International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Greedy Feature Selection and Decision Tree Algorithms for DDOS Attack Detection in Software Defined Network

Nwobodo-Nzeribe Nnenna Harmony [a], Njoku Camillus Ekene [b] , Shamsudeen Mohammed S.[c]

[a] nnennanwobo8@gmail.com [b] camillusnjoku@gmail.com, [c]shamsudeensada@yahoo.com,
[a, b, c] **Department of Computer Engineering, Enugu State University of Science and Technology Enugu, Nigeria**

ABSTRACT

The security of Software-Defined Networking (SDN) systems is seriously threatened by the growing sophistication and frequency of Distributed Denial of Service (DDoS) attacks. This study presents a Decision Tree-based approach for detecting DDoS attacks in SDN systems. The datasets utilized for training the algorithms are the CICDDoS2019 and KDD CUP-1999 datasets. The Greedy Feature Selection (GFS) algorithm is used for feature selection after the system uses data preprocessing procedures. This method optimises the model's performance by reducing the dimensionality of the dataset while maintaining important information. Network traffic is categorised as malicious or non-malicious using a Decision Tree Algorithm (DTA). When the system is examined, it achieves high precision, recall, and F1-score values (above 97%) with an average accuracy of 98.3%. With an average execution time of 2.8 seconds per fold, the model's computational efficiency is also shown, making it appropriate for real-time detection in SDN systems. With the possibility for future enhancements through ensemble approaches and real-time integration with SDN controllers, the results demonstrate that the suggested system offers a reliable, effective, and scalable solution for DDoS detection.

Keywords: DDoS Detection; Software-Defined Networking (SDN); Decision Tree Algorithm; Feature Selection; Greedy Feature Selection (GFS); Machine Learning

1. INTRODUCTION

Web-based software and services have seen a dramatic increase in popularity over the last 20 years. The Internet is currently used by 57% of the world's population (ClickZ, 2019). Consequently, concerns over internet security have significantly escalated. There have often been a number of security risks on the Internet. Trojan horses, malware, port scans, and denial-of-service attacks are a few examples of common internet anomalies (Singh and Behal, 2020). Large and complex networks are hard for traditional network topologies to solve effectively. A different approach called Software-Defined Networking (SDN) uses software, not hardware like switches and routers, to control network traffic. In SDN, the control plane is taken over by a centralised controller that functions as the main decision-maker for the network. Usually, SDN switches manage the data plane and carry out controller instructions. The flexibility and manageability of network administration are enhanced by this modification in the control architecture (Hossain et al., 2018; Sheikh et al., 2021).

SDN offers solutions for a range of network issues. These include low-cost network devices with easier data operations, enhanced Quality of Service (QoS), enhanced link failure detection, dynamic remote setup, vendor-independent device selection, and cost savings through centralised control (Ahuja et al., 2021). Notwithstanding its benefits, SDN has security flaws at multiple architectural levels, including particular dangers.

One major issue is that attackers can take over the central controller and use distributed denial of service (DDoS) attacks to disrupt the network. These attacks are hard to detect and stop, and they often use "botnets" of compromised computers. They are becoming more frequent and intense, which presents a significant obstacle for administrators and service providers in terms of timely detection and mitigation (Wang et al., 2023).

