural Network, Deep Neural Network, Autoencoder), and the traditional Random Forest algorithm.

<sup>a</sup> <https://orcid.org/0000-0003-4151-1432>

The rest of the paper is structured as follows. Section 2 presents related works. Section 3 presents the methodology. Section 4 presents the results, while section 5 concludes the paper.

## 2 RELATED WORKS

This section provides an overview of several previous research papers on DDoS detection using machine learning methods.

Ashi et al. (2020) investigated DDoS attacks detection 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). A dataset comprising the simulation's network traffic flow was created, and Random Forest (RF) 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 the 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 RF. The results of their experiment revealed that J48 is the best classifier with accuracy, F-1, and recall rate of 100%.

Reddy and Thilagam (2020) applied Naive Bayes (NB) classifier to detect DDoS attack traffic by considering 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 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 were used to group unlabelled data, and a voting procedure determines the final classification of traffic flows. To detect DDoS attacks, the supervised learning algorithms K-NN, SVM and RF are applied to labelled data, with accuracy achieved of 95%, 92%, and 96.66%, respectively.

Rios et al. (2021) tested and compared the Multi-Layer Perceptron (MLP), K-NN, 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), 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. MLP outperformed the other four machine learning algorithms when it comes to detecting RoQ attacks.

Doshi, et al. (2018) investigated multiple machine learning algorithms K-NN, Linear SVM, Decision Tree (DT), RF and Neural Network (NN) for DDoS detection for consumer IoT. 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 99.1%.

Mishra et al (2021) investigated DDoS attacks detection in cloud computing. The machine learning algorithms adopted for classification were K-NN, NB and 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 accuracy of 99.58%.

Hekmati 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 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 attack emulation to generate an artificial DDoS attack for the attack ratio of 1 on the 20 selected IoT nodes. The simple FNN achieved a mean accuracy of 94% and 88% on the train and test data, respectively.

Shaaban et al. (2019) proposed the use of a Convolutional Neural Network (CNN) for DDoS detection. For their research, the authors used two datasets: a generated dataset and the NSL-KDD dataset. The results showed that CNN achieved 99% accuracy, and outperformed other algorithms like DT, SVM, K-NN and NN.

### 3 METHODOLOGY

Fig. 1 shows the steps of the research methodology.

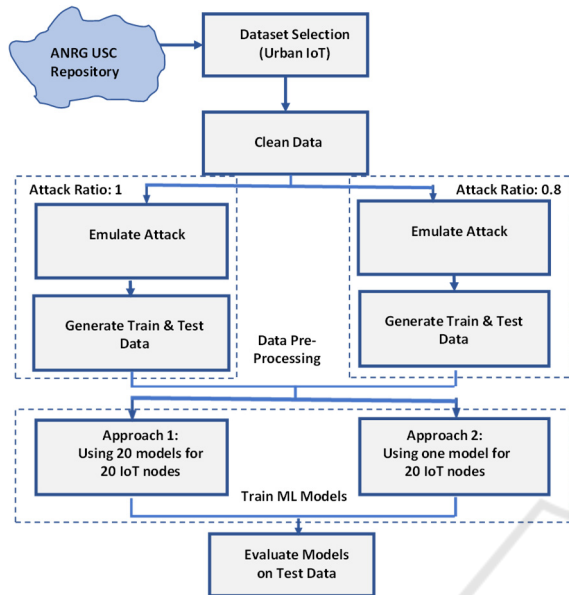


Figure 1: Workflow of the research methodology.

#### 3.1 Data Selection

The dataset selected for this research is the Urban IoT that was captured from the activity status of genuine event-driven IoT nodes installed in a city (Hekmati et al., 2021). The data captured is the activity of 4060 urban IoT devices (nodes) for one month, making it a very realistic IoT dataset. As it is a recent dataset there are very few papers that have used it and there are open research questions.

The original dataset contains the node ID and location (latitude and longitude). The dataset also contains each node's binary activity status at 30 seconds interval over 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 active at midday, but by midnight, only approximately 20% of the nodes are active.