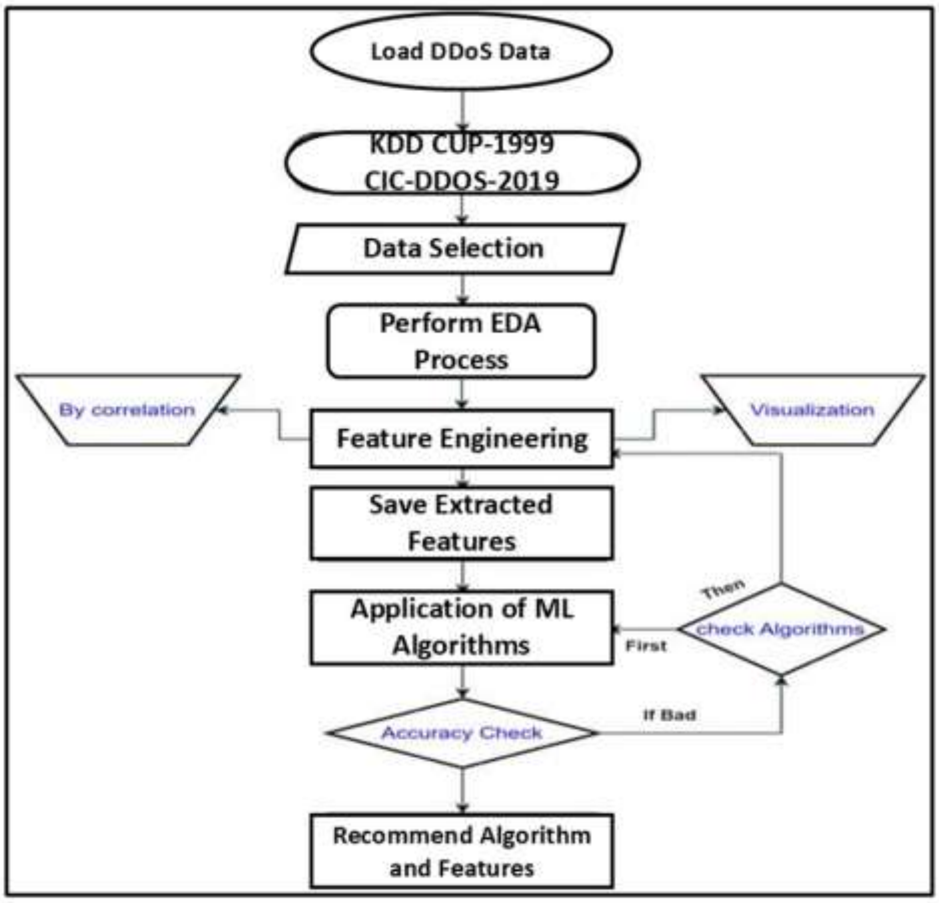
Additionally, attackers are using more advanced techniques, which makes it harder to identify DDoS attacks. Network failure results from the controller's inability to react to requests if it is the target of the attack. Targeting the data plane will result in massive volumes of pointless traffic since the DDoS attack will eat up too much processing power and network bandwidth (Raza et al., 2024). Such traffic interferes with ordinary traffic's ability to be forwarded normally and forces the SDN controller to send extra flow tables to switches for routing, which takes up storage space on SDN switches and adds to the load on the data plane. As a result, research on efficiently identifying and preventing DDoS attacks has shifted to the context of SDN (Liu et al., 2023).

The two main categories of detection techniques are machine-learning-based and statistics-based (Alzahrani and Alenazi, 2021; Mona et al., 2022). In statistics-based detection techniques, flow modelling is used to examine particular network protocols or application layer data in order to identify DDoS attacks. These techniques identify anomalies in attack flow and create predicted behaviour or traffic models. Despite being straightforward and effective, these techniques need manual setup for every new attack type and cannot handle novel attack scenarios (Catak and Mustacoglu, 2019; Njoku et al., 2024). On the other hand, machine-learning-based DDoS detection is a more sophisticated method that automatically detects DDoS attacks by utilising machine learning algorithms to understand typical traffic patterns and identify unusual traffic patterns. This technique can find hidden attack patterns in complicated network settings and adjust to new attack conditions. But in order to train and test algorithms, it needs enormous datasets and processing power. In the field of networking security, combining the benefits of SDN technology with a machine-learning-based strategy can accomplish DDoS detection and effective mitigation through automation, effective response, and Deep Learning (DL) (Ali et al., 2023).

A decision-tree-based method for identifying DDoS attacks in SDN-based systems is presented in this paper. Through the use of a specific simulated experimental network architecture, DDoS attack traffic data is obtained. Techniques for feature selection and hyperparameter tuning are used to maximise the performance of the decision tree models (Oyucu et al., 2024). There are numerous approaches for feature selection, but we must select the most effective one to address the detection issue.

2. RESEARCH METHODOLOGY

Figure 1's flowchart illustrates how machine learning can be applied to detect DDoS attacks. Selecting two datasets is the first stage, after which they are examined using Exploratory Data Analysis (EDA) to look for patterns, linkages, and significant characteristics. The EDA step includes correlation analysis, feature engineering, and data visualisation, all of which contribute to the quality of the data utilised for model training and dataset comprehension.



2.1 Data Collection

To determine the authenticity performance, any AI model needs efficient datasets for testing and training. Since gathering accurate data is a significant issue on a global scale, institutions and scholars are contributing to this field. In addition to using the CICDDOS2019 (Sharafaldin et al., 2019) and KDD CUP-1999 datasets, we also have distinct training and testing files with millions of rows and 88 columns. They include seven attack types for testing (NetBIOS, LDAP, MSSQL, UDP, UDPLag, and SYN) and twelve distinct DDOS attack types in total (DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN, and TFTP). In essence, the two types selected for this experiment are SYN and NetBIOS. These two data formats have a combined capacity of five gigabytes (5GB).

2.2 Data Preprocessing

For machine learning to function well, data preprocessing is essential. It transforms unprocessed data into representations that can be used to make predictions and get insights. In the CICDDoS2019 dataset, filling in the gaps is essential. Error identification and rectification are essential, and model performance may be enhanced by removing columns with significant data breaches. We have found that a large number of characteristics (columns) in the CICDDoS2019 dataset have zero values. Unnamed: 0, Bwd PSH Flags, Fwd URG Flags, Bwd URG Flags, FIN Flag Count, PSH Flag Count, ECE Flag Count, Fwd Avg Bytes/Bulk, Fwd Avg Packets/Bulk, Fwd Avg Bulk Rate, Bwd Avg Bytes/Bulk, Bwd Avg Packets/Bulk, and Bwd Avg Bulk Rate are a few examples of columns that were removed because they typically contain zero values.

Our study included nearly three distinct methods to choose attributes. First, we applied the "SelectBest" method from the scikit-learn library. This method's goal is to automatically choose features based on predetermined criteria, like selecting ten characteristics from the dataset. However, the outcomes did not meet our expectations. We then examined univariate selection techniques, which are helpful in identifying traits that significantly enhance the model's capacity for prediction. Lastly, we used a correlation-based approach to select features that are interdependent, which significantly increased the accuracy of our model. This tactic assisted in elucidating the connections between the variables by measuring the linear correlations between them. It is important to keep in mind, though, that because this approach depends on linear interactions, it might overlook nonlinear correlations, scale inconsistencies, and links across variables, all of which require thorough examination.

2.3 Feature Selection using Greedy Feature Selection (GFS) Algorithm

GFS is a straightforward, implementable mathematical technique that solves complicated problems by reaching useful conclusions (Tsamardinos et al., 2019). Recursively creating object sets from the least likely component parts is how this algorithm works. Either all of the worst qualities are eliminated separately (backward selection) or only the greatest features are chosen (ahead selection). The backward selection method is used in the suggested model. The GFS algorithm is used to load the dataset for pre-processing and feature selection. To complete this procedure, it follows a number of phases, which are detailed in Algorithm 1;

Algorithm 1: GFS Algorithm (Praba and Sridan, 2022)

1. Set up the attack features for the dataset and its source.
2. Create the objects for attribute selection, search algorithms, and the evaluator.
3. Start a Greedy backward search on a certain dataset using a filter based on the search methods and evaluator.
4. For the current population set of features in the current search iteration, calculate the Leave One Out Cross Validation (LOOCV) error of DT classification. For the input set of features of the current iteration $xiter$, this is the fitness cost $f(xiter)$.
5. To carry out greedy procedures in a step-by-step manner, apply a filter. To make the search filters more effective, evaluate it.
6. Get the number of classes and their characteristics. Update the weights of the class indexes after mapping them for a predetermined number of instances.
7. For the maximum number of search iterations, repeat steps 2 through 7.
8. Save and update the minimum fitness cost as $f_{min}(x)$ = min $f(x)$.
9. The important elements from the dataset are included in the GFS final optimal solution.

As the evaluation measure of the feature set x, the fitness cost $f(x)$ is used as the parameter to choose the important features. The GFS is run until the optimal solution, $f_{min}(x)$, is found. The 48-column dataset is thus reduced to 5 columns with just pertinent and targeted features after all these procedures are carried out using GFS. Since feature selection has an impact on accuracy rate, it must be done effectively. In order to efficiently choose the finest features, GFS feature selection is implemented using seven phases. The GFS method usually makes it easier to solve problems than other algorithms. Therefore, putting it into practice will improve the detection accuracy demonstrated by results.

3. DECISION TREE MODEL FOR DDOS ATTACK DETECTION

A popular machine learning method for traversing from a root to its leaves is the Decision Tree Algorithm (DTA) (Mishra et al., 2023). It operates on the basic idea of dividing the dataset into a tree-structured model, with each node standing for a feature, each diverging branch communicating a feature's unique value, and the final leaf node representing a conclusive classification result (Gaur and Kumar, 2022). Decision trees have several benefits, such as being easy to comprehend and interpret, handling nonlinear features, being appropriate for huge datasets, and being able to be used to multi-class situations. To increase classification accuracy, decision trees can also be utilised as an ensemble model in conjunction with other machine learning methods (Sridaran et al., 2022). For classification models in a variety of application settings, decision trees are therefore usually the best option. Decision trees are also widely preferred in the realm of DDoS detection (Santos et al., 2020).