#### 3.2 Data Cleaning

Data cleaning is part of the pre-processing activity carried out on the original dataset. Data cleaning has the potential to increase the efficiency and effectiveness of the training process. To emphasize

the role of data cleaning in 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. In this paper the data cleaning focused on generating benign data for 20 randomly selected IoT nodes. The benign dataset contains the occupancy status of each node between the start and end dates available in the original dataset, with a time step of 30 seconds. An additional attribute, attacked, with the value set to 0, was also added.

#### 3.3 Attack Emulation

An artificial attack emulation was provided by the dataset authors, to make it suitable for training machine learning models for DDoS detection (ANRG USC, 2021). Emulation overcomes the challenges associated with performing real DDoS attacks on active IoT nodes, which may not be allowed by the IoT nodes owner, and it may not be feasible to perform large scale DDoS attacks on many IoT nodes. The attack emulation step created new attributes such as begin\_date, end\_date, num\_nodes, attack\_ratio, attack\_duration.

#### 3.4 Train Machine Learning Models

The ML algorithms employed in the experiment include one traditional algorithm, Random Forest (RF), and three neural network-based algorithms.

**Feed-Forward Neural Network:** This is the simplest type of neural network that only has one hidden layer (Jurafsky and Martin, 2021). The input layer sends a multi-dimensional request to the 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. The aim of using this model is to reproduce the DDoS detection done by Hekmati et al. (2021) using the same dataset. The FNN implemented 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. The output layer consists of a single neuron with the Sigmoid activation function. To discover the attacked

time slots in the dataset, the neural network model is trained for 500 epochs for each node.

**Deep Neural Network:** This model is made up of feedforward neural networks that do not have any feedback connections. DNN consists of the input and output layers, but the main distinction from FNN is that it has more than one hidden layer. 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. 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 non-deep neural networks. This can lead to an increase in the overall performance. 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 is used in the output layer.

**Autoencoder Neural Network:** This is a type of neural network that shrinks multidimensional input data within a hidden region before reconstructing the data from the hidden region (Ozgun and Fatih, 2019). 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 is located between the hidden layer and the output layer. 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 like DNN, it is a multi-hidden layer neural network that can uncover complex correlations and mappings of data and increase the performance. For example, Ozgun 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. In the implemented Autoencoder, 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:** This is a traditional machine learning algorithms that is based on the construction of numerous small trees in a decision tree (Dangwal & Moldovan, 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. According to Mishra et al., (2021), RF is recommended for supervised learning 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. The implemented random forest classifier uses 'n\_samples=1000', 'n\_features=20', 'random state=3', n\_split=10, n\_repeats=3, and n\_jobs=-1.

### 3.5 Evaluation

The performance of the four models to detect DDoS attacks is compared based on different metrics, namely accuracy, recall, and precision. The data is split into 80% for training and 20% for test. The performance of the models is compared for 2 different approaches, (1) implementing 20 models one for each of the 20 IoT nodes, and (2) implement one model on all the combined data from 20 IoT nodes. The 2<sup>nd</sup> approach was not used in previous papers on this dataset. Moreover, two values were used for the attack ratio of 1 and 0.8.

## 4 RESULTS

The results are presented in Tables 1 to 4. Tables 1 and 2 illustrate the results for the four ML algorithms used for attack ratio of 1, which show that overall DNN achieved the highest accuracy. For the 20 models on 20 IoT nodes, DNN achieved an accuracy of 95.9% and 88.6% for train and test datasets, respectively. These results are slightly better than the results of prior work on the same dataset that used a Feedforward Neural Network to identify DDoS, with accuracy 94% (Hekmati et al. 2021). For the 1 model built on the combined data of 20 IoT nodes, DNN again achieved the highest accuracy of 87.4% and 85.4% on the train and test datasets, respectively.



Table 1: Results for 20 models on 20 IoT nodes, attack ratio 1.

	Model	Mean Accuracy	Mean Recall	Mean Precision
Train Data	FNN	0.943	0.936	0.794
	DNN	<b>0.959</b>	<b>0.942</b>	<b>0.842</b>
	Autoencoder	0.957	0.936	0.839
	RF	0.958	<b>0.942</b>	<b>0.842</b>
Test Data	FNN	0.870	<b>0.835</b>	0.680
	DNN	<b>0.886</b>	0.824	<b>0.694</b>
	Autoencoder	0.883	0.791	0.687
	RF	<b>0.886</b>	0.819	<b>0.694</b>

Table 2: Results for 1 model on 20 IoT nodes, attack ratio 1.

	Model	Mean Accuracy	Mean Recall	Mean Precision
Train dataset	FNN	0.796	0.236	0.662
	DNN	<b>0.874</b>	0.894	0.645
	Autoencoder	0.846	<b>0.955</b>	<b>0.666</b>
	RF	0.846	<b>0.955</b>	<b>0.666</b>
Test dataset	FNN	0.801	0.242	<b>0.699</b>
	DNN	<b>0.854</b>	0.875	0.640
	Autoencoder	0.839	<b>0.916</b>	0.643
	RF	0.839	<b>0.916</b>	0.643

Table 3: Results for 20 models on 20 IoT nodes, attack ratio 0.8.

	Model	Mean Accuracy	Mean Recall	Mean Precision
Train Data	FNN	0.933	0.942	0.726
	DNN	0.956	<b>0.945</b>	0.799
	Autoencoder	<b>0.958</b>	0.938	<b>0.803</b>
	RF	<b>0.958</b>	0.938	<b>0.803</b>
Test Data	FNN	0.857	<b>0.817</b>	0.583
	DNN	0.883	0.801	0.610
	Autoencoder	<b>0.885</b>	0.762	<b>0.616</b>
	RF	<b>0.885</b>	0.762	<b>0.616</b>

Table 4: Results for 1 model on 20 IoT nodes, attack ratio 0.8.

	Model	Mean Accuracy	Mean Recall	Mean Precision
Train dataset	FNN	0.832	0.593	<b>0.622</b>
	DNN	0.854	<b>0.920</b>	0.569
	Autoencoder	<b>0.864</b>	0.905	0.591
	RF	0.854	<b>0.920</b>	0.569
Test dataset	FNN	0.827	0.624	<b>0.605</b>
	DNN	0.832	<b>0.917</b>	0.554
	Autoencoder	<b>0.839</b>	0.894	0.574
	RF	0.832	<b>0.917</b>	0.554

Overall, the algorithms show higher performance for the 1<sup>st</sup> approach of training 20 models for each of the 20 IoT nodes than for the 2<sup>nd</sup> approach of training

one model on the combined data of 20 IoT nodes. For example, the test accuracy for 20 models on 20 nodes approach, ranges from 87.0% to 88.6%, whereas the accuracy for one model on 20 nodes varies from 80.1% to 85.4%.

Tables 3 and 4 illustrate the results for the four ML algorithms used for attack ratio of 0.8, which show that overall Autoencoder achieved the highest accuracy. For the 20 models on 20 IoT nodes, Autoencoder achieved an accuracy of 95.8% and 88.5% for train and test datasets. For the 1 model built on the combined data of 20 IoT nodes, Autoencoder again achieved the highest accuracy of 86.4% and 83.9% on the train and test datasets, respectively.

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.

Figures 2 to 3 show the true positive (TP) and false positives (FP) for 20 models on 20 nodes for attack ratio 1, on the training and test datasets, respectively. The duration was set to 16 hours in these experiments. The results indicate that there are fewer FP for training data than for test data. Figures 4 and 5 show the DNN performance when using one model on 20 nodes for attack ratio 1. The explanation for poorer performance of the one model on 20 nodes is the lower true positive rates and during the attack windows.

## 5 CONCLUSIONS

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 DDoS dataset. Four machine learning algorithms were investigated: Feedforward Neural Network (FNN), Deep Neural Network (DNN), Autoencoder, and Random Forest (RF). Two approaches were used for the comparison, building 20 models for 20 IoT nodes each, and building 1 model on the combined data of the 20 nodes.

The results showed that DNN can classify DDoS data with slightly better accuracy than the other three algorithms. 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. Future work may investigate other algorithms such as Convolutional Neural Networks (CNN) with larger number of IoT nodes to do further research on the same dataset.