The roots, leaves, and branches make up the DTA components. Since the tree being studied here is directed, its roots lack edges. Other elements just have one edge. Furthermore, nodes without flow edges are visible in the interior nodes. Additional nodes are leaves that display the terminal or decisional nodes. The Interior node uses minimised feature sets to divide the space decision into several subspaces. Attribute spaces are referred to as conditional ranges when considering numerical characteristics. In order to achieve their respective classes, leaf nodes store target values. The internal nodes are arranged from the root node to the leaf nodes under the conditional values. As a result, DTA is used to feature classification using the steps listed below in Algorithm 2.

Algorithm 2: DT Algorithm for DDoS Classification (Praba and Sridan, 2022)

1. Create a root node (R) at the beginning of the tree that contains the entire dataset.
2. Use ASM (Attribute Selection Measure) to identify all of the dataset's greatest attributes.
3. Using information gain, use ASM (Attribute Selection Measure) to identify all of the dataset's greatest attributes. I. The criterion for determining how much information each feature attribute contains is called information gain. Expressed as, $I = E_R - (A_R * E_x)$ Where, E_R is the entropy of the dataset, E_x is the entropy of the feature x, and A_R is the weighted average of the dataset. This entropy metric aids in specifying the degree of randomness in the data as well as in identifying redundant or superfluous information in an attribute.
4. Determine which DT node has the best properties.
5. Create new DTs recursively with the dataset subsets created in step 3. Continue doing this until a certain point is reached at which no more node classification is possible. These are the leaf nodes, the last nodes.

As a result, DTA receives all of the features chosen by the GFS algorithm and uses the five stages mentioned above to classify the features. The proposed DTD model classifies the attacks as either malicious or non-malicious. The results, which are covered in the next part, validate the effectiveness of the suggested system.

4. SYSTEM IMPLEMENTATION

Three main steps are involved in the MATLAB implementation of the DDoS detection system: feature selection, data preprocessing, and classification using a Decision Tree Algorithm (DTA). To guarantee a high-quality dataset, data preparation is carried out using MATLAB's Data Cleaning Toolbox and Data preparation Tool. Irrelevant features, such as columns with high zero/null values, are eliminated, and missing data is dealt with. The MATLAB Visualisation Toolbox, which offers tools like heatmap, scatter, and histogram to find correlations and trends in the dataset, is used for exploratory data analysis, or EDA. The 88-column CICDDoS2019 dataset is ready for feature selection and classification in this step.

The Statistics and Machine Learning Toolbox, more especially the Greedy Feature Selection (GFS) algorithm with backward elimination, is used to implement feature selection. This entails finding and eliminating features that contribute little to the predicted accuracy of the model iteratively. The Decision Tree Classifier, which is constructed using MATLAB's fitctree function, is then fed the reduced dataset. By dividing the data hierarchically according to specific features, the Decision Tree Algorithm is trained to recognise DDoS attacks. The Optimisation Toolbox is used to do hyperparameter tweaking, which guarantees the optimal model parameter configuration to optimise detection accuracy. The system's capacity to correctly distinguish between dangerous and non-malicious traffic is confirmed by the findings, which are displayed using MATLAB's monitoring tools and assessed using metrics like accuracy, precision, recall, and F1-score.

5. SYSTEM RESULTS

The CICDDoS2019 and KDD CUP-1999 datasets were used to assess the developed system for identifying DDoS attacks in SDN systems, with an emphasis on preprocessing, feature selection, and classification. MATLAB's Data preparation Toolbox was used to resolve missing data and remove unnecessary columns with excessive zero/null values during the data preparation step. In order to improve the quality of the data and assist in the identification of important features, exploratory data analysis (EDA) techniques like scatter and heatmap were utilised to find correlations and trends. The dataset was prepared for the feature selection and classification stages via the preprocessing stage.