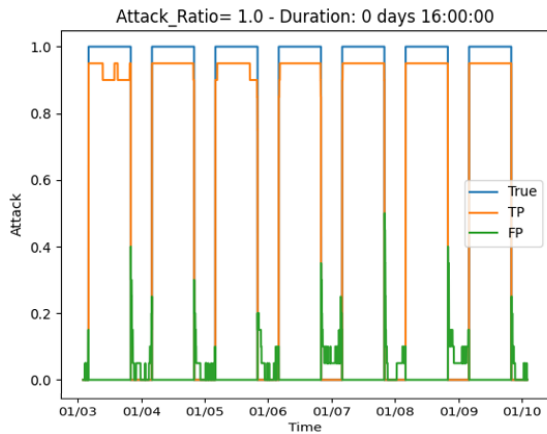


Figure 2: DNN training dataset attack prediction vs. time, for 20 models on 20 IoT nodes, attack ratio 1.

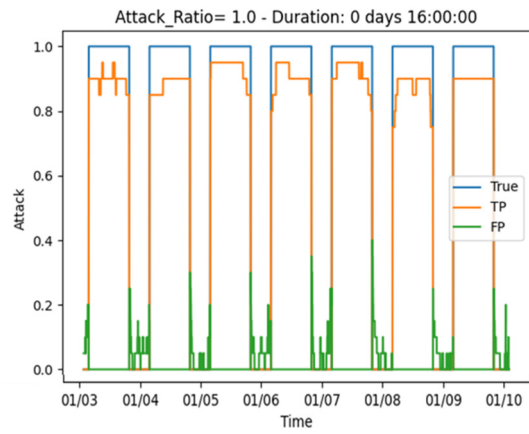


Figure 4: DNN training dataset attack prediction vs. time, for 1 model on 20 IoT nodes, attack ratio 1.

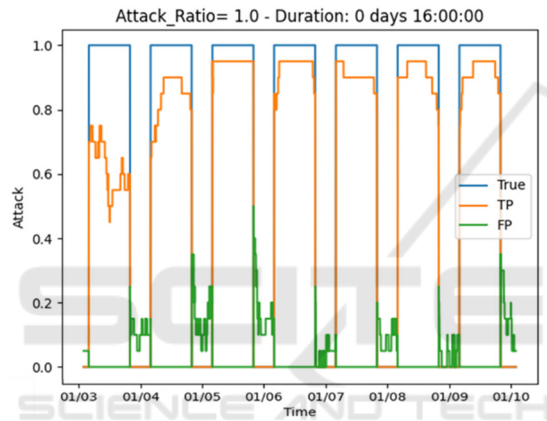


Figure 3: DNN test dataset attack prediction vs. time, for 20 models on 20 IoT nodes, attack ratio 1.

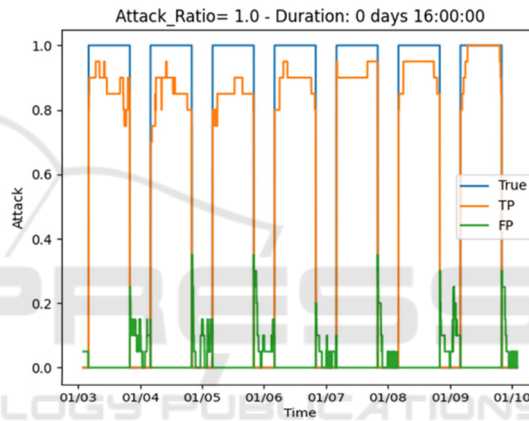


Figure 5: DNN test dataset attack prediction vs. time, for 1 model on 20 IoT nodes, attack ratio 1.

## REFERENCES

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.

Cloudflare (2021) 'What is a DDoS attack?' Available at: <https://www.cloudflare.com/learning/ddos/what-is-a-ddos-attack/>.

Dangwal, S., & Moldovan, A.-N. (2021). Feature Selection for Machine Learning-based Phishing Websites Detection. *2021 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, 1–6. doi: 10.1109/CyberSA52016.2021.9478242

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.

ANRG USC, 'Urban\_IoT\_DDoS\_Data'. Available at: [https://github.com/ANRGUSC/Urban\\_IoT\\_DDoS\\_Data/](https://github.com/ANRGUSC/Urban_IoT_DDoS_Data/).

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.

Jurafsky, D and Martin, J., (2021) 'Neural Networks and Neural Language Models' Available at: <https://web.stanford.edu/~jurafsky/slp3/7.pdf>.

- 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.
- 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.
- 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
- 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.
- 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 ,
- 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.
- 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.