The Greedy Feature Selection (GFS) approach, which was developed with MATLAB's Statistics and Machine Learning Toolbox, was used to choose features. Using this method, the dataset was condensed from 88 columns to 5 key features: Packet Length Std, Bwd Packet Length Min, Fwd Packet Length Max, Total Fwd Packets, and Flow Duration. While maintaining characteristics necessary for precise categorisation, this dimensionality reduction improved computational efficiency. The Decision Tree Algorithm (DTA), constructed with MATLAB's fitctree tool, was then used to classify the condensed dataset. The model's performance was further improved by hyperparameter optimisation, which was accomplished using the Optimisation Toolbox.

As presented in Table 1, the measurements, which were obtained using MATLAB's confusionmat function, confirm how reliable the system is at differentiating between malicious and non-malicious communications.

Table 1: The validation results of the DDoS detection system

Metric	**Mean (%)**	**Standard Deviation (%)**
Accuracy	98.3	0.4
Precision	97.6	0.5
Recall	98.0	0.3
F1-Score	97.8	0.4
Execution Time (seconds per fold)	2.8	0.2

The DDoS detection system's validated findings, which are shown in Table 1, show how reliable, consistent, and successful the applied model is. The model successfully detects fraudulent traffic with little fluctuation across several dataset subsets, as evidenced by its mean accuracy of 98.3% and low standard deviation of 0.4%. For real-time DDoS detection, when mistakes might result in serious security flaws, this high accuracy is essential.

The model's accuracy in detecting true positives and reducing false negatives is demonstrated by its precision and recall values, which average 97.6% and 98.0%, respectively. Strong recall guarantees that the majority of attacks are correctly identified, while high precision indicates the system's capacity to identify malicious traffic precisely, lowering the possibility of false alarms. These indicators are balanced by the model's F1-score of 97.8%, which attests to its well-rounded performance.

With a low variation of 0.2 seconds and an average execution time of 2.8 seconds per fold, the system exhibits computational efficiency in addition to precision. This performance is the consequence of efficient feature selection utilising the Greedy Feature Selection (GFS) algorithm, which preserved important information while drastically reducing the dimensionality of the dataset. The model's stability and applicability for deployment in dynamic SDN systems are further validated by the low standard deviations across measurements, which provide accurate DDoS detection in a range of scenarios.

6. CONCLUSION

This study successfully developed a Decision Tree-based DDoS detection system tailored for Software-Defined Networking (SDN) environments. To improve the model's accuracy and computational efficiency, the implementation used sophisticated data preparation and feature selection techniques using the CICDDoS2019 and KDD CUP-1999 datasets, which cover a variety of DDoS attack types. By guaranteeing that the most pertinent features were kept for model training, the Greedy Feature Selection (GFS) algorithm played a crucial role in lowering the dimensionality of the data, greatly enhancing detection performance.

With an average accuracy of 98.3%, the Decision Tree Algorithm (DTA) showed remarkable classification abilities. The model's robustness and dependability were confirmed by additional metrics that continuously surpassed 97%, such as precision, recall, and F1-score. The study also emphasised how the system's low execution times and computational efficiency make it appropriate for real-time DDoS detection in SDN systems.

To sum up, the suggested system offers a useful and efficient way to identify DDoS attacks in SDN systems. Malicious traffic may be detected accurately and quickly thanks to a well-optimized Decision Tree model and sophisticated feature selection techniques. To further improve the system's scalability and responsiveness to changing network threats, further research could investigate ensemble learning techniques and real-time integration in SDN controllers.

REFERENCES

Ahuja, N., Singal, G., Mukhopadhyay, D., & Kumar, N. (2021). Automated DDoS attack detection in software defined networking. *Journal of Network and Computer Applications, 187*, 103108. https://doi.org/10.1016/j.jnca.2021.103108

Ali, T. E., Chong, Y. W., & Manickam, S. (2023). Machine learning techniques to detect a DDoS attack in SDN: A systematic review. *Applied Sciences, 13*(6), 3183. https://doi.org/10.3390/app13063183

Alzahrani, A. O., & Alenazi, M. J. F. (2021). Designing a network intrusion detection system based on machine learning for software defined networks. *Future Internet, 13*(6), 111. https://doi.org/10.3390/fi13060111

Çatak, F. O., & Mustacoglu, A. F. (2019). Distributed denial of service attack detection using autoencoder and deep neural networks. *Journal of Intelligent & Fuzzy Systems, 37*(5), 3969–3979. https://doi.org/10.3233/JIFS-179258

ClickZ. (2019). *Internet growth usage statistics*. Available online: https://www.clickz.com/internetgrowthusage-stats-2019-time-online-devices-users/235102/ (accessed on 10 January 2024).

Gaur, V., & Kumar, R. (2022). Analysis of machine learning classifiers for early detection of DDoS attacks on IoT devices. *Arabian Journal for Science and Engineering, 47*, 1353–1374. https://doi.org/10.1007/s13369-022-07116-3

Hossain, M. A., Sheikh, M. N. A., Rahman, S. S., Biswas, S., & Arman, M. A. I. (2018). Enhancing and measuring the performance in software defined networking. *International Journal of Computer Networks and Communications (IJCNC)*, *10*(3), 27-39. https://doi.org/10.5121/ijcnc.2018.10303

Liu, Z., Wang, Y., Feng, F., Liu, Y., Li, Z., & Shan, Y. (2023). A DDoS detection method based on feature engineering and machine learning in software-defined networks. *Sensors, 23*(13), 6176. https://doi.org/10.3390/s23136176

Mishra, A., Gupta, N., & Gupta, B. B. (2023). Defensive mechanism against DDoS attack based on feature selection and multi-classifier algorithms. *Telecommunications Systems, 82*, 229–244. https://doi.org/10.1007/s11235-023-00595-2

Mona, A., Waqas, K. Q., Muhammad, T., Muhammad, S., Mai, A., & Fazila, M. (2022). Machine-learning-based DDoS attack detection using mutual information and random forest feature importance method. *Symmetry, 14*(6), 1095. https://doi.org/10.3390/sym14061095

Njoku, C.E., Nwobodo, L.O., Shamshudeen, M.S. (2024). Improving Quality of Service on Wireless Network Using Artificial Neural Network-Based Firewall Intrusion System. Explorematics journal of Innovative Engineering and Technology 5(1),51-59. www.exploremacticsjournal.org.ng

Oyucu, S., Polat, O., Türkoğlu, M., Polat, H., Aksöz, A., & Ağdaş, M. T. (2024). Ensemble learning framework for DDoS detection in SDN-based SCADA systems. *Sensors, 24*, 155. https://doi.org/10.3390/s24010155

Praba, J., & Sridaran, R. (2022). An SDN-based decision tree detection (DTD) model for detecting DDoS attacks in cloud environment. *International Journal of Advanced Computer Science and Applications, 13*(7). https://doi.org/10.14569/IJACSA.2022.0130777

Raza, M. S., Sheikh, M. N. A., Hwang, I.-S., & Ab-Rahman, M. S. (2024). Feature-selection-based DDoS attack detection using AI algorithms. *Telecom, 5*(2), 333–346. https://doi.org/10.3390/telecom5020017

Santos, R., Souza, D., Santo, W., Ribeiro, A., & Moreno, E. (2020). Machine learning algorithms to detect DDoS attacks in SDN. *Concurrency and Computation: Practice and Experience, 32*(16), e5402. https://doi.org/10.1002/cpe.5402

Sharafaldin, I., Lashkari, A. H., Hakak, S., & Ghorbani, A. A. (2019). Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. In *Proceedings of the 2019 International Carnahan Conference on Security Technology (ICCST)*, Chennai, India, 1–3 October 2019 (pp. 1–8).

Sheikh, M. N. A., Hwang, I. S., Ganesan, E., & Kharga, R. (2021). Performance assessment for different SDN-based controllers. In *Proceedings of the 2021 30th Wireless and Optical Communications Conference (WOCC)*, Taipei, Taiwan, 7–8 October 2021 (pp. 24–25). https://doi.org/10.1109/WOCC53343.2021.9624939

Singh, J., & Behal, S. (2020). Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges, and future directions. *Computer Science Review, 37*, 100279. https://doi.org/10.1016/j.cosrev.2020.100279

Sridaran, R. (2022). An SDN-based decision tree detection (DTD) model for detecting DDoS attacks in cloud environment. *International Journal of Advanced Computer Science and Applications, 13*(7). https://doi.org/10.14569/IJACSA.2022.0130777

Tsamardinos, I., Borboudakis, G., Katsogridakis, P., Pratikakis, P., & Christophides, V. (2019). A greedy feature selection algorithm for big data of high dimensionality. *Machine Learning, 108*(2), 149–202. https://doi.org/10.1007/s10994-019-05891-0

Wang, Y., Wang, X., Ariffin, M. M., Abolfathi, M., Alqhatani, A., & Almutairi, L. (2023). Attack detection analysis in software-defined networks using various machine learning methods. *Computers and Electrical Engineering, 108*, 108655. https://doi.org/10.1016/j.compeleceng.2023.108655